# Locus Chain Tech Whitepaper

**Posted: May 6, 2024**

**Revised: May 20, 2024**

**Revised: May 31, 2024** (Correcting minor typos)

## Contents

## 1. Important Notice

We kindly request your attention to the following important notice. It is advisable to carefully review this section. Should you have any inquiries regarding the necessary actions to be taken, we recommend consulting with your legal, financial, tax, or other suitable professional advisor(s).

It is imperative to note that the information provided in this whitepaper is subject to potential adjustments or updates and should not be interpreted as an assurance, commitment, promise, or guarantee by LOCUS CHAIN or any entity or individual referred to in this document regarding the anticipated availability of services related to token usage, or the future value or performance of tokens.

This whitepaper does not aim to solicit or advise the sale of stocks or securities and should not be utilized as the basis for any agreements or commitments. It is not an offer to sell, subscribe to, or purchase any securities. Locus Chain explicitly disclaims responsibility for any direct or consequential losses or damages incurred directly or indirectly due to reliance on, or any errors, omissions, or inaccuracies within, the information presented in this whitepaper, or any subsequent actions taken as a result.

It is essential to emphasize that this whitepaper is not designed for the purpose of soliciting purchases or providing financial guidance; it is meant solely for informational purposes. Under no circumstances should trading or investing in assets such as tokens, coins, or company stocks be based solely on the provided information. As with all investments, there exists a risk of price fluctuations and potential loss of principal. Investors are urged to seek the guidance of professional financial, legal, and tax advisors to conduct their own research or due diligence concerning the topics or issues outlined in this whitepaper before making any investment decisions.

The contents of this document are compiled from sources believed to be reliable and accurate. While market prices, data, and other information are not guaranteed to be complete or accurate, they are derived from selected public market data, reflect current conditions and our perspectives as of the current date, and thus are subject to possible modifications without prior notice.

It is noteworthy that this document may encompass or reference forward-looking statements, which entails statements that do not constitute historical facts. No representations or warranties are offered regarding the accuracy of such forward-looking statements. Any projections, forecasts, and estimates included in this document are inherently speculative and are contingent upon certain assumptions. It is essential to acknowledge that these forward-looking statements may prove inaccurate and can be influenced by flawed assumptions, as well as known or unknown risks, uncertainties, and other factors, most of which may be beyond our control. It is foreseeable that some or all of these forward-looking assumptions may not materialize, or may differ significantly from actual outcomes.

# 2. Introduction

## A. Locus Chain Business Overview

Locus Chain is a public distributed ledger protocol facilitating tamper-proof communication across multiple parties with exceptional efficiency, characterized by large-scale capacity, high-speed transactions, and low costs. It operates as a public blockchain protocol capable of real-time processing, even amidst substantial transaction volumes and growing participation. To achieve this, it integrates DAG-AWTC + BFT and Dynamic Sharding, alongside Verifiable Pruning technologies, ensuring the ledger's structural integrity, consensus mechanism, and minimizing storage space and network load requirements.

Locus Chain's remarkable scalability and ultra-lightweight nodes empower it to achieve infinitely scalable, fully decentralized processing of trust, ownership, and certification for all Internet data, all achieved algorithmically without reliance on third-party intervention. As a decentralized technology, Locus Chain securely processes authenticated data at high speeds, free from bottlenecks or distortion, thus fostering an enhanced data value ecosystem compared to conventional internet services.

### (1) Business Vision

Our vision entails forging an entirely new dimension within the decentralized (Web3) data ecosystem via a vast, scalable public blockchain network known as Locus Chain. Here, data assumes intrinsic value and facilitates direct exchange and trade between providers and users across diverse applications, catalyzing the emergence of fresh business domains and the realization of a new credit society. Leveraging Locus Chain as our foundation, we strive to pioneer a distinctive convergence of existing businesses with limited interconnectivity, thereby propelling the evolution of the data ecosystem through successive convergence phases.

### (2) Development Philosophy

From its inception, our objective is to revolutionize the foundation of social credit by fostering broad multilateral trust without intermediaries, leveraging the expansive scalability of Locus Chain within a highly decentralized network.

### (3) Performance of Locus Chain

Our goal is to establish a truly public platform where people can engage in ledger writing without incurring extra expenses on standard home PCs and network setups. Specifically, the design targets CPU occupancy of less than 5%, network bandwidth occupancy within 5% based on 100Mbps, on-device ledger size within 10GB, and TPS (Transactions Per Second) of at least 4k. Scalability is limited only by network speed since there is no inherent limitation in blockchain processing speed. The processing time per transaction is targeted to range from near-instant to just a few seconds at most. Locus Chain has developed and combined the following proprietary technologies to achieve these goals.

### (4) Applications

Locus Chain's decentralized technology facilitates secure, distortion-free, and high-speed processing of authenticated data without the need for servers. It is also possible to transform most existing internet services to operate in a decentralized environment. This not only enhances current services but also fosters the creation of a new data value ecosystem, paving the way for innovative applications and services built upon it.

By significantly reducing transaction costs for data exchanging, including data with no trading value due to its higher cost compared to worth, a decentralized environment provides an innovative economy for data exchange.  As a lightweight Add-On protocol capable of concurrently supporting diverse applications, it significantly streamlines the creation of decentralized applications and services. In particular, by leveraging the verification function of the Locus Chain, data scattered across devices can be authenticated solely through the chain, providing a performance and structure suitable for On-Device services where data security and sovereignty are crucial.

## B. Locus Chain Technology Overview

Locus Chain is a Layer 1 blockchain primarily dedicated to high-speed, high-performance transaction recording and consensus functions.

The processing capabilities of prominent blockchain systems such as Bitcoin and Ethereum typically reach around 20 transactions per second (TPS). This translates to approximately 20 individuals being able to record information on the blockchain every second. However, when compared to standard real-world benchmarks, this level of performance falls short. For example, the transaction volume of VISA, a global credit card brand, amounts to about 4,000 transactions per second.

However, attempting to increase the throughput of blockchain systems encounters various technical challenges. One significant challenge is the escalation in ledger size and communication volume due to the increased amount of data processing. For instance, if Locus Chain were to achieve a throughput akin to VISA's 4,000 transactions per second, the daily influx of data would total roughly 350 gigabytes. Such an immense volume rapidly depletes the storage capacity of a typical PC's 1TB storage within just three days, imposing a notable strain on personal device processing.

To tackle these impediments in processing speed and capacity, Locus Chain has devised advanced technologies for implementing a public distributed ledger protocol with extensive scalability within a highly decentralized environment. These technologies encompass DAG AWTC + BFT, Verifiable

Pruning, and Dynamic Sharding. As of April 25, 2024, Locus Chain has filed ten domestic and international patents, with four patents successfully granted and six patents under review. Additional patents for multiple technological advancements are also in the pipeline for filing. However, explanations concerning the technologies currently undergoing patent application preparations have been omitted from this whitepaper update, as of the publication date. Detailed status updates on Locus Chain's domestic and international patent applications and registrations as of May 6, 2024, are provided in Annex 1 and Annex 2, respectively.

### (1) Innovation in Ledger System and Consensus Mechanism for Enhanced Processing Performance

Locus Chain has adopted DAG-AWTC to address the speed limitations imposed by consensus mechanisms. The removal of these constraints exposes limitations in network bandwidth, necessitating the implementation of sharding. Traditional multi-endpoint ledger systems utilizing DAG have struggled to integrate BFT Byzantine Fault Tolerance consensus, posing a barrier to scalability through sharding. As a foundational technology enabling sharding and pruning, Locus Chain employs a consensus mechanism comprising an AWTC ledger system based on Directed Acyclic Graph (DAG) information structure and Byzantine Fault Tolerance (BFT) consensus driven by Weighted Proof of Stake (PoS). Locus Chain has integrated DAG-AWTC from the early stages of development, envisioning Dynamic Sharding and developing a consensus mechanism based on algorithmic technology. The DAG + BFT technology, pioneered by Locus Chain, was patented as of August 1, 2019, marking a significant milestone as the world's first in this domain, with the patent application and registration successfully completed.

### (2) Enhancing Processing Performance and Ensuring Fairness through Sharding

In the realm of computer technology, sharding emerges as a prominent approach to tackle processing capacity limitations. Essentially, sharding involves dividing a large workload into smaller, more manageable partitions for efficient processing. While extensively researched and applied in established fields like databases and networks, its adoption in the blockchain domain remains relatively uncommon.

Locus Chain addresses the challenge of processing scalability through **dynamic sharding** technology, which automatically partitions the ledger and network into fair segments based on processing capacity.

Each shard within Locus Chain is designed to handle a comparable volume of transactions. Should transactions become concentrated within a single shard, the system redistributes the workload equitably across other shards through shard reorganization. Consequently, participants in Locus Chain can earn mining rewards proportionate to the fair processing burden, regardless of which shard they participate in or what transactions they handle.

Through Dynamic Sharding, Locus Chain achieves scalability by partitioning the blockchain network into multiple shards as required, reducing network load and enabling throughput expansion by tens to thousands of times compared to traditional methods. By dynamically adapting to increased usage, Locus Chain mitigates blockchain network scalability and security issues, thus equipping it with unlimited scalability. Dynamic Sharding technology was also developed by Locus Chain for the first time in the world, and as of April 25, 2024, two out of eight patent applications filed from June 3, 2022, to November 3, 2022, have been granted, while the remaining six are under review.

### (3) Reducing Information Storage through Pruning

In blockchain systems, processed information is structured into a ledger, a fundamental component within the blockchain network. Previously, computers participating in the blockchain network were required to record and store the entire ledger on local devices. However, in the context of high-speed, large-capacity processing envisioned by Locus Chain, the sheer volume of information becomes impractical to store on typical PCs.

Locus Chain addresses this challenge through **ledger pruning**, a method involving the removal of less frequently accessed historical ledger data from local computers.

Locus Chain adopts a Directed Acyclic Graph (DAG) ledger structure known as AWTC. Its proprietary **Verifiable Pruning** leverages this graph ledger structure to mathematically assess the extent to which the latest ledger state correlates with past transactions. This enables the validation of current and newly received information, even after removing the majority of irrelevant transactional data from the past.

This significantly diminishes the hardware expenses associated with the node devices engaged in ledger creation. Alongside sharding for ledger and network load reduction, this technology substantially downsizes the ledger. While most blockchains necessitate nodes to operate on high-performance computers, Locus Chain enables node operation on low-end devices like mobile devices. Locus Chain is the pioneering developer of Verifiable Pruning technology and applied for and obtained a patent on August 1, 2019.

### (4) Enhancing Blockchain Ecosystems Through Contract Extension Plugin (VME)

The Contract Extension Plugin serves as an Interface Architecture Solution within the Locus Chain Ecosystem, facilitating the integration of external Smart Contracts and server-based projects. Locus Chain ventured into developing this expansion pack during its 1.0 development phase upon identifying a gateway that significantly enhances the platform's versatility. This strategic pivot, extending beyond the initial scope, marks a pivotal structural transformation driving Locus Chain's evolution, offering innovative capabilities and strengths.

Locus Chain's Smart Contract engine features a **"plugin" VM structure** tailored to execute related Smart Contract transactions independently amidst the high transaction throughput of thousands per second.

Moreover, it boasts a network layer primarily leveraging P2P communication for seamless information exchange within and between shards. This network layer not only supports Locus Chain's ledger and consensus mechanism but also functions as an underlying structure within the Smart Contract engine, enabling the execution of Smart Contracts based on ledger-recorded transactions.

By integrating Smart Contracts from other blockchain mainnets, such as Ethereum, into the Locus Chain core engine via the Contract Extension Plugin, they seamlessly operate as sub-blockchain projects within the Locus Chain ecosystem. Projects spanning finance, smart cities, gaming, and beyond can function as subsystems through VME or directly on the core engine, diversely integrating into the Locus Chain economy based on their unique requirements.

Through the implementation of such technology, Locus Chain is currently in the process of applying blockchain systems to real-world applications, utilizing them as communication networks and state storage databases.

### (5) Expanding the Boundaries: A Versatile Platform in Blockchain Technology

Many blockchain projects are confined to usability within projects relying solely on distributed ledgers. Locus Chain breaks this mold by facilitating integration with server-based systems into the core engine through VME. While maintaining complete decentralization, it surpasses the limitations of the blockchain ecosystem by accommodating server-based systems, showcasing remarkable versatility.

Locus Chain serves as a core protocol enabling the operation of microdevices and other edge devices for unlimited and efficient information processing and exchange. Presently, groundwork is being laid for the distribution of an SDK, which will empower external developers to create projects based on the Locus Chain Protocol.

Upon completion of these endeavors, the versatility of Locus Chain is expected to extend to encompass all blockchain projects. Moreover, server-based systems will seamlessly integrate with Locus Chain, a fully decentralized public blockchain, through expansion packs, broadening the application scope from distributed ledger-based systems to encompass server-based systems and beyond.

# 3. Locus Chain Ledger and Transaction Structure

## (1) Ledger

Much like traditional blockchains, the ledger of Locus Chain serves as a decentralized repository where participants can freely append information. In a narrow scope, the ledger comprises the cumulative transactions issued and added by participants in the blockchain. Expanding its definition, it encompasses consensus-related data generated to validate transactions and operational information pertinent to the blockchain's functioning.

## (2) Account

An account in the Locus Chain ledger denotes an entity empowered to actively contribute information by initiating transactions. Each account possesses a cryptographic key pair—a private key and a public key—utilized to sign the transactions it initiates. Accounts are uniquely identifiable through addresses derived from their public keys.

Account addresses and public keys are publicly accessible data, whereas the private key remains confidential to the account owner. Accounts leverage their private keys to execute actions that authenticate their identity, such as issuing transactions.

## (3) Account State and Transactions

The account state denotes the current status of internal information within an account. One significant aspect of the account state is the balance of coins held by the account. Additionally, transactions allow for the generation and updating of arbitrary information values.

Changes to the account state can only be initiated by the account owner. These alterations are occurred through transactions, encapsulating the modifications. For instance, if Account A intends to transfer 10 coins to Account B, the owner of Account A issues a transaction containing the necessary details to adjust the balance in the account state.

```
{
  issuer: A,   // the issuer
  tx_no: 4,    // transaction number
  partner: B,  // related Account
  pseudo_command: {
    MOVE coin(10) FROM A TO B;
  }
}
```

*Figure 1: Example of a Transaction Issued by Account A*

While the details of the account state are public, not every value within it is recorded in the ledger. However, transactions may include representative values indicating when and what changes occurred, enabling participants to verify the accuracy of the account state they are aware of by scrutinizing transaction records.

### (4) Message Transactions

In Locus Chain, transactions serve a role akin to messages exchanged between accounts.

Considering the provided example, upon A's transaction issuance, it is assumed that A's coin balance diminishes. However, as only the owner of Account B can alter its state, immediately after A's transaction issuance, B's coin balance remains unaffected. To augment Account B's coin balance, its owner must explicitly issue a transaction for the increase in coins. In this scenario, A's initial transaction acts as a reference for the coin increment.

```
{
  issuer: B,      // the issuer
  tx_no: 7,       // transaction number
  input: [A:4],  // reference to other TX
  pseudo_command: {
    ADD coin(10) USING [A:4]
  }
}
```

*Figure 2: Example Transaction Issued by B*

Upon such transaction issuance, the amount of coins in B's account state is updated, and A's transaction fulfills its purpose. From B's perspective, A's transaction serves as a message indicating the transfer of coins to B.

Viewing Locus Chain transactions as messages exchanged between accounts, the ledger in Locus Chain can be perceived as an intermediary form between Bitcoin's UTXO ledger and Ethereum's account-based ledger. In Locus Chain, transactions involving two or more accounts remain unspent until reaching the recipient account. Once the recipient account receives and validates the transaction, it transitions to a spent state. However, unlike Bitcoin's UTXO model, where outputs are received and produced, inputs in Locus Chain are not transactions but rather shared account states, thus negating the possibility of double spending at the transaction level. Nonetheless, it is conceivable that adversarial participants may issue conflicting transactions involving the same account state simultaneously. In such cases, the account state perceived by different nodes may diverge based on the received transactions, potentially leading to scenarios interpreted as deliberate attempts at double spending.

