TECHNICAL WHITEPAPER BRIEF

ico@sphere.ee

ABSTRACT

Sphere is a distributed light-weight network for smart-contracts execution and digital commodities management. It ensures instant transaction execution in designated geo-zone, provides intelligent scalability, guarantees network synchronisation and resilience, and eliminates the need of mining.

Sphere Network. Network core is an open source software written in Scala. It is based on Akka stack deployed in a cloud (Google Clouds and AWS). It maintains processing, verification, execution and chaining of the transaction into its own blockchain. Transaction processing throughput is from **30 000 per second within one geo-cluster**.

SPH tokens. SPH is a cryptocurrency, an investment gold backed token fixed by investment gold price on LBMA (London Bullion Market Association). Tokens are both an asset and the means of payment in the material world and can also play the role of an exchange instrument for the SPH owner. 1 SPH is equal to 1 ounce of gold under LBMA at the moment of purchase or sale.

Mining. Once computing power is used for transactions processing investor's income is formed as a fee for transaction. Unlike "traditional" Proof of Work of mining, SPH mining is absolutely efficient (investor is guaranteed to receive income from each transaction) and environment friendly.

Open API. Open API helps to integrate own marketplace into Sphere solutions ecosystem painlessly. Third-party integrations broaden and scale the Network increasing the number of transactions and thus investor's income.

Third-party integrations. As an open-source software, Sphere can be engaged as a scalable and resilient fintech tool for internal (B2B) and external (B2B, B2C, C2C) intelligent transactions (smart-contracts):

- **B2B.** Organisations can issue their own Sphere-compliment tokens over Sphere networks spawned in a private cloud. It may be used internally (for transactional costs reducing, financial control, accounting simplification and automatisation) or externally (for mutual settlements, smart-contracts, etc).

- **B2C and C2C.** Sphere Platform (see the next chapter) is the reference implementation of B2C and C2C with Sphere, however, one can also deploy own Sphere-based online platform with own Sphere-compliment token to create a new marketplace.

Current whitepaper covers theoretical foundations and technical implementation of Sphere. For wider economical and legal rationale, please, check Legal whitepaper.

HISTORY

Since blockchain technology has been introduced, it had 2 major issues: scalability and trust (consensus). Persisting massive real-time data for million clients in distributed network implies massive async messaging over the wire and eventual data log consistency, which causes scalability problems in classic blockchain approach.

In classical Proof of Work (PoW) model, there's an existential scalability restriction, a mining. As block creation requires more computational resources, its issuing speed is confined with worldwide mining powers and network throughput.

In addition, PoW miners competition is absolutely inefficient, it causes ecological problems and appears to be pseudo-random mining race winner takes all merits for block, but in fact, larger mining pools will always apprehend. Sometimes even a block size is not optimal, but dictated by historical or economical reasons.

Proof of State consensus model, from the other hand, switches from «hashing race» mining to «wallet mining» - nodes, that carry wallets with largest amounts, are «heavier» and thus are more trusted. It leads rewards centralisation and network problems due to unbalanced traffic towards «heavier» nodes.

SPHERE NETWORK

Sphere effectively resolves following modern blockchain networks issues.
Main features are:

- **Cloud native.** Sphere utilises all the advantages of cloud computing. Cloud platform provides all the tools for seamless and rapid development and cluster provisioning with Akka Cluster: intelligent tasks dispatching, auto-scaling and load balancing, monitoring, self-healing, concurrent networking with no idle time.

- **Efficient data sharing** across the network with Akka Distributed Data. Due to efficient biased gossip protocol, a consistency across Distributed Data is gained quickly. With consistency control tools and relying on special CvRDTs (convergent replicated data types) is simplifies concurrency control and data sharing across computing cluster.

- **Efficient data-transfer protocol.** Sphere gRPC high-performance, open-source universal RPC framework that utilises protobuf, an industrial standard protocol for structured data serialisation.

- **Scalable.** Proven scalability up to 30k transactions per second

- **Safety.** All Full nodes are legally identified entities or persons, verified by Estonian police and government and authorised by Sphere's legal partners. All operations, including consensuses gathering are happening under regular clients surveillance.

- **Fair participation rewards.** Full node holders immediately receive their fee for transactions verification and execution.

