Lysium Network
Scalable Blockchain Infrastructure

Lysium Network Core Development Team
developer@lysium.network
https://lysium.network

Abstract

Distributed Ledger Technology (DLT), also known as „Blockchain", represents a new paradigm in how applications are designed and built. Even tought the blockchain emerged as a consumer-focused technology (form Bitcoin), it's current technological level is not suited for mass adoption. The main hypothesis is that the blockchain establishes a system of creating a distributed consensus in the digital online world. This allows participating entities to know for certain that a digital event happened by creating an irrefutable record in a public ledger. It opens the door for developing a democratic open and scalable digital economy from a centralized one. There are tremendous opportunities in this disruptive technology and revolution in this space has just begun.

Lysium Network is built with the smart-contract functionality and utility in mind. The specificity of the Lysium blockchain relies on the built-in fully fledged Turing-complete programming languag. This feature allows developers to create what are usually called "smart contracts". Smart contracts can be used to describe automatic computations executed in a blockchain. The smart-contracts enables users, developers and partners to enhance the usability of the Lysium network by building complex applications like: Decentralized Exchanges, NFTs, Lending and Borrowing protocols, Stablecoins, etc.

1. Introduction

The algorithm that incentives the holding and running a node (and thus securing the network) as well as fees for the usage of the network is not fair in the traditional blockchain infrastructures:

- Power-users (users that use the network/protocol) extensively are not incentives to continue to use the infrastructure as their total cost of operation are increasing.
- Members that are running nodes and securing the network and not being rewarded the fair share that is generated from the growth of the network. As more users transact more, the reward stays the same for them.

We are proposing a fair system for collecting and distributing the fees generated by the network, as well as keeping the total supply of tokens in a deflationary system.

- The fee for a transaction will be based on the overall consumption of the network from the respective account.

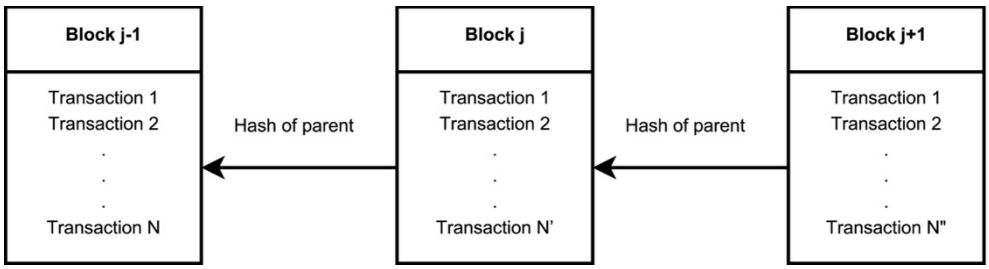
- The system will be deflationary by burning a certain percentage of native tokens (LSX), depending on the type of the block (normal, mega or giga block).

2. Securing the Network

The Lysium Network is a blockchain infrastructure that runs on a set of pre-defined rules.

Each block contains the hashcode of the previous block in the chain (called its parent block), thus forming a chain of blocks that can't be broken. If someone tries to modify a block in the chain, the chain will be broken and it will be rejected by all the actors in the blockchain. Fig. 1 depicts a part of a blockchain:

Fig. 1: Network of blocks



The Lysium blockchain is a decentralized ledger of transactions. No entity is controlling a blockchain, there are only actors participating in its growth and integrity. Based on the Proof of Trade protocol, the system cannot be hacked, modified, or shut down by any of the users.

The transactions in a blockchain are the mechanism for external users to interact with the system. A currency, whose units are called tokens, is associated with each blockchain. These tokens can be transferred through transactions, like bitcoins in the Bitcoin blockchain or LSX in Lysium Network. A unit of Lysium token is called a Lys, and one billion Lys is a GLys. A transaction is always sent from one account and interacts with other accounts. The classical accounts are called “Externally owned accounts” and are owned by one or a group of persons. They are characterized by a token balance, a private key, and a public key.

3. Lysium Block Structure

An innovation of the Lysium Network is the implementation of 3 different types of blocks: Kilo (or normal) blocks, that are issued every 4 seconds, Mega blocks that are issued every 24 hours, and Giga blocks that are issued every 365 days.

The Lysium Network is a blockchain system that validates transactions through the issuance of blocks. A block contains a list of transactions that are confirmed by the network and verified by the validator that signs the block.