### (5) Transaction Numbers

In Locus Chain, the state of an account evolves with each transaction it undergoes, and each of these transactions is assigned a with a sequence number (a unique identifier) known as the transaction number. These transaction numbers provide a chronological order to the changes in an account's state, allowing for effective tracking and verification. The transaction numbering system commences with 0, allocated to the inaugural transaction initiated upon the creation of an account. Subsequent transactions are sequentially numbered, with each new transaction incrementing the transaction number by 1.

This systematic numbering scheme facilitates the detection of potential double-spending incidents. If multiple transactions within the same account share identical transaction numbers, it raises a red flag for potential double spending.

### (6) Locus Chain Account-wise Transaction Chain (AWTC)

Transactions in Locus Chain are issued one at a time, with each newly issued transaction automatically establishing a reference link to the preceding transaction from the same account. These transactions from a single account can seamlessly link together, forming what is known as the Account-wise Transaction Chain (AWTC). Essentially, each account maintains its distinct transaction chain, allowing for the unrestricted addition of transactions at any given moment.

Furthermore, as depicted in the previous example of coin transfer, a transaction can refer to transactions issued by other accounts. As these referenced transactions always pertain to past transactions in terms of time, these references function as unidirectional links, creating a Directed Acyclic Graph (DAG) structure.

A transaction can reference multiple transactions, and it can also specify multiple related accounts. Consequently, the relationships between transactions construct a DAG, with each transaction node potentially having multiple input and output edges.

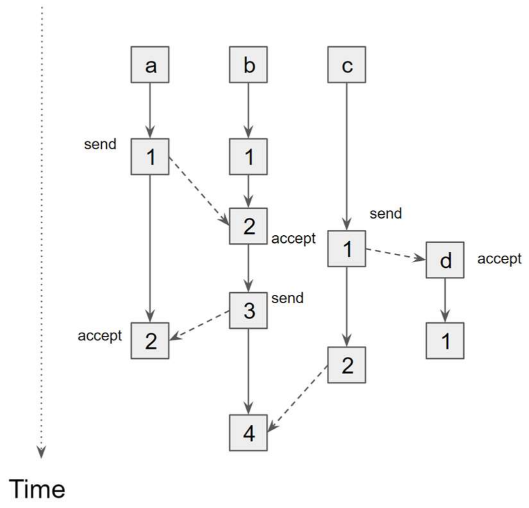


*Figure 3: AWTC Chain DAG Configuration*

The ledger of Locus Chain consists of a collection of transactions, thereby forming a DAG ledger composed of account AWTCs that reference each other. Each account's AWTC in the DAG ledger adheres to a single rule: transaction numbers must be consecutive and unique. If this rule is maintained, each account is free to issue transactions as needed.

### (7) Partitioning of Shard Ledger

Locus Chain's ledger can be partitioned into shards as required, with each shard managing a shard ledger.

Transactions are allocated to shards at the account level. This means that an account is consistently linked to a specific shard, and all past transactions initiated by that account are regarded as part of the ledger associated with that shard.

Locus Chain's dynamic sharding enables shard reorganization. Since transactions are grouped at the account level, shard reorganization involves transferring entire accounts to different shards. Therefore, during shard reorganization, accounts are moved between shards, and all transactions related to the moved accounts are relocated from the ledger of the original shard to the ledger of the new shard.

#### Inter-shard Transaction Routing

While transactions are always recorded in the ledger of the corresponding shard, there is a necessity to route transactions to other shards for inter-shard data exchange. For instance, when data exchange occurs between two accounts belonging to different shards, transactions issued by each account must be routed to the respective shard of the recipient in a verifiable format.

### (8) Node and Network Shard

In Locus Chain, sharding is a process where the ledger is logically divided, and the network is physically partitioned (“Sharded”). This means that nodes participating in the network are grouped into shards, where the primary exchange and sharing of shard ledger information occur among nodes within the same shard. The number of network shards corresponds to the number of ledger shards in Locus Chain, with each network shard being responsible for managing one ledger shard.

### (9) Node

Nodes in Locus Chain are computer programs connected to the internet. They communicate with other nodes and exchange information using a peer-to-peer (P2P) approach.

Each node is associated with a Locus Chain account, known as the node owner or representative account. This account is always linked to a specific shard, and the node belongs to the shard to which its owner account is assigned.

All nodes participating in the Locus Chain network have the responsibility and obligation to perform basic transaction transmission and verification roles. Unlike other blockchains, Locus Chain does not include light nodes that perform limited functions.

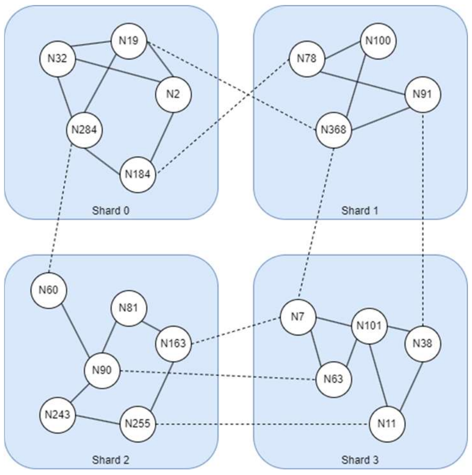


*Figure 4: Locus Chain Node Sharding and P2P Communication Concept Diagram*

### (10) Peer-to-Peer Communication Network

In Locus Chain, nodes communicate directly with each other through peer-to-peer (P2P) communication. This P2P communication operates at a standard level, wherein information generated by a node is disseminated to all connected neighboring nodes, and information received from one node is relayed to others. Unless there's a specific reason not to, nodes generally respond to requests for information from other nodes if they possess the requested data.

Each node typically maintains connections with around 4 to 8 other nodes through P2P connections, which remain active for relatively extended periods, ranging from a few minutes to several minutes. Approximately half of these connections are within the same shard, while the remaining connections link nodes across different shards.

### (11) Partitioning Shard Network

When a node establishes a new connection with another node, it can identify the shard affiliation of the peer node based on its representative account information. By routing information differently based on the shard affiliation of the peer node, the network effectively partitions into shards.

For instance, newly generated transactions within a shard are uniformly distributed to nodes within the same shard but aren't typically broadcast to nodes in other shards by default. However, if a new transaction involves an account in a different shard and there are connected nodes leading to that shard, the transaction is forwarded accordingly.

# 4. Consensus Algorithm

## A. Shard Consensus

The shard ledger represents a compilation of transactions from accounts within a specific shard. Whenever an account in a shard initiates a transaction, it must be incorporated into that shard's ledger. To ensure this process, it's vital to confirm that transaction information has been reliably disseminated to most nodes within the shard.

The division of the ledger into shards in Locus Chain is a crucial aspect to consider. Information is not inherently shared across different shards. Thus, for a shard to utilize a transaction from another shard, it needs to receive sufficient validation information (proof) to verify its validity.

This validation information must remain immutable and finalized once provided. Locus Chain achieves this by employing a method that achieves a complete and unchangeable block state through the consensus of multiple nodes.

In Locus Chain, transactions issued within each shard are aggregated to form consensus blocks, which are conclusively determined using a Byzantine Fault Tolerance (BFT) consensus algorithm. This process of block creation and confirmation within a shard is known as shard consensus.

The generation of consensus blocks in Locus Chain involves executing a BFT consensus algorithm with multiple leader nodes. Nodes participating in consensus are selected using a Weighted Proof-of-Stake mechanism through a random function.

Before reaching the BFT consensus stage, in the event of detecting duplicate double-spending transactions, a convergence voting consensus involving all nodes within the shard is conducted during the transaction propagation phase to resolve such conflicts.

Each shard's consensus operates independently.

### (1) Pre-consensus Stage: Evaluation of Transaction Validity

Accounts generate transactions by packaging information into a transaction format and signing it. This process allows accounts to potentially create transactions with arbitrary content, opening the door to the possibility of dishonest transactions, such as double-spending. Within blockchain systems, only transactions that align with the historical context of the agreed-upon blockchain are considered valid. Honest nodes within the network are responsible for including only valid transactions in their processing, while disregarding or rejecting any deemed invalid. Similarly, in the context of Locus Chain, transactions' validity is judged in a reasonable manner within blockchain technology.

In Locus Chain, each account and node follow formal and procedural methods to determine the validity of transactions. Initially, all transactions must be signed using the private key of a valid account. Moreover, transactions originating from the same account must be assigned a sequential number, starting from zero and incrementing by one.

Transactions that have not yet reached consensus must be assigned consecutive numbers, beginning from the last agreed-upon transaction of the account. In cases where multiple transactions share the same number, the consensus process selects one transaction for inclusion.

Once consensus is achieved, transactions must have their information incorporated into the agreed-upon shard blockchain and world state chain.

The arbitrary content included by an account in a transaction is not utilized in the determination of its validity within Locus Chain. While transactions may contain arbitrary information, it's neither appropriate nor feasible for participants to evaluate this content from an external (Third-Party) perspective. However, information defined and publicly known within Locus Chain, such as coin amounts, is utilized in the validity assessment process.

### (2) Duplicate Detection and Converging Voting Competition in Transaction Transmission

Newly issued transactions are disseminated within a shard via P2P communication. If a node receives distinct transactions sharing the same account and transaction number, these transactions are flagged as duplicates. Among duplicates with identical transaction numbers, only the transaction most widely propagated across the shard is considered valid, while the other is deemed invalid and discarded.

A converging voting competition, initiated upon the occurrence of duplicate transactions during transmission, serves as a mechanism to select a single transaction from duplicates. This process engages all nodes within the shard and commences asynchronously upon the detection of duplicate transactions.

#### Duplicate Transaction Detection Voting Message

When a node encounters two or more transactions within the same round featuring the same account and transaction number, it broadcasts a duplicate detection voting message across the shard via the P2P network. This message includes details of the duplicate transactions identified by the node and indicates which transaction it received first.

By aggregating duplicate detection messages generated across the shard, nodes can roughly discern the transaction that has been most widely disseminated. Nodes prioritize the transaction that has garnered the most votes based on the aggregation, rather than the one received first. Nodes tasked with generating block candidates utilize the aggregation of these duplicate detection messages during block creation to select the transaction with the highest vote count among duplicates for inclusion in the block.

### Confirmation of Transactions Without Duplicate Detection Messages

If two duplicate transactions occur with a significant time gap between them, the first transaction essentially takes precedence. Due to the nature of P2P communication, transactions are typically propagated throughout the entire shard after approximately $D$ steps, where $D$ represents the diameter of a random graph. Consequently, once the first transaction has reached most nodes after $D$ steps, it's highly likely to be acknowledged as the primary transaction in duplicate detection messages triggered by subsequent transactions. Therefore, it becomes practically unfeasible for subsequent transactions to alter the aggregation results of duplicate detection messages.

As a result, honest nodes can consider a transaction confirmed and proceed with processing if no duplicates occur for a significant period after its reception. In Locus Chain, the time required for a transaction to be sufficiently propagated within a shard is set to around 10 seconds. In essence, if no duplicate detection messages are triggered within this timeframe of receiving a transaction, it's deemed practically valid and confirmed for processing even before the initiation of block creation consensus.

While most user applications may suffice with treating transactions as valid once they have been confirmed for a practical duration, establishing a benchmark for sharding and pruning necessitates recording an irreversible final state of the ledger within the shard. To achieve this, Byzantine Fault Tolerance (BFT) consensus is employed to ensure consensus on the entire ledger state within the shard.

## (3) Assumptions for Locus Chain's BFT Consensus

In executing the consensus algorithm of Locus Chain among nodes existing on the actual internet, the following communication assumptions are expected.

### Stability of P2P Communication

Locus Chain relies on P2P communication for exchanging information like transactions. Thus, Locus Chain assumes that once transactions are issued, they can be swiftly disseminated to the majority of nodes within the shard via P2P communication in a short timeframe. Specifically, the interconnectivity among nodes within the shard resembles that of a random graph, and the distance between any two nodes within the network falls within the diameter ($D$) of the random graph. Consequently, it's expected that information originating from any node will propagate to all nodes within the shard within $D + \alpha$ steps of communication.

### Accuracy of Node Clocks

Locus Chain's consensus algorithm incorporates real-time-dependent processes such as timeouts. Therefore, it's assumed that the majority of nodes participating in Locus Chain possess reasonably accurate real-time clocks. More precisely, node clocks are assumed to exhibit an error margin that is handled within network communication timeouts, with practical error expectations within 10 seconds. Nodes with significantly inaccurate timekeeping are deemed Byzantine.

#### Distribution of Honest Nodes

While nodes in Locus Chain may exhibit adversarial behavior like issuing fraudulent transactions or engaging in dishonest voting during consensus, it's presumed that a sufficient proportion of nodes act honestly for the system's functioning.

Specifically, in the consensus process, it's assumed that more than $\frac{1}{3}$ of randomly selected nodes are consistently honest. Locust Chain's consensus process accommodates consensus failures and retries, prioritizing the prevention of reaching consensus with incorrect content over achieving consensus with complete content. If incorrect consensus can be averted, block chain creation proceeds through retries. If over $\frac{1}{3}$ of the total voting nodes are honest, it can prevent incorrect content from prevailing in consensus.

### (4) Round State (RS) Consensus Block

A consensus block serves as a structural component for finalizing transactions. While resembling traditional blockchain blocks in many aspects, it diverges by not directly encompassing transactions but solely containing the root hash of the shard. It's alternatively denoted as a round block or round state, contingent on the context.

The pivotal data within a block comprises the list of hash values of generated transactions. Furthermore, it incorporates details representing the state of the blockchain and shard, encompassing the hash value of the preceding block, alterations in shard-wide stake distribution, and the chronicle of account modifications.

Blocks attain finality through cryptographic signatures, ensuring an appropriate and verifiable confirmation process.

### (5) RS Consensus Time Units

#### Consensus Round

In Locus Chain, blocks are generated at specific time intervals. Currently, one block is generated every 2 minutes. In other words, the consensus algorithm executes approximately every 2 minutes, encapsulating transactions accrued over this duration within each block. This 2-minute cycle is termed a ***Round***.

Following the conclusion of a round, the consensus algorithm commences after a 40-second interval. This buffer period allows ample time for transactions occurring just before the round's termination to propagate adequately within the shard.

### Epoch

Another temporal metric is the epoch, which is a unit used for convenience to stabilize the internal state of the shard. For example, the stake proof weight used to select consensus nodes is determined at the beginning of each epoch and remains unchanged throughout the epoch. Currently, each epoch spans 24 hours, equivalent to 720 rounds.

## (6) RS Consensus Committee

Typically, around 50 nodes engage in each consensus round. The selection of consensus committee nodes is primarily based on a weighted random process.

### Types of Consensus Participant Nodes

The nodes involved in consensus encompass proposer nodes, voter nodes, or committee nodes. Proposer nodes take on the role of generating candidate blocks, while committee nodes execute the consensus algorithm on these generated blocks, producing signatures to finalize suitable blocks.

### Selection Process for Consensus Participant Nodes

Determining whether a particular node acts as a proposer node relies on factors like a fixed value derived from stake changes up to the start of the epoch, the current round number, and the node's representative account address. This is executed by comparing the computed selection values of each node. Additionally, for the selection of committee nodes, actual participation records within the committee are considered.

Each node calculates its own "selection value" for each round in a verifiable manner. If this value surpasses a certain threshold, it is disseminated within the shard via P2P communication. Any node can verify these selection values. Nodes validate and sort the selection values of other nodes based on appropriate criteria, selecting a new set of nodes accordingly. While a specific number of committee nodes are chosen, the count of selected proposer nodes may fluctuate based on the round.

Once selected, nodes retain their committee eligibility for a set number of rounds. Presently, nodes remain eligible to participate in the committee for 7 rounds. Consequently, approximately 14.3% of the total committee nodes undergo changes with each round.

### Communication During Consensus

Nodes within the consensus group communicate closely, mirroring direct communication between nodes.

Nodes not partaking in consensus lack visibility into the communications occurring during the process. They can only observe the final consensus outcome.

## B. BFT Consensus Algorithm

The consensus algorithm of Locus Chain progresses through four stages. Participating nodes autonomously execute each stage according to real-time clocks, with each stage running at intervals of 10 seconds, without the need for separate communication for synchronization.

The consensus stages are as follows:

- Block proposer nodes generate block candidates.
- Voter nodes select a block through the first round of voting.
- Voter nodes finalize block selection through the second round of voting.
- Block Signing

### (1) RS Block Candidate Generation

After a round concludes, each block proposer node collects transactions received during that round to create block candidates.