HOW SPHERE WORKS

There are three fundamental entities in Sphere Network:

- **Client Nodes** (client-side applications communicating with Full Nodes via sRPC)
- **Full Nodes** (computing clusters)
- **Archive Nodes** (blockchain replicators)

Clients Nodes. Client Node is effectively a thin client. As a client-side application Client Node could be installed anywhere - mobile phone, bare metal server, cloud, etc. It requires only a Sphere wallet to become a Client Node. Client to Full node communication utilises **sRPC**, smart-contracts API over gRPC with TLS/SSL transport level security.

Full Nodes. Full Nodes are clusters of computing machines grouped by geographical principle (5 clusters over 4 continents). Full Nodes cluster is spawned on Google Cloud zones.

To operate a Full Node, one must pass ICO as an investor. To learn more about investor status requirements, check Legal whitepaper.

Each Full Node is equivalent to specific **geo-cluster**. Geo-cluster means reducing network overhead. A transaction between geo zones is bound to the issuer locality. For example transaction from the New York to the UK, is served by US geo-cluster.

Full node is mainly responsible for smart-contracts (transactions) computation lifecycle:

- listening to Client nodes
- filling own memory pool with transactions
- computing transactions from memory pool
- creating a new block
- emitting a block to the blockchain
- adopting a block to the blockchain
- carrying recent blocks in memory (blockchain cache)
- carrying network state (state tree)

Full node is rather complex, so we will take a closer look on them in the next chapter.

Archive Nodes. Archive nodes are merely carriers of all Sphere blockchain history. Unlike Full Node, everyone can spawn own Archive Node at bare metal server, private network or cloud to have own fast-access copy of all blockchain history.

Archive nodes form own network mesh for smart replication. For smart replication implementation, lifecycle and monitoring details, please, check Technical whitepaper.

FULL NODES

Full Node is auto-scaled cluster of dedicated servers carrying actors under Akka Cluster provision. Joint network of Full Nodes conforms an internal network mesh across continents.

Full Node itself is designed as reactive unit (actor) and relies on actor-based Akka concurrency model. Full Node is resilient, self-healing Akka Cluster, that carries/interacts with other units, such as:

- **Replicator actor** is a service actor for Distributed Data control provided by Akka out of the box. Distributed Data shares data between nodes in an Akka Cluster, so it is eventually consistent. Consistency levels control is provided out of the box.

- **Memory Pool** is ORMultiMap (memPool implements ORMultiMap) data structure handled by Distributed Data. Memory Pool is a shared map for all transactions came from outside world. The key in memPool is mempool generation (version), the value is GSet (growing Set) with transactions code.

- **Front-end actors** is a listener to Client nodes requests (transactions) within own geo-zone based on Akka HTTP. Once Front-end actor receives a request, it stores its data into Distributed Data with Replicator actor with consistency WriteMajority (2n+1/2 VM in a cluster). Thus, even if a VM carrying this front-end actor dies and/or fails to commit a transaction into memory pool, a front-end node of another VM will eventually do that.

- **Scheduler actor** a singleton actor, handling memory pool drain e.g. dispatching computation of certain mempool generation stored in Distributed Data by Replicator actor. Scheduler assigns attack to compute a certain mempool generation to the least loaded Computational actor. It happens a under special condition, a window moment - whether its atomic counter is near an optimal transactions-per-block number, or it has been timeout (equivalent to Kafka’s linger.ms).

- **Computational actors** are waiting to that task to be assigned by Scheduler actor. Once all the transactions are verified, calculated, and stored into transaction Merkle tree, Computational actor form a block and notifies Scheduler to clean up mempool once that block will be accepted and committed into the Surface (top level blockchain).

The Surface is another Distributed Data, operating across multiple Full Nodes (geo-clusters). It accepts blocks issued from every Full Nodes.

Blocks ordering is guaranteed by special data structure (SHash), that carries all the blockchain (HashSet, ever-growing custom data structure that extends GSet) and two registers (LWWRegister) for new-coming block and current State PATRICIA tree (mainly, wallet balances). It also guarantees updates atomicity and true data synchronisation using version vectors (Lamport's logical clock implementation).

This paper covers the very basics of Sphere. For more details, such as

- cluster topology
- full data lifecycle
- AI module for better window moment prediction
- consensus and consistency levels
- custom data types description
- state handling
- block architecture
- data migration, provisioning, administration
- monitoring

please, check Technical whitepaper.