The Lysium Network has the following block structure:

```
■ Previous Block Hash: Block hash stored in previous block.

■ Transaction Hash Root: Root node of transaction hash.

■ Receipt Root Hash: Root node of receipt hash.

■ State Root Hash: Root node of world state.

■ Timestamp: Unix timestamp that represents the time when mining
ends.

■ Difficulty: Difficulty is a value that shows how difficult to
find a hash.

■ Nonce: A nonce is a number that can only be used once. In
cryptography, a nonce is a one-time code chosen randomly to
transmit password securely and prevent replay attacks.

■ Gas Limit: Gas limit set for the block.

■ Gas Used: Sum of all the gas used by all transaction in the
block.

■ Extra Data: An optional and free field to store extra data.

■ number: Counting number of the block. The number increments
sequentially. 0 is a genesis block.
```

4. Lysium Block Types

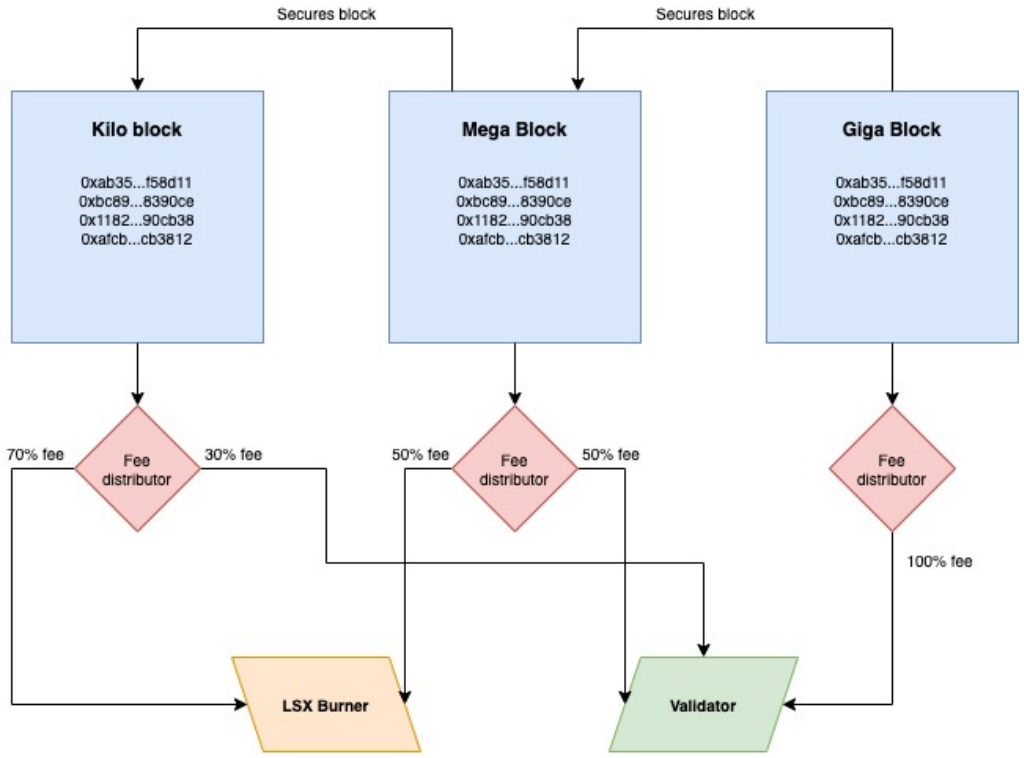
The 3 different types of blocks in the Lysium Network blockchain is implemented with 2 reasons in mind: secure the blockchain by issuing larger blocks that secure previous smaller blocks, as well as for reward distribution.

Most transactions in the Lysium Network will be included in the normal (or kilo) block. As these transactions as mined, part of the rewards will be distributed to the Validator, and part will be burned. Thus, the LSX token is deflationary and it's total supply is constantly decreasing.

As a new larger block is generated (a Mega block is issued every 24 hours), all the older and smaller blocks (kilo blocks), will be hashed and their hash included in the larger block. This is creating a mechanism in which to tamper or modify a kilo block, you also need to modify the mega block that is referencing it.

The same mechanism is applying for Giga blocks, that are issued every 365 days and they are securing the Mega blocks by including their hashes.

Fig. 2: Lysium Network block types



5. Proof of Trade

To incentivize power users and strategic partners to use our infrastructure for deploying their capital and distributed applications, we developed an algorithm that provides a scalable and low-fee system for the users of the network.