Block candidates consist of the hash values of each transaction arranged in a Merkle Tree structure. Block proposer nodes organize the received transactions appropriately, listing the hash values of each transaction in the order they're arranged to generate a block.

If the received transactions are identical, the resulting block candidates are identical as well. For each round, a number of block candidates equal to the number of block proposer nodes are generated. In cases where all block proposer nodes receive the same transactions, all resulting block candidates are identical.

Block proposer nodes transmit the created block candidates to all voting nodes.

#### Detection of Inappropriate Transactions

Each candidate node excludes inappropriate transactions when creating a block.

First, the transaction number of each transaction is verified. The transaction number, a sequential identifier assigned to transactions by an account, must be contiguous. Thus, any account with duplicated or discontinuous transaction numbers indicates inappropriate transactions.

In cases of discontinuous numbers, transactions occurring after the discontinuity are disregarded.

For duplicated numbers, if a transaction with the same number exists in a previous block, the new transaction is dismissed.

In scenarios where different transactions with the same number are detected as duplicates within the same round, the transaction with the highest propagation through the aggregation of duplicate detection messages is chosen.

### (2) RS Block Selection Voting (First Voting)

The block candidate created by the block proposer node is disseminated to each voting node. Every voting node compares the received block candidate with the transactions it directly received to select the block that best fits its received transaction results. Subsequently, it signs the hash value of the selected block to produce a first voting message.

If the received block candidate includes transactions not received by the voting node, it may request those transactions from other nodes to update its received transactions and assist in block selection.

The generated first round block selection voting messages are then transmitted to other voting nodes.

### (3) RS Block Confirmation Voting (Second Voting)

Each voting node aggregates its own and other nodes' first voting messages, aiming to identify a block candidate with over $\frac{2}{3}$ of the total voting nodes' votes. If such a candidate exists, each voting node issues and signs a "round block confirmation message" for that block.

Should there be no block candidate garnering more than $\frac{2}{3}$ of the votes from the voting nodes, each voting node issues and signs a "round block confirmation failure message" for that round.

The signed round block confirmation and confirmation failure messages are propagated throughout the shard via P2P. Every node within the shard can receive and review each voting node's round block confirmation message.

### (4) RS Block Finalization

When any node within the shard receives round block confirmation messages issued by over $\frac{2}{3}$ of the total voting nodes for a specific block candidate, that block candidate is considered finalized as the block representing the state of the round.

To ensure the persistence of block finalization status, round block confirmation messages are included in the subsequent round's block.

## (5) Block Generation Consensus Failure Determination

If a node receives round block confirmation messages for a specific block from fewer than $\frac{2}{3}$ of the voting nodes and receives round block confirmation failure messages from over $\frac{2}{3}$ of the voting nodes, the consensus for that round is deemed failed.

Failure scenarios on a step-by-step basis may include:

### Failure to Generate Block Candidates

This could occur if all block proposer nodes fail to create a block, are unable to create one, or if the created block fails to propagate correctly.

If no block candidates are generated by the time round block state voting begins, voting nodes immediately issue round block confirmation failure messages. If over $\frac{2}{3}$ of the voting nodes within the shard issue such messages, the round block generation is considered a failure.

### Failure during Round Block State Voting

If voting nodes fail to validate all generated block candidates, they issue round block confirmation failure messages. Similar to scenarios where block candidates are not generated, if at least $\frac{1}{3}$ of the total voting nodes issue round block confirmation failure messages, it results in block confirmation failure.

### Failure during Round Block Confirmation Voting

If voting nodes fail to receive a sufficient number of round block confirmation messages, it indicates a breakdown in network reliability or the honesty assumption of the nodes, leading to the issuance of block confirmation failure messages.

## (6) Consensus Retry

If consensus for block generation fails, a retry is conducted in the subsequent round. In the next round, both the consensus from the previous round and the current round are simultaneously executed in parallel.

During a retry, temporarily increasing the number of newly elected consensus committee members can reduce the likelihood of adversarial node elections.

## C. World Round State (WRS) Consensus

Shard ledgers operate autonomously, managed and executed independently for each shard. Consequently, nodes lack direct means to verify whether transactions originating from other shards have been appropriately confirmed into blocks.

The process of World Round State (**WRS**) Consensus amalgamates the consensus results from each shard's rounds to produce a WRS Block. These WRS Consensus outcomes are disseminated across all nodes within the Locus Chain network and serve as a basis for validating both the legitimacy of round blocks in other shards and the transactions enclosed within.

### (1) WRS Block

The WRS Block is crafted by consolidating data from the shard round blocks generated by each shard.

WRS Blocks are generated in every round, encompassing hashes of all previously agreed-upon shard blocks from the preceding round, accompanied by their confirmation signature messages.

The agreed-upon WRS Blocks are distributed to all nodes.

### (2) WRS Block Consensus Algorithm

WRS Blocks are generated through the coordination of the WRS Consensus Committee.

The algorithm employed closely mirrors the shard round consensus process. Each round involves transmitting information about the shard round blocks agreed upon in the preceding round by all shards to the WRS Consensus Committee. This transmitted data includes the hashes of the round blocks and the round block confirmation messages. The WRS Consensus Committee, upon receiving this information, proceeds to sign WRS Consensus block confirmation messages for the generated block candidates, thereby effectively finalizing the blocks.

### (3) WRS Block Consensus Committee

The WRS Block Consensus Committee serves as an expansion of the shard consensus committee.

Members of the WRS Block Consensus Committee are drawn from the existing consensus committees of each shard. During each round, one shard is randomly chosen, and the consensus committee nodes of the selected shard are incorporated into the WRS Block Consensus Committee. Similar to the shard consensus process, these added nodes disengage after a predetermined number of rounds, currently set at 7 rounds.

# 5. Dynamic Sharding

Locus Chain addresses the scalability issues of blockchain through dynamic sharding. It partitions the ledger and network into shards based on transaction volume. As the total transaction volume increases, new shards are generated to prevent any single shard from becoming overloaded. Moreover, if the transaction volume in a shard fluctuates, shard reorganization occurs by transferring individual accounts between shards to maintain uniform shard sizes.

## (1) Locus Chain’s Dynamic Sharding

Dynamic sharding in Locus Chain primarily aims to enhance transaction throughput. The volume of transactions a shard can handle is limited by the size of the network and the number of participating nodes. Therefore, when the overall transaction volume of the system increases, it becomes necessary to increase the number of shards. Ideally, each shard should process an equal number of transactions.

Dynamic sharding in Locus Chain refers to the algorithmic adjustment of the number and size of shards to bring the entire network closer to an ideal state. This process involves measuring the network's state, adjusting shard sizes through node migration to achieve balance, and dynamically regulating the number of shards for optimal performance.

## (2) System State Evaluation

To perform shard partitioning and restructuring, it is essential to first assess the state of each shard and the entire world. The state of the world is represented by factors such as the number of nodes, transaction volume, and node stakes.

The workload of each shard is determined by its transaction volume. The throughput balance between shards can be determined from the transaction volume of all shards. Balancing the workload among shards involves redistributing the workload through node and account migration if the imbalance exceeds a certain threshold. Additionally, efforts are made to align the number of nodes and stakes across shards. If the overall workload of the world exceeds a certain threshold, additional shards are created.

The System state evaluation involves collecting information at epoch intervals and round intervals. Shards gather the following information near the end of each epoch for consensus and use it as a basis for subsequent epoch's PoS calculation.

- Number of accounts considered for consensus and sharding
- Stake amount of each account

Additionally, information such as transaction volume and the number of participating accounts is recorded at each round for consensus and is used to determine shard restructuring.

- Transaction Volume
- Number of participating accounts

### (3) Shard Reconstruction through Account and Node Migration

The decision on which accounts to include in each shard is determined based on transaction volumes between shards. If a shard experiences a high transaction volume, accounts with a higher probability of transactions are moved to shards with lower transaction volumes.

During the migration of accounts between shards, the corresponding AWTC ledger information is first copied to the destination shard and then removed from the original shard. Alongside the account migration, the associated node also relocates. The migrating node integrates into the new shard's network by obtaining information from it, excluding the data it already manages.

The decision to migrate nodes and accounts is made independently by each node. The collected world and shard information is shared among all nodes, and if consensus is reached based on this information, each node proceeds with account migration accordingly. If a malicious node refuses to comply with the consensus decision, it fails to exchange information with honest nodes, leading to its exclusion from the network.

Information regarding account and node migration is recorded in the round blocks of each shard through a consensus mechanism.

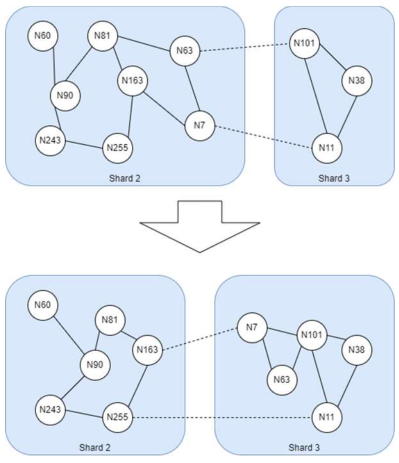


*Figure 5: Shard Reconstruction through Node Migration*

### (4) Shard Increment and Decrement

As transaction volumes in the network increase, expanding the overall transaction processing capacity requires an increase in the number of shards. Shard splitting, a method whereby existing shards are divided into two, is utilized for this purpose.

**Determining the Shard number for Splitting**

Each shard is assigned a sequential ID. When adding a new shard, its ID is incremented. To split a shard and create a new one, the most significant bit (MSB) of the new shard's ID is inverted. For example, if there are currently five shards numbered from 0 to 4, the ID of the newly added sixth shard becomes 5. To increase the number of shards, shard 1 (the second shard) with the most significant bit (MSB) of the binary representation 101 of decimal ID 5 inverted is targeted for partitioning. This method ensures that shard splitting occurs fairly and sequentially.

**Adjustment of Shard Size Before Splitting**

It's essential to ensure that the number of nodes after the split is of an appropriate size. Therefore, when shard splitting is decided, the target shard is inflated to approximately 1.8 times larger than the others.

Once the shard number to split is determined, nodes are gradually moved from other shards to the target shard over several rounds, increasing the number of nodes to about 1.8 times that of the other shards. Efforts are made to move nodes with fewer transaction occurrences to maintain a balanced transaction processing capacity among shards.

Splitting the target shard into two when the number of nodes increases ensures that each shard has a similar number of nodes after the split.

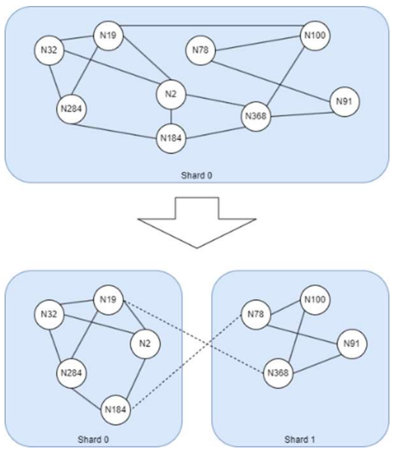


*Figure 6: Increasing Shard Count through Shard Splitting*

### (5) Shard Directory and Home Shard

As part of shard reconfiguration, both accounts and nodes may be reassigned to different shards in any given round. Consequently, when multiple accounts need to send transactions to other accounts, it's crucial to determine which shard the target account currently resides in.

In Locus Chain, an account directory management mechanism based on the account address is used to pinpoint the shard of an account.

Each shard is assigned a unique shard number (ID). The shard to which the account currently belongs is referred to as the working shard. Apart from the account's working shard, a fixed shard ID can be computed from the account address. For instance, dividing the account address by the current number of shards and taking the remainder yields a numerical value corresponding to the shard count. This calculated fixed shard ID for each account is termed the home shard. Even if an account moves between shards and its working shard ID changes, the home shard ID remains unchanged. Therefore, by managing directories of accounts designating a particular shard as their home shard, it's possible to determine the current shard of an account.

If a node wishes to determine the current working shard ID of a specific account, it can request communication (query) with nodes connected to it belonging to the home shard of the account in question, thus identifying the working shard of the account.

During an account's shard migration, ledger information is transferred between the shard before migration and the shard after migration. Simultaneously, an update request is made to refresh the working shard directory information for the account's home shard.

# 6. Verifiable Ledger Pruning

With a blockchain system capable of processing 1,000 transactions per second, each is containing 1,000 bytes of data, then the daily volume of ledger information amounts to 86.4 gigabytes. Even though sharding reduces the amount of information handled by each node to 1 / (number of shards), storing the data generated over several years becomes impractical for all nodes.

In Locus Chain, storage space can be conserved by implementing pruning, removing information from nodes' local storage that they and associated accounts do not directly require.

While pruning is not mandatory for all nodes, Locus Chain operates under the assumption that most nodes will engage in pruning by default.

## (1) Understanding the Relevance of Transaction Information

From the viewpoint of each account, the majority of information generated within the blockchain system and recorded in the ledger lacks direct relevance to the account itself. The ledger encompasses data concerning AWTC DAG transactions of each account and information related to block creation consensus, with the degree of relevance determined by the connectivity within the mathematical graph.

Accounts and nodes executing pruning can actively remove transactions with limited relevance.

### Account-Related AWTC Transactions

The ledger of Locus Chain consists of transactions linked in a time directional graph known as a Directed Acyclic Graph (DAG). Transactions initiated by an account always include a hash value pointing to the preceding transaction of the same account, and in some cases, may include additional links through hash values pointing to transactions outside of this scope. Within this DAG ledger structure, an account can trace back the links of past transactions referenced by its own issued transactions, thereby discerning all transactions relevant to itself. By calculating the distance in links from its own transactions to others, the level of association can be quantified.

Using this, an account can remove transactions from its AWTC ledger that are deemed beyond a certain degree of relevance. From the perspective of an account, most transactions are those with an infinite link distance (∞) and hold no reason for preservation by the node. Conversely, transactions with closer relevance must be retained.

### Ledger Consensus-Related Information

Nodes participating in block creation and consensus within a shard must validate all newly generated transactions within their jurisdiction. To verify a new transaction, it's essential to have information on past transactions referenced by the new one. The crucial reference for a new transaction is the preceding transaction of the account that issued it. Therefore, nodes participating in consensus should have access to the final issuance transactions of all accounts within their node for validation purposes.

Other transactions referenced by a new transaction, aside from the immediate preceding one, are acquired as needed. By ensuring nodes to acquire the transactions necessary for validation, accounts issuing new transactions must provide the required reference transactions and validation information along with the new transaction. This allows previously pruned past transactions necessary for validation to be (temporarily) re-referenced alongside the issuance of a new transaction, even if they have been completely pruned within the shard.

### (2) Verification of Pruned Past Transactions

In Locus Chain, when necessary past transactions for validating a transaction have been pruned and are not locally available, there's a need to acquire them afresh from other nodes to revalidate their legitimacy.

In traditional blockchains, transactions are organized in a linear linked list. Hence, to ascertain the validity of any given transaction, it's imperative to trace back through its referenced past transactions and forward through its dependent future transactions until reaching a known, confirmed transaction. This validation process demands a computational complexity of $(n)$.

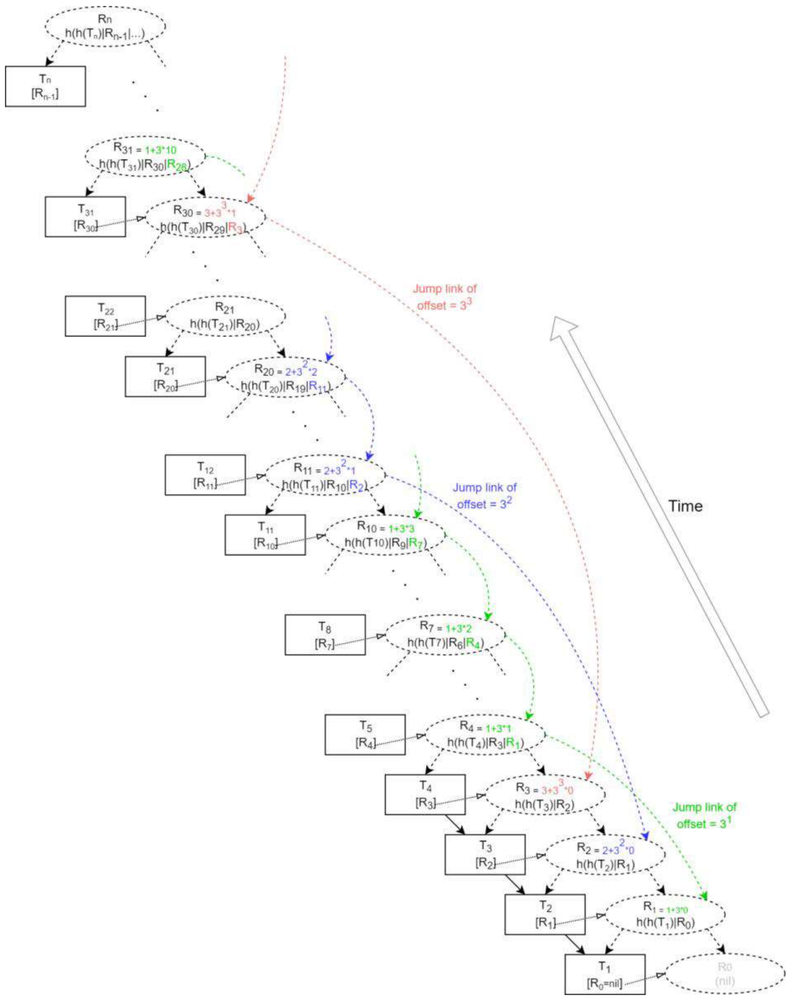
In Locus Chain, transactions are structured within a Hierarchical Skewed Merkle Tree, which employs an optimized data structure and verification algorithm. This allows for transaction validation with a computational complexity of $(\log n)$.

### (3) Hierarchical Skewed Merkle Tree

A Merkle tree is a tree data structure where each leaf node contains a cryptographic hash of the underlying data, and each non-leaf or branch node contains the cryptographic hash derived from the contents of its child nodes. Merkle trees are widely used for proving the existence of certain information in a list.

A simple Skewed Merkle Tree (**SMT**) is a type of Merkle tree where each internal node has two child nodes. One child node is a leaf node, and the other is the previous internal node. The Skewed Merkle Tree functions like a linked list based on the Merkle tree. When new information is added to the SMT, it is added on top of the existing root node by linking a new root node. The first child node of the new root node points to the newly added information (leaf node), and the second child node becomes the existing root node. Verification of nodes in the SMT involves hash verification of the Merkle tree. This is accomplished by comparing the hash value derived from the information of the child nodes with the hash value contained in the node. Since the verification of Merkle tree hash values is done recursively through the second link, it essentially requires computing the hash values of all past information.

In a Hierarchical Skewed Merkle Tree (**H-SMT**), each internal node has three child node. Similar to the SMT, two of these child node consist of a leaf node and the previous internal node. However, the third node holds the hash value of a child node that is not a leaf but rather an intermediary node. This third child node, termed the "jump link," facilitates traversing a linked list exponentially within the H-SMT structure.



Jump link of offset = 3^3
Jump link of offset = 3^2
Jump link of offset = 3^1
Time

*Figure 7: A Hierarchical SMT of base=3*

### Calculating Jump Link Distance

The jump link points to another node in the past Merkle tree. To determine how far back in the past the link should reference, the H-SMT employs the exponential power of a base distance $b$.

For an H-SMT with a base distance $b$, the distance $d$ that a Merkle tree node at height $n$ references with its jump link is calculated as follows:

$$\text{Distance } d = -\left(b^{(1+(n-1) \bmod b)}\right)$$

A node at height $n$ refers to nodes in the past calculated by $d$. For instance, with a base distance $b$ = 3, a node at height 4 references a node at height 1, which is $d$ = -($3^1$) distance away. Similarly, a node at height 11 references a node at height 2, which is $d$ = -($3^2$) distance away. Continuing this pattern, a node at height 30 references a node at height 3, which is $d$ = -($3^3$) distance away.

#### Verification of H-SMT Nodes

Using jump links, the H-SMT acts as a data structure akin to skip lists, but with verification conducted through Merkle trees rather than linked lists.

Thus, validating information contained within the H-SMT resembles finding the shortest path from the leaf node of that information to the root node of the H-SMT, identifying the Merkle hash values along this path.

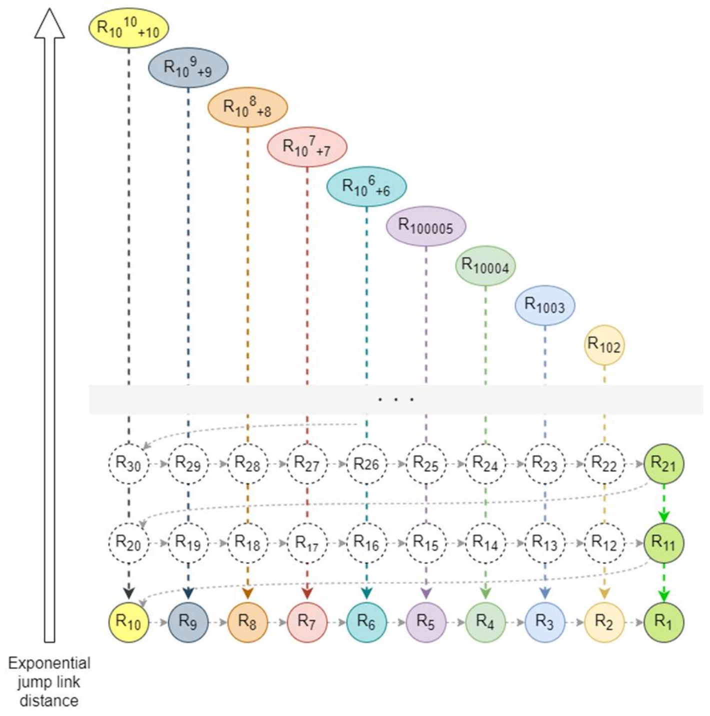
By leveraging jump links, it becomes possible to traverse a distance equivalent to a power of the base distance, significantly reducing the amount of information required for verification to approximately the logarithm of the overall graph height $h$.

### (4) Verifiable Pruning

Locus Chain implements transaction verification using H-SMT, allowing for the validation of any transaction with a computational complexity of (log $n$).

The information necessary for transaction validation can be readily generated by extracting relevant data held by the transaction issuer at the time of issuance or by successful verifiers. When transmitting a transaction to other accounts and nodes, including this validation information guarantees the transaction's legitimacy. This facilitates the exchange of information with assured legitimacy, irrespective of the pruning status on the recipient's end.

As an additional advantage of verifiable pruning, the initialization time for new nodes is shortened. By taking into account the minimal transaction data required for ledger integrity checks and the essential validation information for transactions, nodes can be reconstructed efficiently. In real-world settings, the startup time for new nodes typically ranges from several tens of seconds.



$R_{10^{10}+10}$
$R_{10^{9}+9}$
$R_{10^{8}+8}$
$R_{10^{7}+7}$
$R_{10^{6}+6}$
$R_{100005}$
$R_{10004}$
$R_{1003}$
$R_{102}$
. . .
$R_{30}$ $R_{29}$ $R_{28}$ $R_{27}$ $R_{26}$ $R_{25}$ $R_{24}$ $R_{23}$ $R_{22}$ $R_{21}$
$R_{20}$ $R_{19}$ $R_{18}$ $R_{17}$ $R_{16}$ $R_{15}$ $R_{14}$ $R_{13}$ $R_{12}$ $R_{11}$
$R_{10}$ $R_{9}$ $R_{8}$ $R_{7}$ $R_{6}$ $R_{5}$ $R_{4}$ $R_{3}$ $R_{2}$ $R_{1}$
Exponential jump link distance

Figure 8: Hierarchical Skewed Merkle Tree and Jump Links exceeding exponential distances.

# 7. Inter-Shard Communication

## (1) The Importance of Inter-Shard Communication

Each shard operates independently without inherent knowledge of others. However, for tasks like delivering transaction details and account movement, inter-shard information exchange is essential, necessitating a method of communication between shards.

## (2) Inter-Shard Communication Among Nodes

Roughly half of the communication connections for each node consist of intra-shard communication, while the remaining half involves communication with nodes from other shards, referred to as inter-shard communication. In intra-shard communication, transactions and messages are directly transmitted without further consideration. The legitimacy of received messages is individually verified by each node.

Inter-shard communication mainly occurs when a transaction involves accounts from different shards. For instance, consider a transaction where account A intends to send coins to account B. If account A belongs to shard *i* and account B to shard *j*, the transaction initiated by A is issued in shard *i* but information also needs to be sent to shard *j* where B resides. In such cases, inter-shard communication facilitates the transmission of the transaction from shard *i* to shard *j*.

During communication with other shards, simple query messages are transmitted and executed directly without much consideration. However, in cases requiring clear verification, such as transactions, it is preferable to exchange necessary verification information with communication.

## (3) Nodes Engaged in Inter-Shard Communication

Every node within a shard actively participates in inter-shard communication. If a node receives a transaction through intra-shard communication that needs to be transmitted to another shard (let's call it shard j), it will relay the transaction to its neighboring nodes that are belonging to shard j. However, nodes do not establish new communication channels exclusively for inter-shard communication.

When a node receives a transaction from another shard, it transmits this transaction within its own shard using the standard intra-shard communication methods.

## (4) Strength of Verification Information

In transaction validation, there are multiple levels of verification information with varying degrees of strength. These can broadly be categorized into hard proof and soft proof verifications.

#### Hard Proof Verification

Hard proof verification comprises information that fully demonstrates the consensus status of a transaction based on the world state. For instance, if a transaction is included in a block generated through intra-shard consensus, it will be accompanied by a Merkle tree path derived from the WRS block.

Complete hard proof verification becomes available only after the confirmation of world consensus, typically requiring the time required for at least one or more round, from transaction creation. Although robust, it may lack immediacy.

#### Soft Proof Verification

When transmitting transactions in an unconfirmed state, not yet included in blocks, to other shards, it's possible to provide information linking to existing transactions verifiable from the WRS. For example, by transmitting information that verifies the immediately preceding transaction of a newly generated transaction, it can offer an indication that the transaction is not entirely fabricated.

Soft proof verification cannot offer full consensus validation for a transaction. However, among nodes possessing adequate information, such as consensus-executing nodes, it provides sufficient verification capability to be considered reliable information.

### (5) Request for Verification Information

If a shard receiving a transaction lacks adequate information to verify it, it can initiate a request to the sender for verification proof. For instance, if a transaction received from a third shard references another transaction not known to the receiving shard, verification information can be requested from the shard associated with the unknown transaction to enable verification.

Since the party requested for verification information can be any arbitrary node, nodes can probabilistically request verification information through a third party node and conduct cross-verification.

# 8. Smart Contracts

Locus Chain is a high-performance blockchain system designed to handle thousands of transactions per second. The high-speed transaction processing presents various technical challenges, and how Locus Chain overcomes these challenges has been discussed in the preceding chapters. Similarly, executing smart contracts at such transaction processing speeds brings about issues that were not previously apparent.

## (1) Smart Contract Execution Calculation Model

Present-day smart contracts essentially possess functionality equivalent to regular computer programs. Consequently, executing smart contracts necessitates additional hardware resources such as CPU and memory, much like running multiple regular computer programs concurrently. The resource requirements for smart contract execution scale proportionally with the number of smart contracts being processed. In Locus Chain, which targets performance several hundred times greater than existing blockchains, computation would demand several hundred times more CPU and memory than current smart contracts require accordingly to rough calculation. This surpasses the capabilities of a typical home PC and makes it unfeasible for each node in the existing smart contract model to execute all loaded smart contracts.

To tackle this challenge, Locus Chain employs a segmented approach to smart contract execution. Simply put, smart contract execution is partitioned into multiple groups and processed in parallel. Each group handles a sufficient number of smart contracts that can be executed without issues. By distributing different smart contracts among multiple groups for parallel execution, the model enables the overall execution of a large volume of smart contracts.

## (2) Smart Contract Accounts and Execution Groups

Each smart contract running on Locus Chain possesses a unique address, akin to standard Locus Chain account addresses.

The role of smart contract execution groups is to generate transactions initiated by smart contract account addresses. Initially, a regular account triggers smart contract execution by issuing an input transaction to the smart contract address. The smart contract execution group then processes the input transaction in the appropriate sequence, creating transactions for the respective smart contract address. The output of smart contract execution results is logged in the ledger of the associated AWTC chain address and undergoes sharding and pruning similar to ledger entries of regular accounts.

In essence, smart contract execution groups are clusters of node computers within Locus Chain responsible for executing and finalizing the results of the smart contracts under their management, recording them in the Locus Chain ledger.

### (3) Node Participation in Smart Contract Execution

Nodes have the discretion to decide whether they will engage in smart contract execution.

Locus Chain boasts an incidental advantage of functioning efficiently on low-performance, low-capacity IoT devices through pruning and sharding technology. However, smart contract execution demands additional CPU performance and memory, alongside ledger verification functionality. This may strain IoT device hardware, which is typically compactly designed for specific purposes and may lack spare resources for smart contract execution.

Therefore, in Locus Chain, smart contract execution is not mandated for all nodes. Instead, nodes with sufficient hardware performance capabilities can opt to participate in smart contract execution. Participating nodes receive additional incentives, distinct from Locus Chain transaction management, encouraging nodes with available resources to join in smart contract execution.

Nodes can independently choose the smart contract execution group in which they wish to participate. Each smart contract execution group may have varying hardware requirements. Nodes can select a group aligned with their performance capabilities.

Smart contract execution groups can be formed at any time by participants. To establish a new execution group, an account selects a supported smart contract language VM environment and pledges assets as initial capital for the group. Once created, other nodes can join or leave the group relatively freely.

### (4) Locus Chain VME: Smart Contract VM Execution Environment Service

Locus Chain operates a framework for executing a multitude of smart contracts by running multiple smart contract groups independently on a high-performance transaction processing base. To facilitate this framework, there exists a system interface that links the Locus Chain ledger with smart contract groups, known as the "Locus Chain Virtual Machine Environment (**VME**)."

The Locus Chain VME encompasses blockchain-level functionalities, such as supporting smart contract execution result transaction generation and consensus node selection, as well as network-level functionalities like packet communication support between nodes. It serves as a kind of "layer 1.5" system service. For instance, through the VME interface, seamless interaction via transaction issuance is facilitated between smart contract execution groups issuing standard transactions.

Each smart contract execution group within Locus Chain is an entirely independent entity and primarily interacts only through transactions on the Locus Chain ledger. Hence, each group can possess different functionalities and characteristics. For example, each group can execute different smart contract languages. Groups are considered as virtual machines (**VMs**) for smart contracts, and each VM can run any language environment that satisfies the Locus Chain VME interface. Execution information of VMs is shared only among nodes participating in the respective group, allowing nodes to flexibly participate in desired smart contract execution groups by plugging in VMs suitable for their execution capabilities and purposes.

Currently, Locus Chain primarily employs an EVM engine compatible with Ethereum at the language level and is considering additional implementations of various engines with different characteristics, such as the Move language VM and WASM-based VM engine. Since the VME engine supports ledger and network-level services universally, it can also serve as an experimental basis for third parties to develop new language VMs.

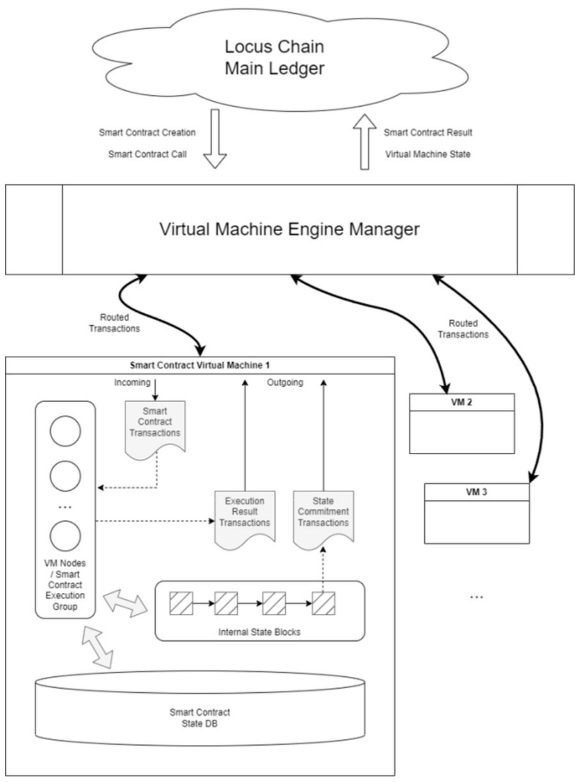


*Figure 9: Locus Chain VME Structure*

## (5) External System Integration via VME's "Plug-in" Extension Structure

From an everyday user's standpoint, the smart contract execution mechanism might seem like a black box. What's crucial for users is simply that upon sending a request to a smart contract account address, the smart contract executes, and they receive the outcome. Consequently, the design of the smart contract engine itself can be adaptable as needed. Furthermore, by linking the smart contract VM with external data sources, it becomes possible to generate results that interact with the real world.