The Proof-of-Trade (PoT) consensus is built directly into the blockchain infrastructure and the nodes block-building source-code. The validators (the nodes that are creating the blocks), select the best transactions to be included based on the "*nonce*" of the sender account (address).

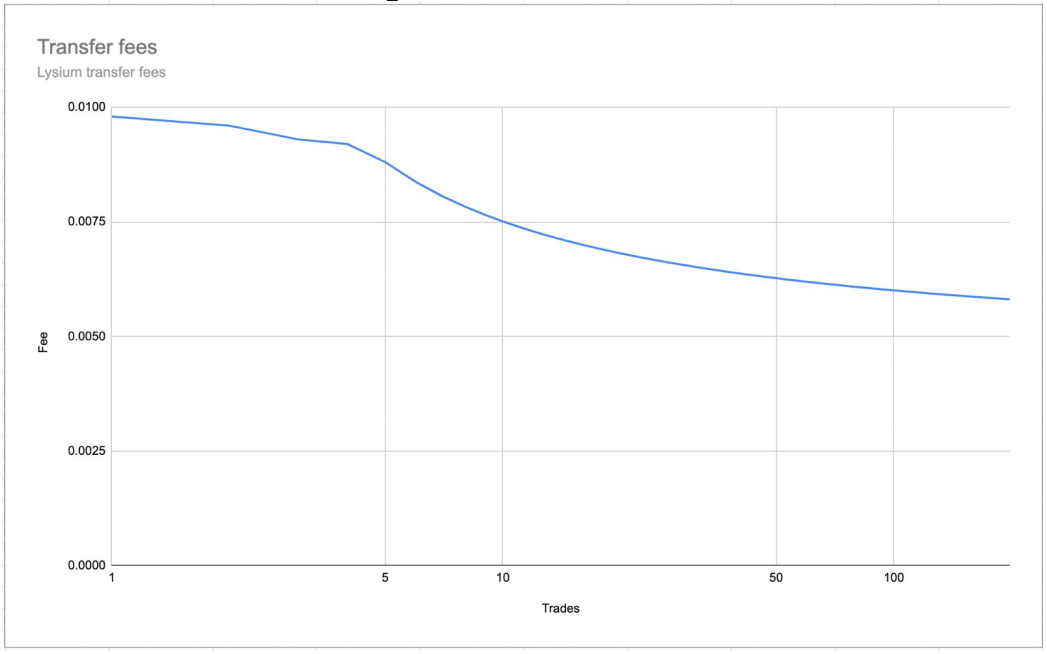
Fig. 3: Fee calculation function

```
47  func (s *senderFromServer) Sender(tx *types.Transaction) (common.Address, error) {
48          if s.addr == (common.Address{}) {
49                  return common.Address{}, errNotCached
50          }
51          return s.addr, nil
52  }
```

The miner(validator) node software orders the transactions based on the "*potLevel*" that represents the number of transactions that respective account has made in the past. By implementing a function that is returning a logarithmic (log) value, we ensure that the users are rewarded to make more transactions, as well as making sure it's still economically viable for miners/validators to process transactions.

The fee system can be described as the following chart:

Fig. 3: LSX Transfer fees



6. Scalability

Scalability is a major differentiator for blockchain networks and one of the most pressing issues in terms of achieving mass-adoption for a network. Lysium fixes the scalability issue by implementing a system that can process 10,000 – 15,000 transactions and reward validators in a fair way.

A key difference between Lysium Network and other decentralized networks is the consensus protocol. Over time, people have come to a false understanding that blockchains have to be slow and not scalable.

The Lysium Network protocol employs a novel approach to consensus to achieve its strong safety guarantees, quick finality, and high-throughput without compromising decentralization. Capable of 10,000 – 15,000 real transactions per second (up to 200,000 TPS with LSX 2.0 Upgrade) - an order of magnitude greater than existing blockchains (fastest EVM-Compatible Chain).

To achieve large level of scalability, Lysium Network plans to implement sharding technology. Sharding is designed to spread out the workload of a network into partitions, which may help reduce latency and allow more transactions to be processed by the blockchain. Without sharding each node in a network must process or handle all the transaction volumes within the network. Nodes in a blockchain are independent and are responsible for maintaining and storing all of the data within a decentralized network. In other words, each node must store critical information, such as account balances and transaction history. Blockchain networks were established so that every node must process all of the operations, data, and transactions on the network.