To integrate external information, it's essential that all nodes participating in the smart contract execution group have access to a shared external data source (an oracle) for reference. Each execution node should then be able to access the same external information through this oracle. Meeting these requirements isn't overly complex in today's web service APIs. Therefore, implementing, and operating smart contract programs directly connected to external API systems on Locus Chain is viable.

As an illustrative example, consider a smart contract agreement that autonomously determines transaction directions by integrating with reliable foreign exchange rate records. Alternatively, systems could be developed to exchange information between blockchains by referencing transactions issued by other publicly verifiable blockchain systems.

Furthermore, systems could interact with specific services, such as games, by leveraging APIs provided by these services. This would allow Locus Chain accounts to assess resources within the game service and interact accordingly. Conversely, new services could utilize existing smart contracts on Locus Chain by issuing transactions on the network.

### (6) Locus Chain Corescript

The VME structure of Locus Chain operates under the premise that only nodes with sufficient hardware resources to execute Turing-complete computer languages will selectively participate. However, running a simplified script language for basic transaction-level processing is feasible even on low-performance devices. A prime example is the stack-based script system used in Bitcoin.

Similarly, Locus Chain features a simple script system known as Corescript, enabling basic transaction-related processing. Corescript commands consist of bytecode scripts with limited length and restricted instructions, included within each transaction. All nodes are obligated to execute this Corescript during transaction processing.

While Corescript alone cannot execute advanced processing such as transaction issuance, it can handle simple condition evaluation tasks, such as determining the application conditions of issued transactions. For instance, commonly referred to tasks like atomic swaps can be implemented using Corescript alone.

# 9. Cryptographic Keys

## (1) Locus Chain's Cryptographic Key Hierarchy

Similar to many other blockchain platforms, each account of Locus Chain is related to a pair of public and secret key supplied by a user. Generally, users must securely manage their secret key.

In Locus Chain, there are three kinds of cryptographic keys related to an account. There is a master key, a normal key, and a validation key.

A validation key is utilized by each node for round consensuses and similar operations. A validation key is automatically generated from the normal key. The validation key is supposed to be re-generated frequently.

A normal key is a key for general-purpose operations such as transaction signing. A normal key should be delivered by an algorithm balanced between cryptographic strength and ease of use. Currently, Locus Chain generates normal keys with an elliptic function cryptography algorithm.

A master key is a key for generating a normal key. A master key should be generated with a strong, computing-intensive algorithm usually not suitable for real-time operation. The algorithms for master keys are not finalized but working on both post-quantum cryptography systems and elliptic-function cryptography systems.

If a normal key becomes inappropriate for some reason, the user can discard the key and register a new key by issuing a key-exchange transaction with a new normal key generated by using the user's master key. The new normal key becomes valid for the next round of the consensus process which settles the key-exchange transaction.

Furthermore, Locus Chain is working on an approach for adding and changing algorithms for normal keys. We hope this approach could be a future-proof facility for the post-quantum computing era.

## (2) Quantum-resistant Cryptographic Signatures

One of the biggest threats to blockchains is the emerging threat of quantum computers. There is a possibility that quantum computers may break many currently used signature algorithms.

Yet, there are also researches regarding Post-Quantum Cryptographies (**PQC**) that are secure against quantum computers. However, current PQC are infeasible for personal computers and mobile devices because they require immense amount of computation and data compared to traditional algorithms. Also, there are no matured standards for PQCs yet, and it is hard to tell the precise property of PQC algorithms in actual use due to the lack of scientific and technological establishments.

To overcome this situation, Locus Chain incorporates a layered key system composed of two cryptographic keys, a master key and a normal key, of different strengths. A user uses the normal key, which depends on a current-generation cryptography system, for signing transactions. If the user's normal key is compromised, then PQC-applied master key is used to re-generate and change the broken normal key.

The algorithm for master keys can consume significant computing resources for PQC computation since master keys are only used in exceptional situations. Also, Locus Chain is working on a built-in mechanism to add additional signature algorithms. In the future, Locus Chain will be able to exchange the normal key algorithm itself with a Post-Quantum algorithm once PQC algorithms become widely available. Moreover, Locus Chain may use a hybrid of PQC and current-generation cryptographies for a while since the property of PQC is not yet completely understood. Even when vulnerabilities of PQC signature algorithms are found, Locus Chain can make use of current-generation algorithms to quickly, temporarily cover the vulnerabilities.

# 10. Economic Structure (Rewards, Coins, Grants)

## (1) Rewards for Locus Chain Participants

The voluntary engagement of accounts and nodes within the Locus Chain system is paramount for its seamless operation. Similar to conventional blockchain ecosystems, Locus Chain incorporates the Coin and Grant concepts to incentivize active participation among nodes and accounts. While the system accommodates a broad spectrum of data types, Coins and Grants serve as specialized data units symbolizing value within the system and are integral to its optimal functionality.

## (2) Coins and Grants

In the Locus Chain ecosystem, Coins function as numerical representations of general value metrics. They are transferrable between accounts, with each account receiving an initial Coin allocation upon creation. Essentially, the first transaction constituting the first block for each account, except those generated during the genesis round, involves receiving Coins.

Grants, on the other hand, represent internal system values critical for Locus Chain's operational functions. For instance, transaction creation necessitates a Grant payment, with additional Grants necessary for transactions involving services like smart contracts. In cases of insufficient Grant, Coin can be consumed or "burned" as an alternative.

The quantities of Coins and Grants are publicly accessible data, allowing any participant to determine the amounts associated with each account based on transaction history.

## (3) Coin Stake

Coins play a vital role in determining stake amounts within the Locus Chain system. In the process of probabilistically electing the Committee members from the nodes for each round's consensus in each shard, nodes with higher stake amounts are more likely to be selected. This design, with a Delegated Proof of Stake (**DPoS**)[1] structure, ensures that nodes with relatively larger stakes contribute more significantly to operation and maintenance of the system.

Accounts within the Locus Chain have the option to delegate their stake to other accounts. This feature allows accounts that are not always connected to the network to effectively utilize their stakes.

[1] The Delegated Proof of Stake (dPoS) used in EOS and the Delegated Proof of Stake (DPoS) used in Locus Chain are not the same concept. In EOS, the number of nodes eligible to participate in consensus and receive rewards is limited to 21, meaning that even if a regular account holds coins, it cannot participate in consensus. Instead, it must delegate its stake to one of the pre-selected 21 nodes. However, Locus Chain fundamentally differs from EOS as it does not restrict node participation.

The stake of a node comprises the sum of the stake held by the node's host account and the stake delegated from other accounts.

### (4) Epoch: Incentive Calculation Unit

Coins serve as incentives (rewards) for nodes and accounts contributing to operating and maintaining the Locus Chain. These coin incentives are calculated based on a predetermined formula using the historical records of nodes' contributions to consensus rounds. Consequently, any node calculating the distribution of incentives would arrive at the same result.

Transactions are confirmed and finalized on a per-round basis. To streamline the process, incentive calculation aggregates rounds over a specific period in the past. Presently, the Locus Chain computes 'daily rewards' by consolidating rounds over a 24-hour span into an 'Epoch,' marking the round when the incentive calculation occurs. In other words, the distribution of incentives occurs on an epoch basis.

The round chosen as the reference for epoch calculation is termed the Epoch Pivot Round. In rounds requiring epoch calculation, each shard utilizes ledger information from the Epoch Pivot Round to compute crucial parameters such as the reference stake amount and the number of reference nodes used in the incentive calculation and consensus algorithm. Following the completion of these calculations, from the subsequent round onwards (Epoch Transition Round), all nodes within the shard apply the computed reference stake amount until it is updated in the next epoch.

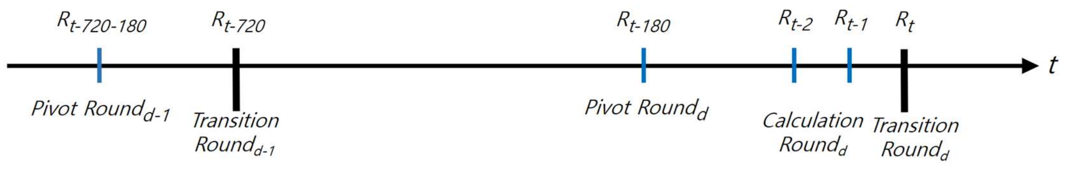


*Figure 10: Epoch Pivot Round, Epoch Calculation Round, and Epoch Transition Round*

The calculated values of incentives, etc., at the Epoch Pivot Round are recorded on the ledger alongside other transactions of that round through consensus execution. Subsequently, starting from the round immediately after the ledger entry (Epoch Transition Round), shards employ the newly computed information in the consensus algorithm for each round.

Each eligible account receives coins as incentives by issuing transactions for receiving rewards based on the results of epoch calculation.

Unlike coins, grants are automatically calculated and distributed at regular round intervals to accounts participating in consensus and those delegating authority to participate. However, there are respective upper limits for these two scenarios to prevent grants from accumulating beyond specified thresholds.

### (5) Coins as Incentives

Coins allocated as incentives for mining are distributed across accounts in proportion to their respective contributions to the Locus Chain, on an epoch-by-epoch basis.

To encourage the activation of the Locus Chain platform in the early stages, nodes that participate initially are granted relatively more coins to acknowledge their higher contributions. As epochs progress, the incentives paid in Locus Coins from the pre-issued reserves are designed to gradually decrease.

The magnitude of epoch-level incentives is impacted by fluctuations in the reserve of Locus Coins, which are replenished through fees paid in coins.

In order to mitigate any potential disadvantages in stake allocation for new participants relative to long-term participants actively engaged in the Locus Chain consensus process, the correlation between stake amount and coin holdings is intentionally modified to deviate from strict proportionality.

### (6) Grants

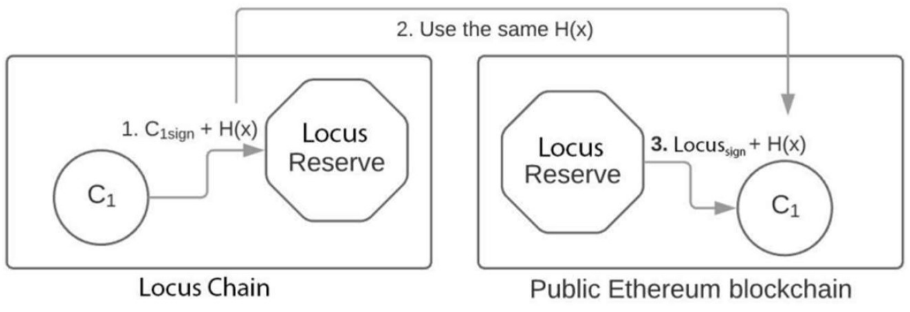
Grants serve as internal resource units necessary for transaction issuance. They cannot be transferred and have an upper limit, preventing accumulation beyond this threshold. There are two methods to acquire grants. The host account of a node participating in the consensus receives grants periodically based on its contribution to the consensus over specific intervals. The host account cannot exceed the periodic limit of grants. Guest accounts, which do not partake in the consensus and delegate stake to host accounts, receive a predetermined quantity of free grants periodically.

Grants facilitate small transactions without coin expenditure, enabling microtransactions. Furthermore, their limited allocation prevents excessive simultaneous transaction issuance. When grants are insufficient, transactions must be initiated by spending coins.

# 11. Tokenomics Design

## (1) Token

The Locus Tokens currently in circulation are ERC20 tokens based on Ethereum, issued by Locus Chain Foundation headquartered in Singapore. However, direct utilization of existing Ethereum network-based tokens on Locus Chain is not feasible. Plans are in place to perform an Atomic Swap of the existing tokens to Locus Chain-based coins at a 1:1 ratio when the Locus Chain core engine is fully deployed. As the Locus Chain core engine initiates comprehensive business support, a transition to a new coin system will take place to secure the liquidity necessary for expanding the Locus Chain platform ecosystem. Locus Chain Foundation will actively support the transition of Locus Tokens to new Locus Coins with a designated grace period for the completion of the swap.



*<Figure 11> Atomic Swap (Locus Chain <> Ethereum Mainnet)*

## (2) Token Economics

The design of Locus Chain's token distribution and economic incentive structure places the primary focus on enhancing the efficiency of mechanisms that incentivize sustainability of the Locus Chain ecosystem's value and encourage the voluntary participation of platform components. In designing these mechanisms, three core concepts served as the foundation:

- Stability of Value and Security
- Validity of Incentive-Based Contributions to Ecosystem Establishment and Growth
- Decentralization through Distributed Ownership and Decision-Making Structures

Locus Chain has adopted a Delegated Proof of Stake (DPoS) mechanism based on the Byzantine Fault Tolerance (BFT) consensus system, which is highly efficient in meeting these three criteria.

A notable aspect of Locus Chain's DPoS system is that voter selection is autonomously determined among participating nodes within the network, rather than pre-decided at the network's inception. This autonomy extends to individuals able to become nodes by downloading the ledger and fulfilling voter participation criteria, thereby qualifying as voters with a specific stake level in the network following current design standards.

This model allows participation in node activities even on low-spec desktop PCs connected to the internet in the background. Participants can become nodes mining Locus Coins as long as they meet stake requirements and internet connectivity criteria, regardless of hardware specifications or hashing power.

When an account delegates its stake, it is classified as a "Guest Account." While Guest Accounts do not mine coins or maintain ledgers, they can trade through nodes to which they delegate their stake. This model provides a system participation method for coin holders who do not wish to participate as nodes.

Locus Chain's DPoS features low barriers to mining Locus Coins, making it accessible for all without participation inequalities based on equipment performance. The system only allows programming rules that prevent errors or Byzantine failures (malicious hacking attempts), maintaining a high degree of decentralization and disallowing other forms of artificial interference in the system.

The general advantages of Locus Chain's incentive mechanism include rapid processing speeds and avoiding unnecessary energy consumption in ledger creation. Moreover, it is highly effective in acquiring resources for ensuring stable and continuous growth of the platform, a challenge faced by new blockchain platform operators during the early stages of operation. This mechanism enables the introduction of reasonable mechanisms to consider compensation for risks and opportunity costs associated with participation from the platform design phase to early activation stages. It also facilitates rewarding participants based on their varied contributions. Rather than solely focusing on the monetary value at stake, it sends a message of appreciation towards those who endured risks associated with building a new platform and fairly rewarded early participants who enabled the development and growth of our platform.

### (3) Token Allocation

A total of 7 billion Locus Tokens have been pre-mined, and the entire token supply will be distributed through Private Sales rather than public participation.

All foundational assets are owned by Locus Chain Foundation and are allocated solely for business benefits. Tokens for advisors and partner companies will be mostly locked up for 6 to 24 months.

Aside from the 7 billion Locus Tokens, there will be no additional issuance beyond the mining rewards following the launch of the mainnet. The transition of Locus Tokens to Locus Coins will be scheduled to undergo a process and will be fully swapped to Locus Coins through Atomic Swap.

| Distribution | Token Quantity | Percentage | Explanation |
|---|---|---|---|
| Foundation Reserved | 1,200,000,000 | 17% | Tokens held by Locus Chain to leverage better strategic benefits |
| Founders & Team | 600,000,000 | 9% | 2 year lock-up |
| Advisors & Partners | 1,200,000,000 | 17% | 6 ~ 24 month lock-up |
| Contributor | 4,000,000,000 | 57% | |
| TOTAL | 7,000,000,000 | 100% | |

*Figure 12. Token Distribution Quantity*

The mining of Locus Coins will commence alongside the official operation of the core engine-based platform, and the maximum amount of minable coins is set at 5 billion. Coins paid as mining fees are stored in the Locus Coin reserve. According to current mining mechanisms, an additional 5 billion Locus Coins will be mined over 50 years. This caps the total supply of Locus Coins at 12 billion.

Notably, transaction fees in Locus Chain are reclaimed to the Locus Coin reserve for mining rewards. The reclaimed coins will affect the coin reserve from which future mining rewards are determined. While the initial mining period is set at 50 years, as transaction fees are reclaimed to the Locus Coin reserve, leading to an increase in coin holdings in the reserve proportional to the continued increase in mineable duration. Thus, the mining period theoretically has the potential to extend indefinitely with a capped total supply of 12 billion coins.

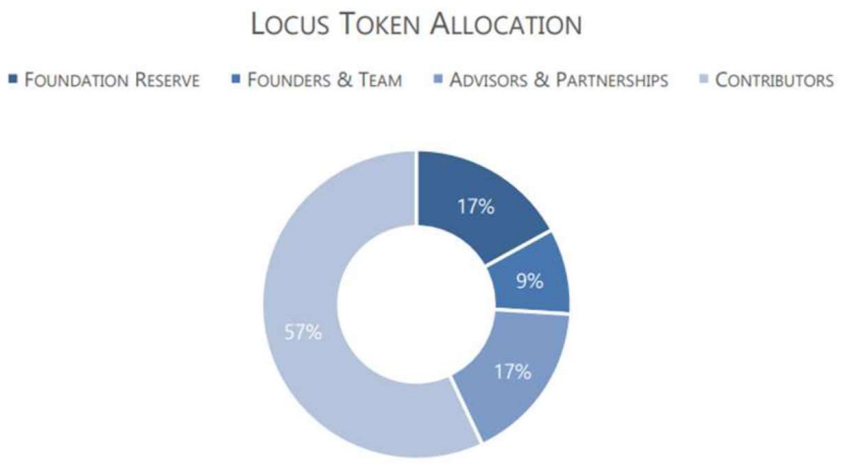


*Figure 13. Locus Token Allocation*

### (4) Utilization of Token Sale Proceeds

The proceeds from the token sales to Contributors will be allocated towards the planning, development, marketing, and promotion of Locus Chain, as well as the cultivation of its ecosystem. Since blockchain technology development is essential for project success, 50% of the funds will be dedicated to aspects related to blockchain technology development. The remaining 20% will support the Locus Chain ecosystem, aid various ecosystem participants, and cover ecosystem development costs. Additionally, 15% will be directed towards Locus Chain's marketing and promotional activities, while 10% will cover project operational expenses. The final 5% will be set aside as contingency reserves.

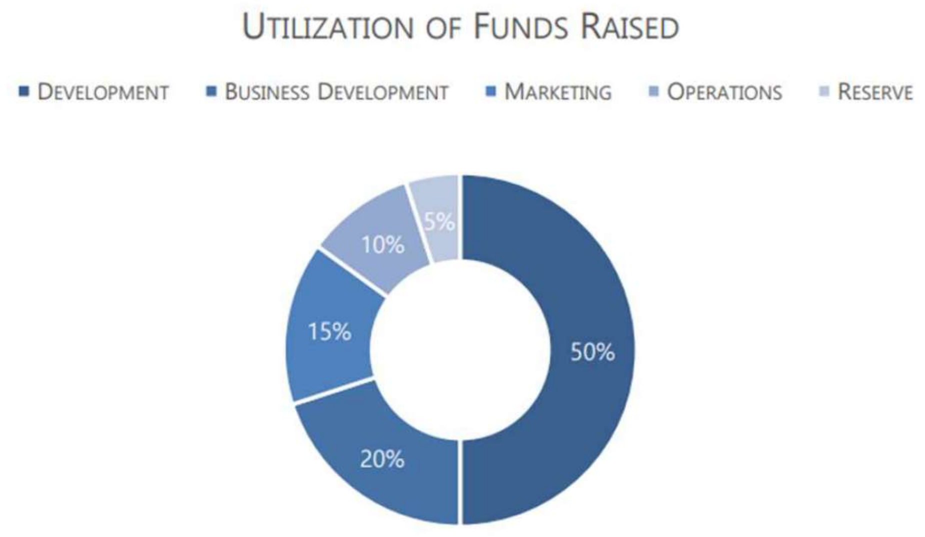


*Figure 14. Utilization of Funds Raised*

### (5) Governance

Locus Chain has devised the following governance operational plan. According to policies outlined in the DPoS process, voting rights are granted based on the quantity of coins held by each node. Delegating holdings from a guest account to a host account also delegates the voting rights. Detailed procedures concerning policy proposals, modifications, and voting protocols will be facilitated through the foundation's official wallet. More comprehensive information on initiatives, constraints, voting processes, execution, and penalties will be disclosed in upcoming announcements.

# 12. Applications of Locus Chain Technology

Locus Chain technology enables the swift processing of large transaction volumes on PCs and IoT devices, facilitated by innovations like dynamic sharding and verifiable pruning. This advancement has opened avenues for Locus Chain to address the limitations previously associated with existing blockchains, particularly in real-time information exchange and storage systems.

Locus Chain has transcended theoretical realms and is now actively integrated into various applications. As of the publication of the whitepaper, it powers products like "Locus Game Chain," a serverless online gaming platform, and "Locus Web Meeting," a serverless remote conferencing system.

Furthermore, ongoing development is underway for other online game products, metaverse worlds, real-time trading platforms, and similar purposes.

Below, we will examine the currently publicly available applications of Locus Chain technology.

## A. Locus Game Chain

### (1) Limitations of Traditional Game Server Systems

#### Constraints of Centralized Server-based Games

The typical setup for network game services involves **game client programs** running on devices like PCs or mobile phones, which connect to **centralized server systems** managed by game service companies. Players' game client programs establish connections to these centralized server systems via the internet, where communication occurs, and player game data is stored and processed.

Within such centralized server systems, all storage and processing of game information are concentrated. Consequently, as user numbers grow, there arises a need for high-performance central server systems to handle the load. Conversely, a decline in user numbers can lead to reduced revenue due to the high operating costs associated with maintaining centralized server systems, potentially resulting in premature service termination.

#### Challenges of Peer-to-Peer (P2P) Communication

As an alternative approach to running game services without centralized server systems, some utilize Peer-to-Peer (P2P) communication technology. A peer is a computer that is connected to the internet. Instead of connecting to a central server, computers running the game program connect directly to each other for communication and game progression.

However, P2P communication has clear limitations. While it reduces the communication burden on central servers and the workload for game progression to some extent, it lacks a central oversight system for monitoring misconduct. This absence of supervision can make it easier for malicious players to engage in misconduct, such as manipulating game information by connecting directly to each other. Similarly, due to the challenge of trusting the game save results of other players, there remains a need for a central server to verify and securely store game progression details.

### (2) Locus Game Chain

Locus Game Chain represents a decentralized gaming service system that transcends the constraints of conventional centralized server technology and P2P methods. It achieves this by leveraging the distributed ledger system inherent in Locus Chain.

Locus Game Chain is a distributed server application tailored for Windows, built upon Locus Chain technology. By interfacing with Locus Game Chain, game client programs gain access to services akin to those provided by traditional centralized game servers. These include functionalities such as data management resembling that of centralized servers, communication with fellow players, and exchange of information among nodes.

#### "Serverless" Network Game System

The distributed ledger system of Locus Chain enables the storage of verifiable and consistent information in a distributed computing environment. In essence, it ensures the secure storage of players' game progress information in a stable and auditable manner, even in the absence of a central server.

The capability to store information in a distributed environment (known as distributed database functionality) is a fundamental aspect of blockchain technology. Through the process of block creation, verified information becomes recognized as accurate, facilitating the sharing of precise data that can be verified by anyone within the distributed network. This characteristic theoretically allows almost all blockchain systems to operate as distributed databases. However, traditional blockchain systems encounter limitations, such as a transaction throughput of approximately 20 transactions per second, making them unsuitable for programs with a large number of concurrent users, such as gaming applications.

Locus Game Chain addresses this challenge by leveraging Locus Chain's **dynamic sharding** technology, enabling large-scale information processing and storage. Players participating in the game become nodes within the Locus Game Chain network. As the player count increases, so does the number of nodes in the Locus Game Chain network. With dynamic sharding technology applied to the expanding node count, the number of shards in Locus Chain automatically increases. Locus Chain's transaction processing capacity scales proportionally with the number of shards, thereby accommodating the increased volume of stored information. Consequently, in Locus Chain with dynamic sharding technology, as the player count rises, the information processing capacity naturally expands in tandem.

Furthermore, Locus Chain's **verifiable pruning technology** ensures that players only manage the essential minimum of game information. The game client held by a player temporarily retains data solely from surrounding clients engaged in real-time gameplay, discarding any redundant information through pruning. Even if a player's opponent changes during the game, the necessary game data is automatically exchanged between the new opponents to facilitate uninterrupted gameplay. With verifiable pruning technology, the received information's legitimacy from opponents can be verified, effectively deterring misconduct.

In essence, Locus Game Chain is a program that leverages Locus Chain technology to implement server-less network games without relying on a central server.

Presently, Locust Game Chain has been integrated into the real-time strategy simulation game "Kingdom Under Fire" and is available to the public on the renowned gaming platform "Steam."

## B. Locus Web Meeting

### (1) Locus Web Meeting: A Web Conferencing System without a Central Server

The information storage and exchange capabilities provided by Locus Game Chain extend beyond gaming to other network applications. Locus Web Meeting stands out as a prime example, leveraging Locus Chain's information exchange abilities within a web application.

Functioning entirely within a web browser, Locus Web Meeting offers a video conferencing solution.

Traditional video conferencing systems typically involve creating virtual meeting rooms, with users joining via URLs or similar means. In Locus Web Meeting, Locus Chain technology is applied to manage both virtual meeting room information and user data. Details of created virtual meeting rooms are securely recorded on the blockchain, enabling participants to directly verify the room's status and seamlessly join the meeting.

### (2) Security Advantages of Locus Web Meeting

Locus Web Meeting offers several security advantages for recorded meeting information.

Utilizing Locus Chain technology, Locus Web Meeting conducts video conferencing through direct communication, eliminating the need for a central server. Communication records are exchanged solely between participants' computers, with no data stored elsewhere. This approach mitigates risks associated with eavesdropping and data breaches at the program level.

Information concerning virtual meeting rooms is recorded on the blockchain in Locus Web Meeting. However, due to Locus Chain's verifiable pruning, blockchain-recorded information leaves minimal traces on the internet. Verifiable pruning ensures that Locus Chain nodes only retain necessary information, reducing the chance of meeting data persisting on nodes not directly involved in the meeting.

Meeting room information can be encrypted to allow access only to participants, rendering it inaccessible to third parties. Participants are identified solely by their Locus Chain account addresses, with no details such as nicknames stored. Additionally, meta-information like IP addresses is not stored on the blockchain. Verifiable pruning also allows participants to objectively validate details such as meeting creation time and participant involvement.

As a secondary advantage, Locus Web Meeting facilitates high-quality video and audio communication among participants without the need for server communication overhead, ensuring smooth and high-resolution meetings.

By directly applying the security and safety of distributed ledger systems to web conferencing soluti ons, Locus Web Meeting demonstrates the scalability of Locus Chain. As of the whitepaper publication date, the Locus Web Meeting is currently being prepared for trial service by the development team.

# 13. Closing Remarks

Locus Chain is an innovative blockchain platform to resolve performance and scalability issues without sacrificing the value of decentralization. Locus Chain is a high-speed, high-capacity blockchain system for diverging users on many kinds of hardware devices.

In this document, we described technical aspects that support the exceptional performance of the Locus Chain system.

Locus Chain adopted a nonlinear DAG structure to introduce parallelism and flexibility to the ledger. Locus Chain's Account Wise Transaction Chain (AWTC) structure is a parallel DAG data structure that describes relations between transactions. AWTC's multiple endpoints enable simultaneous high-speed referencing and processing of transactions, essential for Locus Chain's ultimate performance. AWTC's parallel nature makes ledger sharding a reality.

Locus Chain's consensus algorithm combines a Proof-of-Stake (PoS) based Byzantine Fault Tolerance (BFT) consensus with transaction-level competitive consensus, which is invoked only when necessary. This approach aims to address the inefficiencies and uncertainties associated with Nakamoto consensus, which relies on Proof-of-Work (PoW) and consumes significant CPU resources. To enhance the speed of BFT consensus, Locus Chain employs a fair probabilistic sampling method to select nodes participating in consensus from the entire network. This selection process considers various contributions of each node to the Locus Chain network, with coin ownership being a primary metric through PoS, supplemented by factors such as node uptime, ensuring a comprehensive consensus selection process.

Locus Chain resolves several resource issues using Dynamic Sharding. Sharding reduces each node's data processing load by dividing ledgers. Sharding also reduces network communication load by localizing communications. Each shard independently executes the BFT consensus algorithm, and the total transaction throughput is multiplied by the number of shards. Locus Chain ensures fairness between shards by dynamically adjusting the number of nodes of shards in semi-realtime.

Verifiable Pruning is the key technology to the storage space problem. Verifiable Pruning drastically reduces each node's storage space requirement while keeping ledger integrity. Locus Chain could fully operate on devices with limited storage capacity like IoT devices.

## Annex 1. Locus Chain's Domestic Patent Application and Registration Status

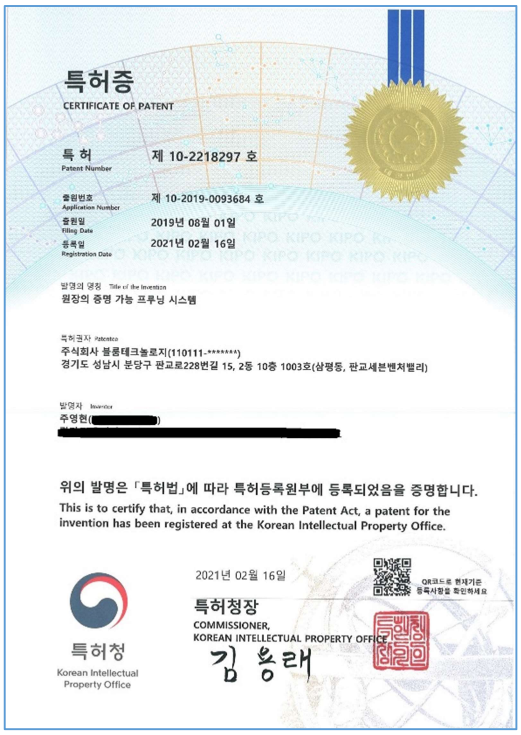
**As of the whitepaper update on May 6, 2024, Locus Chain has submitted a total of 10 patent applications domestically, with 4 patents registered and 6 patents under review.**

**A. Registered (4) and International Patent Applications (2)**

1. DAG-AWTC Ledger System with BFT Deterministic Consensus Method (August 1, 2019)
2. Verifiable Pruning System for Ledgers (August 1, 2019)
3. Inter-shard Transaction System and Method in Blockchain Networks (June 3, 2022)
4. System and Method for Changing a Working Shard of an Account in a Blockchain Network (June 14, 2022)

**B. Under Review (6)**

1. Verifiable Inter-shard Transaction System and Method in Blockchain Networks (July 12, 2022)
2. Dynamic Sharding System and Method in Blockchain Networks (August 17, 2022)
3. Blockchain Consensus System and Method (September 13, 2022)
4. System and Method for Changing Account Directory in Blockchain Networks (September 22, 2022)
5. System and Method for Account Creation in Blockchain Networks (October 25, 2022)
6. Node Distribution System and Method in Blockchain Networks (November 3, 2022)



특허증
CERTIFICATE OF PATENT

| | |
|---|---|
| 특 허 Patent Number | 제 10-2218297 호 |
| 출원번호 Application Number | 제 10-2019-0093684 호 |
| 출원일 Filing Date | 2019년 08월 01일 |
| 등록일 Registration Date | 2021년 02월 16일 |

발명의 명칭 Title of the Invention
원장의 증명 가능 프루닝 시스템

특허권자 Patentee
주식회사 블룸테크놀로지(110111-*******)
경기도 성남시 분당구 판교로228번길 15, 2동 10층 1003호(삼평동, 판교세븐벤처밸리)

발명자 Inventor
주영현([redacted])
[redacted]

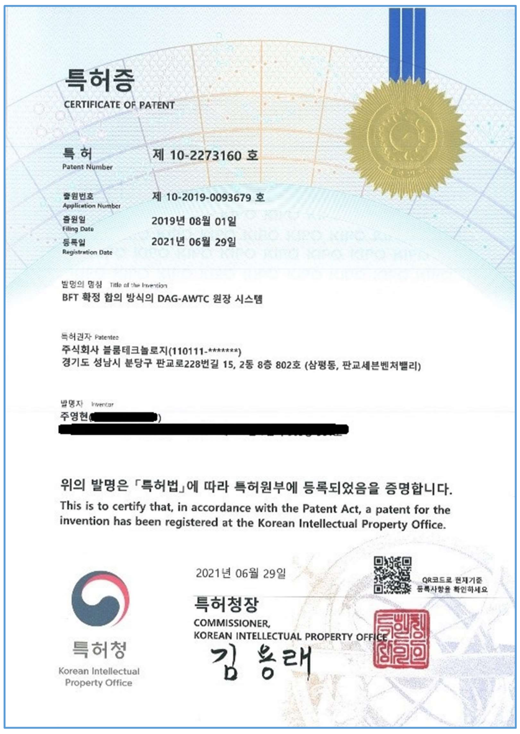
위의 발명은 「특허법」에 따라 특허등록원부에 등록되었음을 증명합니다.
This is to certify that, in accordance with the Patent Act, a patent for the invention has been registered at the Korean Intellectual Property Office.