Sharding is accomplished through the horizontal partitioning of databases through division into rows. Shards, as the rows are called, are conceptualized based on characteristics. For example, one shard might be responsible for storing the state and transaction history for a specific type of address. Also, it might be possible to divide shards based on the type of digital asset stored in them. Transactions involving that digital asset might be made possible through a combination of shards.

Each shard is still able to be shared amongst the other shards, which maintains a key aspect of blockchain technology—the decentralized ledger. In other words, the ledger is still accessible to every user allowing them to view all of the ledger transactions.

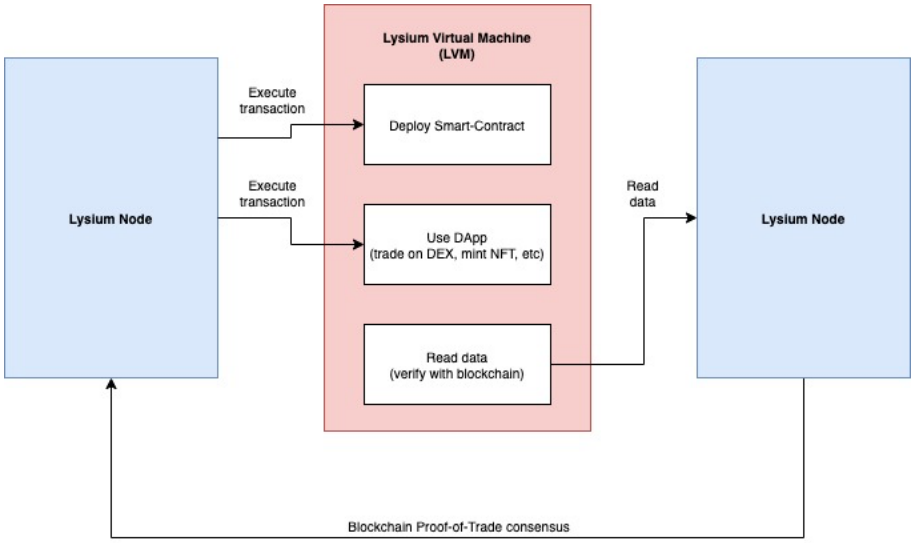
7. Distributed Applications (DApps)

Lysium Network provides an environment to run distributed applications (DApps). These applications are executed as smart-contracts and the network can support a various type of architectures and programming languages. This will bring enormous value and usage for the LSX tokens and increase the number of transactions the networks processes.

Smart-contract transactions are processed through execution of the bytecode (compiled smart-contract) on a virtual machine. Lysium Network has implemented both its own virtual machine name Lysium Virtual Machine (LVM), as well as other executing instances like EVM (Ethereum Virtual Machine) and Tendermint.

Lysium Virtual Machine (LVM) is similar to EVM but it's optimized for our own Proof-of-Trade consensus mechanism. LVM supports all smart-contract applications that EVM supports, thus making the Lysium Network capable of running all the applications that have already been developer for Ethereum.

Fig. 4 – Lysium Virtual Machine



8. Blockchain Security – Cryptographic Signatures

Two of the main purposes of cryptography are to prove knowledge of a secret without revealing that secret and to prove the authenticity of data (digital signature). Cryptography is used extensively within Lysium Network, and one place that users have contact with it is via Lysium accounts.

Proof of ownership of Externally Owned Accounts (EOAs) is established through private keys and digital signatures. The private keys are used almost everywhere within Lysium during user interactions, and the Lysium address of an EOA is derived from the private key. In other words, the Lysium address is the last 20 bytes of hash of the public key controlling the account with 0x appended in front.

Smart contracts on Lysium Network have access to the built-in ECDSA signature verification algorithm through the system method **ecrecover**. The built-in function lets you verify the integrity of the signed hashed data and recover the signer's public key.

As the first virtual machine, Lysium is using Solidity as programming language. Within Solidity, an ECDSA signature is represented by its r, s and v values. The precise mathematical relationship between the public key, the message hash, r, s, and v can be checked to ensure that only the person who knows the corresponding private key calculates r, s, and v. However, due to the symmetric structure of elliptic curves, for every set of r, s, and v, there is another, easy-to-compute set of r, s, and v that also has the same precise mathematical relationship. This results in TWO valid signatures and violates the idea that only the person with the private key can compute a signature.

```
function recoverPublicKey(uint8 v, bytes32 r, bytes32 s, bytes32 messageHash) external
returns (address) {
    address signer = ecrecover(messageHash, v, r, s);
    return signer;
}
```