2021년 02월 16일

QR코드로 현재기준 등록사항을 확인하세요

특허청장
COMMISSIONER,
KOREAN INTELLECTUAL PROPERTY OFFICE
김 용래

특허청
Korean Intellectual Property Office



특허증
CERTIFICATE OF PATENT

| | |
|---|---|
| 특 허 Patent Number | 제 10-2273160 호 |
| 출원번호 Application Number | 제 10-2019-0093679 호 |
| 출원일 Filing Date | 2019년 08월 01일 |
| 등록일 Registration Date | 2021년 06월 29일 |

발명의 명칭 Title of the Invention
BFT 확정 합의 방식의 DAG-AWTC 원장 시스템

특허권자 Patentee
주식회사 블룸테크놀로지(110111-*******)
경기도 성남시 분당구 판교로228번길 15, 2동 8층 802호 (삼평동, 판교세븐벤처밸리)

발명자 Inventor
주영현([redacted])
[redacted]

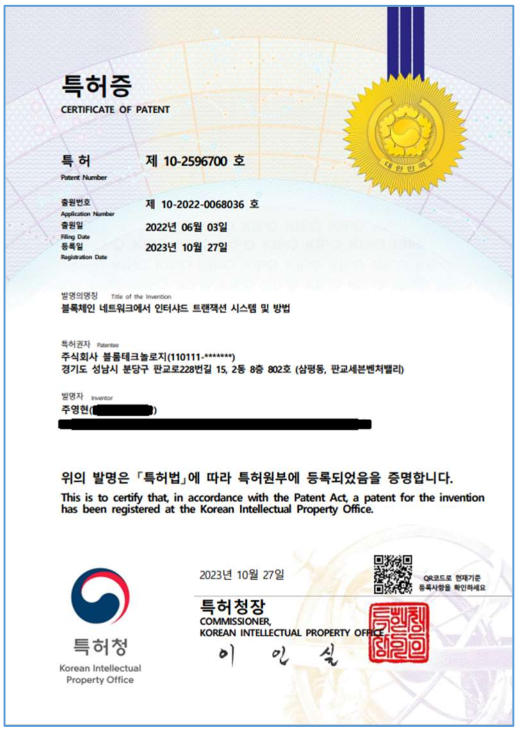
위의 발명은 「특허법」에 따라 특허원부에 등록되었음을 증명합니다.
This is to certify that, in accordance with the Patent Act, a patent for the invention has been registered at the Korean Intellectual Property Office.

2021년 06월 29일

QR코드로 현재기준 등록사항을 확인하세요

특허청장
COMMISSIONER,
KOREAN INTELLECTUAL PROPERTY OFFICE
김 용래

특허청
Korean Intellectual Property Office



특허증
CERTIFICATE OF PATENT

| | |
|---|---|
| 특 허 Patent Number | 제 10-2596700 호 |
| 출원번호 Application Number | 제 10-2022-0068036 호 |
| 출원일 Filing Date | 2022년 06월 03일 |
| 등록일 Registration Date | 2023년 10월 27일 |

발명의명칭 Title of the Invention
블록체인 네트워크에서 인터샤드 트랜잭션 시스템 및 방법

특허권자 Patentee
주식회사 블룸테크놀로지(110111-*******)
경기도 성남시 분당구 판교로228번길 15, 2동 8층 802호 (삼평동, 판교세븐벤처밸리)

발명자 Inventor
주영현([redacted])
[redacted]

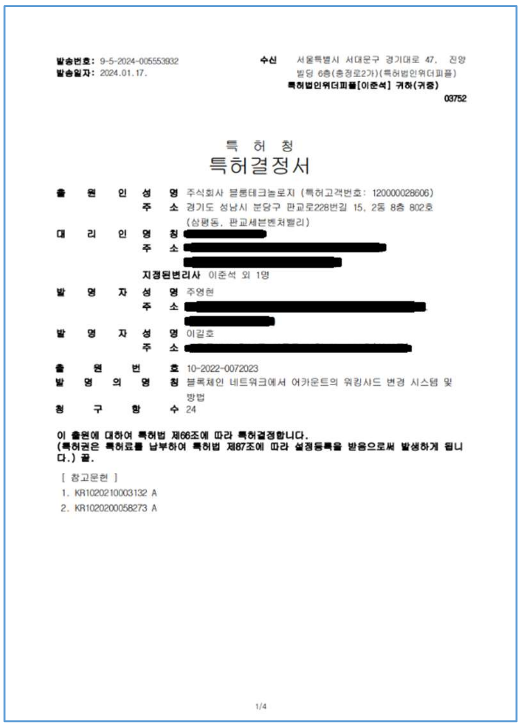
위의 발명은 「특허법」에 따라 특허원부에 등록되었음을 증명합니다.
This is to certify that, in accordance with the Patent Act, a patent for the invention has been registered at the Korean Intellectual Property Office.

2023년 10월 27일

QR코드로 현재기준 등록사항을 확인하세요

특허청장
COMMISSIONER,
KOREAN INTELLECTUAL PROPERTY OFFICE
이 인 실

특허청
Korean Intellectual Property Office



발송번호: 9-5-2024-005553932
발송일자: 2024.01.17.

수신 서울특별시 서대문구 경기대로 47, 진양빌딩 6층(충정로2가)(특허법인위더피플)
특허법인위더피플[이준석] 귀하(귀중)
03752

특 허 청
특허결정서

| | | |
|---|---|---|
| 출원인 | 성명 | 주식회사 블룸테크놀로지 (특허고객번호: 120000028606) |
| | 주소 | 경기도 성남시 분당구 판교로228번길 15, 2동 8층 802호 (삼평동, 판교세븐벤처밸리) |
| 대리인 | 명칭 | [redacted] |
| | 주소 | [redacted] |
| | 지정된변리사 | 이준석 외 1명 |
| 발명자 | 성명 | 주영현 |
| | 주소 | [redacted] |
| 발명자 | 성명 | 이길호 |
| | 주소 | [redacted] |
| 출원번호 | | 10-2022-0072023 |
| 발명의 명칭 | | 블록체인 네트워크에서 어카운트의 워킹샤드 변경 시스템 및 방법 |
| 청구항수 | | 24 |

**이 출원에 대하여 특허법 제66조에 따라 특허결정합니다.**
**(특허권은 특허료를 납부하여 특허법 제87조에 따라 설정등록을 받음으로써 발생하게 됩니다.) 끝.**

[ 참고문헌 ]
1. KR1020210003132 A
2. KR1020200058273 A

1/4

# Annex 2: Locus Chain's Overseas Patent Application and Registration Status

**As of the whitepaper update on May 6, 2024, among the 2 patents applied for in 7 countries, 6 patents have been registered in 4 countries, and 8 patents are under review.**

**A. United States**

1. DAG-AWTC Ledger System with BFT Deterministic Consensus Method (Patent Entry Date: January 19, 2022)
2. Verifiable Pruning System for Ledgers (Patent Entry Date: January 14, 2022, Registration Date: April 2, 2024)

**B. Japan**

1. DAG-AWTC Ledger System with BFT Deterministic Consensus Method (Patent Entry: January 18, 2022, Registration: June 27, 2023)
2. Verifiable Pruning System for Ledgers (Patent Entry: January 18, 2022, Registration: June 2, 2023)

**C. Russia**

1. DAG-AWTC Ledger System with BFT Deterministic Consensus Method (Patent Entry: February 2, 2022, Registration: December 14, 2023)
2. Verifiable Pruning System for Ledgers (Patent Entry: February 2, 2022, Registration: February 15, 2023)

**D. Australia**

1. DAG-AWTC Ledger System with BFT Deterministic Consensus Method (Patent Entry: January 21, 2022)
2. Verifiable Pruning System for Ledgers (Patent Entry: January 21, 2022, Registration: December 14, 2023)

**E. Europe**

1. DAG-AWTC Ledger System with BFT Deterministic Consensus Method (Patent Entry: December 29, 2021)
2. Verifiable Pruning System for Ledgers (Patent Entry: December 29, 2021)

**F. Canada**

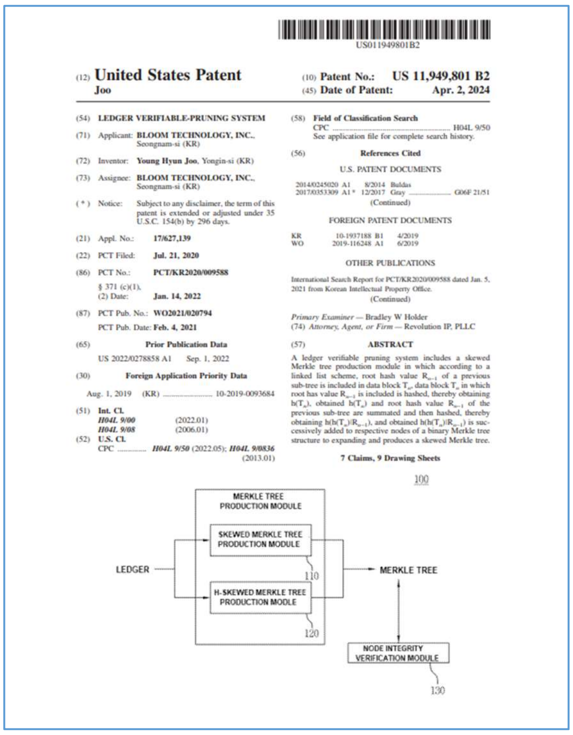
1. DAG-AWTC Ledger System with BFT Deterministic Consensus Method (Patent Entry: January 6, 2022)

2. Verifiable Pruning System for Ledgers (Patent Entry: January 5, 2022)

**G. China**

1. DAG-AWTC Ledger System with BFT Deterministic Consensus Method (Patent Entry: January 21, 2022)

2. Verifiable Pruning System for Ledgers (Patent Entry: January 21, 2022)



US011949801B2

(12) **United States Patent**
**Joo**

(10) **Patent No.: US 11,949,801 B2**
(45) **Date of Patent: Apr. 2, 2024**

(54) **LEDGER VERIFIABLE-PRUNING SYSTEM**

(71) Applicant: **BLOOM TECHNOLOGY, INC.**, Seongnam-si (KR)

(72) Inventor: **Young Hyun Joo**, Yongin-si (KR)

(73) Assignee: **BLOOM TECHNOLOGY, INC.**, Seongnam-si (KR)

( * ) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 296 days.

(21) Appl. No.: **17/627,139**

(22) PCT Filed: **Jul. 21, 2020**

(86) PCT No.: **PCT/KR2020/009588**
§ 371 (c)(1),
(2) Date: **Jan. 14, 2022**

(87) PCT Pub. No.: **WO2021/020794**
PCT Pub. Date: **Feb. 4, 2021**

(65) **Prior Publication Data**
US 2022/0278858 A1 Sep. 1, 2022

(30) **Foreign Application Priority Data**
Aug. 1, 2019 (KR) ........................ 10-2019-0093684

(51) **Int. Cl.**
***H04L 9/00*** (2022.01)
***H04L 9/08*** (2006.01)

(52) **U.S. Cl.**
CPC .............. ***H04L 9/50*** (2022.05); ***H04L 9/0836*** (2013.01)

(58) **Field of Classification Search**
CPC ........................................................ H04L 9/50
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

2014/0245020 A1 8/2014 Buldas
2017/0353309 A1 * 12/2017 Gray .................... G06F 21/51
(Continued)

FOREIGN PATENT DOCUMENTS

KR 10-1937188 B1 4/2019
WO 2019-116248 A1 6/2019

OTHER PUBLICATIONS

International Search Report for PCT/KR2020/009588 dated Jan. 5, 2021 from Korean Intellectual Property Office.
(Continued)

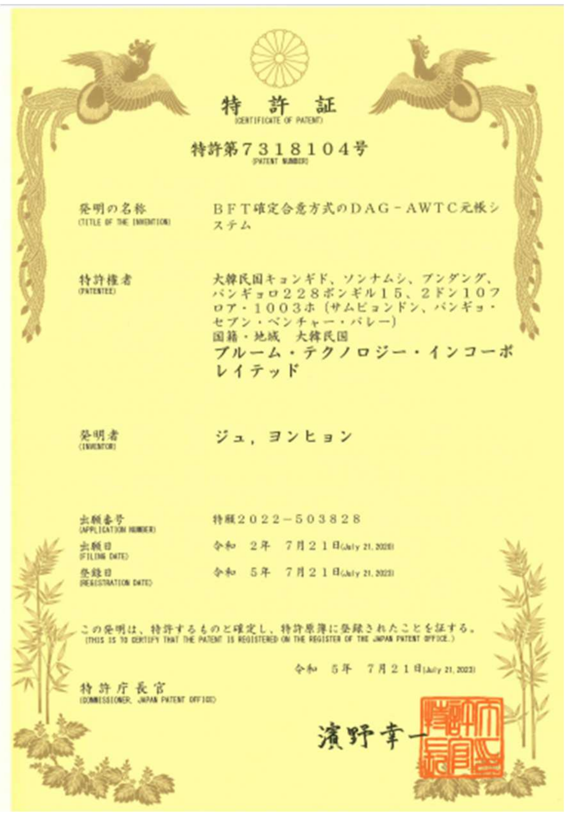
*Primary Examiner* — Bradley W Holder
(74) *Attorney, Agent, or Firm* — Revolution IP, PLLC

(57) **ABSTRACT**

A ledger verifiable pruning system includes a skewed Merkle tree production module in which according to a linked list scheme, root hash value $R_{n-1}$ of a previous sub-tree is included in data block $T_n$, data block $T_n$ in which root has value $R_{n-1}$ is included is hashed, thereby obtaining $h(T_n)$, obtained $h(T_n)$ and root hash value $R_{n-1}$ of the previous sub-tree are summated and then hashed, thereby obtaining $h(h(T_n)|R_{n-1})$, and obtained $h(h(T_n)|R_{n-1})$ is successively added to respective nodes of a binary Merkle tree structure to expanding and produces a skewed Merkle tree.

**7 Claims, 9 Drawing Sheets**

100
MERKLE TREE PRODUCTION MODULE
SKEWED MERKLE TREE PRODUCTION MODULE
110
LEDGER
H-SKEWED MERKLE TREE PRODUCTION MODLE
120
MERKLE TREE
NODE INTEGRITY VERIFICATION MODULE
130



特　許　証
(CERTIFICATE OF PATENT)

特許第７３１８１０４号
(PATENT NUMBER)

発明の名称 (TITLE OF THE INVENTION)　ＢＦＴ確定合意方式のＤＡＧ－ＡＷＴＣ元帳システム

特許権者 (PATENTEE)　大韓民国キョンギド、ソンナムシ、ブンダング、パンギョロ２２８ボンギル１５、２ドン１０フロア・１００３ホ（サムピョンドン、パンギョ・セブン・ベンチャー・バレー）
国籍・地域　大韓民国
**ブルーム・テクノロジー・インコーポレイテッド**

発明者 (INVENTOR)　ジュ，ヨンヒョン

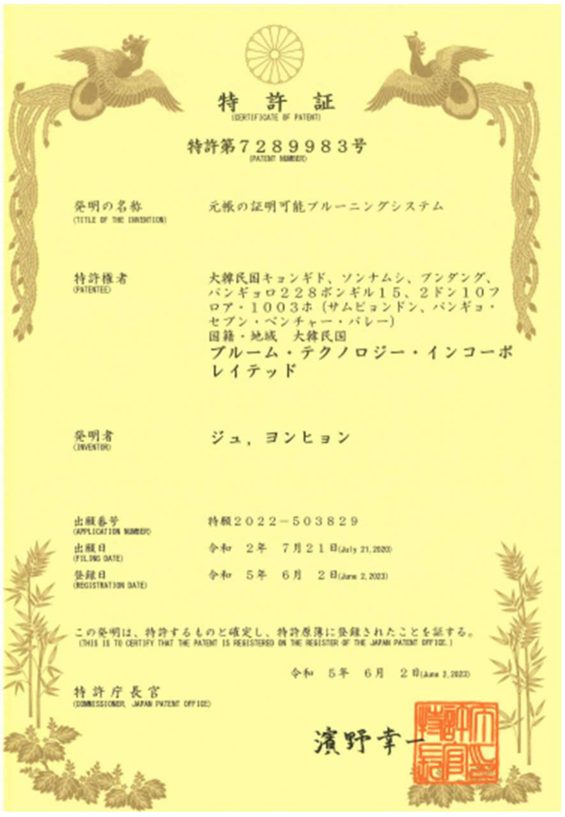
出願番号 (APPLICATION NUMBER)　特願２０２２－５０３８２８
出願日 (FILING DATE)　令和　２年　７月２１日(July 21, 2020)
登録日 (REGISTRATION DATE)　令和　５年　７月２１日(July 21, 2023)

この発明は、特許するものと確定し、特許原簿に登録されたことを証する。
(THIS IS TO CERTIFY THAT THE PATENT IS REGISTERED ON THE REGISTER OF THE JAPAN PATENT OFFICE.)

令和　５年　７月２１日(July 21, 2023)

特許庁長官
(COMMISSIONER, JAPAN PATENT OFFICE)

濱野幸一



特　許　証
(CERTIFICATE OF PATENT)

特許第７２８９９８３号
(PATENT NUMBER)

発明の名称 (TITLE OF THE INVENTION)　元帳の証明可能プルーニングシステム

特許権者 (PATENTEE)　大韓民国キョンギド、ソンナムシ、ブンダング、パンギョロ２２８ボンギル１５、２ドン１０フロア・１００３ホ（サムピョンドン、パンギョ・セブン・ベンチャー・バレー）
国籍・地域　大韓民国
**ブルーム・テクノロジー・インコーポレイテッド**

発明者 (INVENTOR)　ジュ，ヨンヒョン

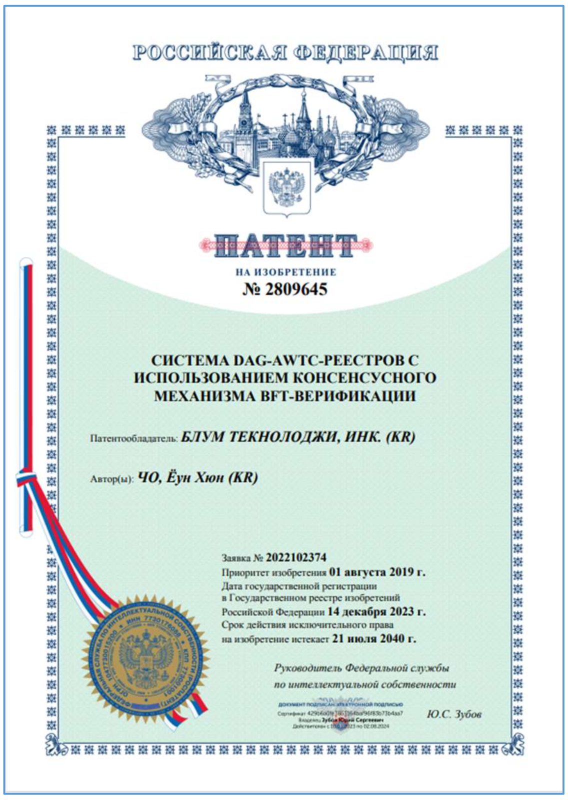
出願番号 (APPLICATION NUMBER)　特願２０２２－５０３８２９
出願日 (FILING DATE)　令和　２年　７月２１日(July 21, 2020)
登録日 (REGISTRATION DATE)　令和　５年　６月　２日(June 2, 2023)

この発明は、特許するものと確定し、特許原簿に登録されたことを証する。
(THIS IS TO CERTIFY THAT THE PATENT IS REGISTERED ON THE REGISTER OF THE JAPAN PATENT OFFICE.)

令和　５年　６月　２日(June 2, 2023)

特許庁長官
(COMMISSIONER, JAPAN PATENT OFFICE)

濱野幸一


РОССИЙСКАЯ ФЕДЕРАЦИЯ

ПАТЕНТ

НА ИЗОБРЕТЕНИЕ

№ 2809645

**СИСТЕМА DAG-AWTC-РЕЕСТРОВ С ИСПОЛЬЗОВАНИЕМ КОНСЕНСУСНОГО МЕХАНИЗМА BFT-ВЕРИФИКАЦИИ**

Патентообладатель: ***БЛУМ ТЕКНОЛОДЖИ, ИНК. (KR)***

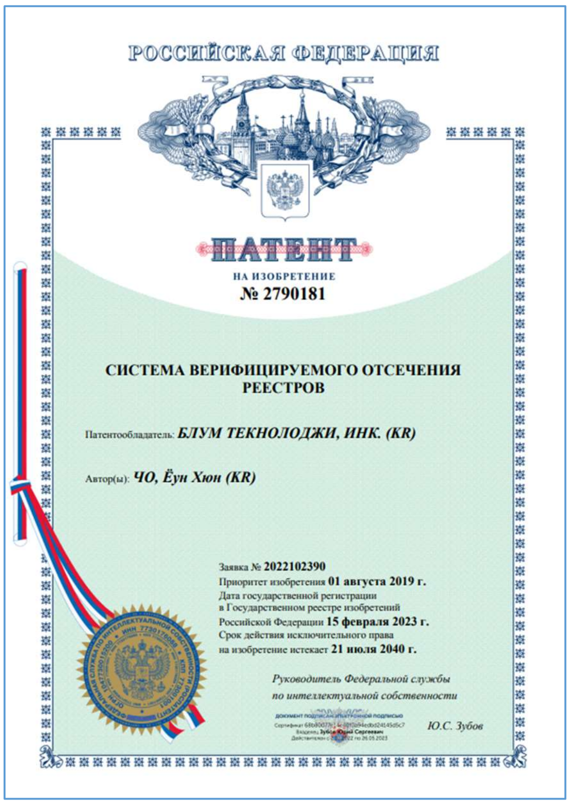
Автор(ы): ***ЧО, Ёун Хюн (KR)***

Заявка № **2022102374**
Приоритет изобретения **01 августа 2019 г.**
Дата государственной регистрации
в Государственном реестре изобретений
Российской Федерации **14 декабря 2023 г.**
Срок действия исключительного права
на изобретение истекает **21 июля 2040 г.**

*Руководитель Федеральной службы*
*по интеллектуальной собственности*

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
Сертификат 429bfa0[illegible]b4ba96f83b73b4aa7
Владелец **Зубов Юрий Сергеевич**
Действителен с [illegible] 2023 по 02.08.2024

*Ю.С. Зубов*


РОССИЙСКАЯ ФЕДЕРАЦИЯ

ПАТЕНТ

НА ИЗОБРЕТЕНИЕ

№ 2790181

**СИСТЕМА ВЕРИФИЦИРУЕМОГО ОТСЕЧЕНИЯ РЕЕСТРОВ**

Патентообладатель: ***БЛУМ ТЕКНОЛОДЖИ, ИНК. (KR)***

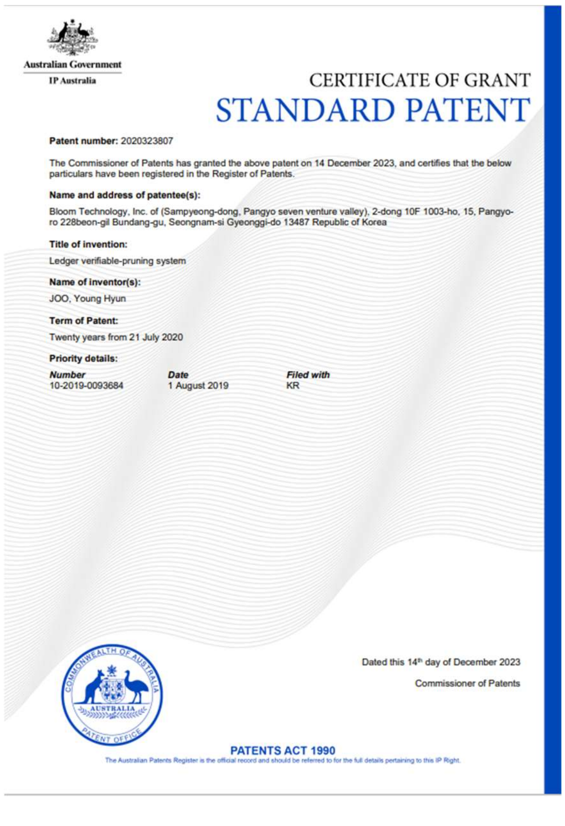
Автор(ы): ***ЧО, Ёун Хюн (KR)***

Заявка № **2022102390**
Приоритет изобретения **01 августа 2019 г.**
Дата государственной регистрации
в Государственном реестре изобретений
Российской Федерации **15 февраля 2023 г.**
Срок действия исключительного права
на изобретение истекает **21 июля 2040 г.**

*Руководитель Федеральной службы*
*по интеллектуальной собственности*

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
Сертификат 68b80077[illegible]a94edbd24145d5c7
Владелец **Зубов Юрий Сергеевич**
Действителен с [illegible] 2022 по 26.05.2023

*Ю.С. Зубов*


Australian Government
IP Australia

CERTIFICATE OF GRANT
STANDARD PATENT

**Patent number:** 2020323807

The Commissioner of Patents has granted the above patent on 14 December 2023, and certifies that the below particulars have been registered in the Register of Patents.

**Name and address of patentee(s):**
Bloom Technology, Inc. of (Sampyeong-dong, Pangyo seven venture valley), 2-dong 10F 1003-ho, 15, Pangyo-ro 228beon-gil Bundang-gu, Seongnam-si Gyeonggi-do 13487 Republic of Korea

**Title of invention:**
Ledger verifiable-pruning system

**Name of inventor(s):**
JOO, Young Hyun

**Term of Patent:**
Twenty years from 21 July 2020

**Priority details:**

| *Number* | *Date* | *Filed with* |
|---|---|---|
| 10-2019-0093684 | 1 August 2019 | KR |

Dated this 14th day of December 2023

Commissioner of Patents

COMMONWEALTH OF AUSTRALIA
AUSTRALIA
PATENT OFFICE

**PATENTS ACT 1990**
The Australian Patents Register is the official record and should be referred to for the full details pertaining to this IP Right.

# Bibliography

Association, T. L. (n.d.). *An Introduction to Libra*. Retrieved from https://libra.org/en-us/whitepaper

Bentov, I., Gabizon, A., & Mizrahi, A. (2016). *Cryptocurrencies without proof of work.* Retrieved from https://arxiv.org/abs/1406.5694

Bernstein, D. J. (2015). Multi-user Schnorr Security, Revisited. *Cryptology ePrint Archive; Vol. 2015/996.* IACR.

*Bitcoin: what a waste of resources*. (2017, 11). Retrieved from New Scientist: https://www.newscientist.com/article/mg23631503-300-bitcoin-what-a-waste-of-resources/

Buterin, V. (n.d.). *Ethereum Sharding FAQ*. Retrieved from Ethereum Wiki: https://github.com/ethereum/wiki/wiki/Sharding-FAQ

Buterin, V., & Gavin, W. (2013). *Ethereum*. Retrieved from https://www.ethereum.org/

Castro, M., & Liskov, B. (1999). Practical byzantine fault tolerance. *Proceedings of the Third Symposium on Operating Systems Design and Implementation (OSDI'99)*, (pp. 173-186).

Chen, L., Jordan, S., Liu, Y.-K., Moody, D., Peralta, R., Perlner, R., et al. (2016). *Report on Post-Quantum Cryptography.* National Institute of Standars and Technology, Internal Report 8105.

Dang, H., Dinh, T. T., Chang, D. L.-C., Lin, Q., & Ooi, B. C. (2019). Towards Scaling Blockchain Systems via Sharding. *2019 International Conference on Management of Data (SIGMOD '19).* ACM.

Demers, A., Greene, D., Hauser, C., Irish, W., Larson, J., Shenker, S., et al. (1987). Epidemic Algorithms for Replicated Database Maintenance. *Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing (PODC '87)* (pp. 1-12). New York: ACM.

*Directed Acyclic Graph*. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Directed_acyclic_graph

Gilad, Y., Hemo, R., Micali, S., Vlachos, G., & Zeldovich, N. (2018). *Algorand: Scaling Byzantine Agreements for Cryptocurrencies.* Retrieved from https://algorandcom.cdn.prismic.io/algorandcom%2Fa26acb80-b80c-46ff-a1ab-a8121f74f3a3_p51-gilad.pdf

Lamport, L., Shostak, R., & Pease, M. (1982). The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems, 4.3*, 382-401.

LeMahieu, C. (2017, 11). *Nano: A Feeless Distributed Cryptocurrency Network.* Retrieved from https://nano.org/en/whitepaper

Maymounkov, P., & Mazières, D. (2002). Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. *International Workshop on Peer-to-Peer Systems (IPTPS 2002)*, (pp. 53-65).

Merkle, R. C. (1987). A digital signature based on a conventional encryption function. *A Conference on the Theory and Applications of Cryptographic Techniques on Advances in Cryptology (CRYPTO' 87)*, (pp. 369-378).

*Merkle-Patrica Trie Specification*. (n.d.). Retrieved from Ethereum Wiki: https://github.com/ethereum/wiki/wiki/Patricia-Tree#main-specification-merkle-patricia-trie

Micali, S., Vadhan, S., & Rabin, M. (1999). Verifiable Random Functions. *IEEE 40th Annual Symposium on Foundations of Computer Science (FOCS'99)*, 120-130.

Mills, D., Wang, K., Malone, B., Ravi, A., Marquardt, J., Chen, C., et al. (2016). *Distributed ledger technology in payments, clearing, and settlement.* Divisions of Research & Statistics and Monetary Affairs, Federal Reserve Board.

Nair, G. R., & Shoney, S. (2017). BlockChain Technology: Centralised Ledger to Distributed Ledger. *International Research Journal of Engineering and Technology, Vol.4, Issue 3*.

Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. Retrieved from https://bitcoin.org/bitcoin.pdf

Popov, S. (2018, 4). *The Tangle.* Retrieved from https://www.iota.org/research/academic-papers: https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf

*Proof of Stake*. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Proof_of_stake

*Proof of Work*. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Proof_of_work

Schnorr, C. P. (1991). Efficient signature generation by smart cards. *Journal of Cryptology, Vol. 4, no.3*, 161-174.

*Shard (database architecture)*. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Shard_(database_architecture)

*Visa acceptance for retailers - Security and reliability*. (n.d.). Retrieved from visa.com: https://usa.visa.com/run-your-business/small-business-tools/retail.html

Wood, G. (2016). *Ethereum: A Secure Decentralised Generalised Transaction Ledger.* Retrieved from http://gavwood.com/paper.pdf

Zamani, M., Movahedi, M., & Raykova, M. (n.d.). RapidChain: Scaling Blockchain via Full Sharding. *Proceedings of ACM SIGSAC Conference 2018*, (pp. 931-948).