אleph

Aleph, a decentralized application network

Moshe Malawach (moshe.malawach@protonmail.com)

March 5, 2019

Abstract

This will be a great abstract when I finally get around to writing it.

Contents

1. Introduction

The Internet was originally built to connect computers together. It has been used as a marvelous communication and data storage model. Protocols originally designed for decentralization (TCP, HTTP, SMTP... and even blockchains) are already showing their limitations.

In a purely economically driven world; simplicity, centralization, and non-standard data models tend to win out.

What is our internet today? Silos loosely connected together, users having data in each of those silos (Facebook, Google, and many others), with no simple way to port it over. (see Drake 2015)

There are a lot of attempts at breaking those silos, from connection from silo to silo (ifttt, rgpd for data request, opendata programs...), to decentralized networks of silos (mastodon). We are living in a world of data silos.

In itself, that isn't bad if those silos were easily accessible and decentralized. Users don't know it, but they need to reclaim their data.

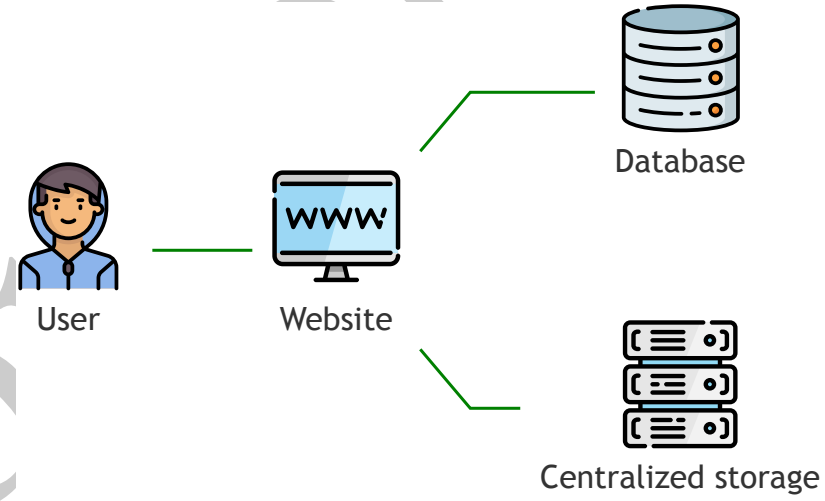


Figure 1.1: Typical centralized application

2. Existing solutions

2.1 Blockchain

DLT[1] allows to have a single data silo (albeit specialized) that is spread accross all users of the network. Smart-Contracts make use of that ability to allow decentralized computing (among other benefits).

Common naming of dApps[2] is using smart contracts on blockchain platforms like Ethereum. A few issues arise from its model:

- Paid transactions by the user (meaning a dApp user should buy some underlying network asset to use the Dapp)
- Reliance on inclusion in a block (dependant on block time of the relevant blockchain, making dApps slow)
- On-chain storage is very expensive, making it unsuitable for medias or big documents.

There is a lot of existing blockchain platforms currently active (see "Coinmarketcap" 2019).

[1] Distributed Ledger Technology, mostly BlockChain
[2] Decentralized applications

2.2 DHT storages like IPFS

Storage networks like IPFS or Swarm are great to find a resource based on its hash in the network. Some things are currently needed or not easily useable for our dApps:

- Incentivization of storage (to be implemented in their FileCoin for IPFS, implemented on ethereum for Swarm)
- Unique resource naming system (provided in a very slow way currently by IPNS, or through Ethereum smart contracts in Swarm)

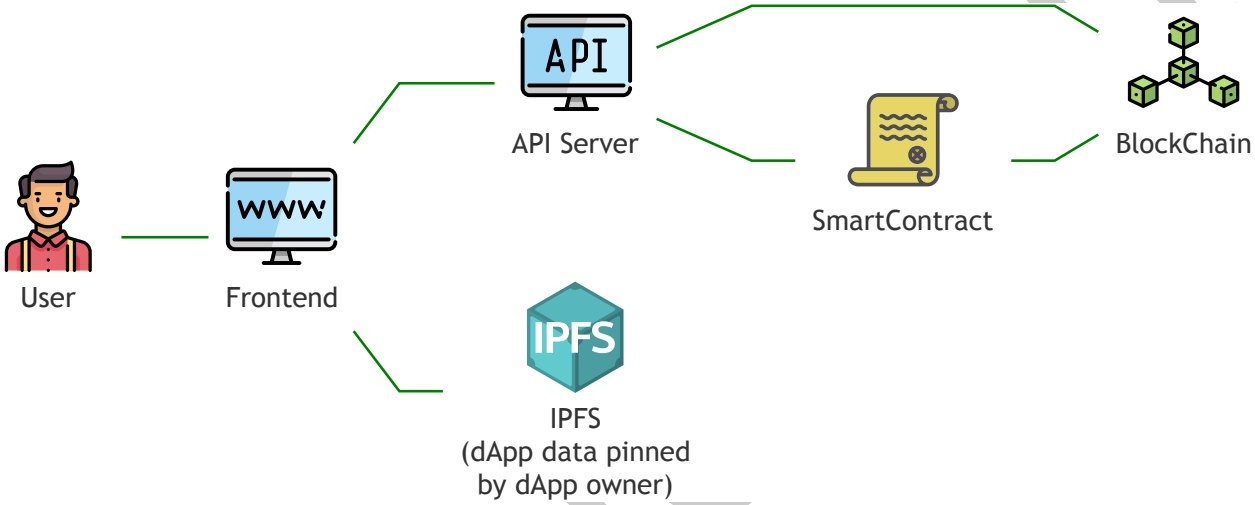


Figure 2.1: Typical dApp using IPFS

3. Aleph Architecture

In the current solutions involving smart contracts, the current state of applications is written (or computed from) a blockchain, this is immutability. But what if you want instant actions, batching user requests for non critical things?

You'll need a second layer of truth, a distributed state that you can influence by signing messages.

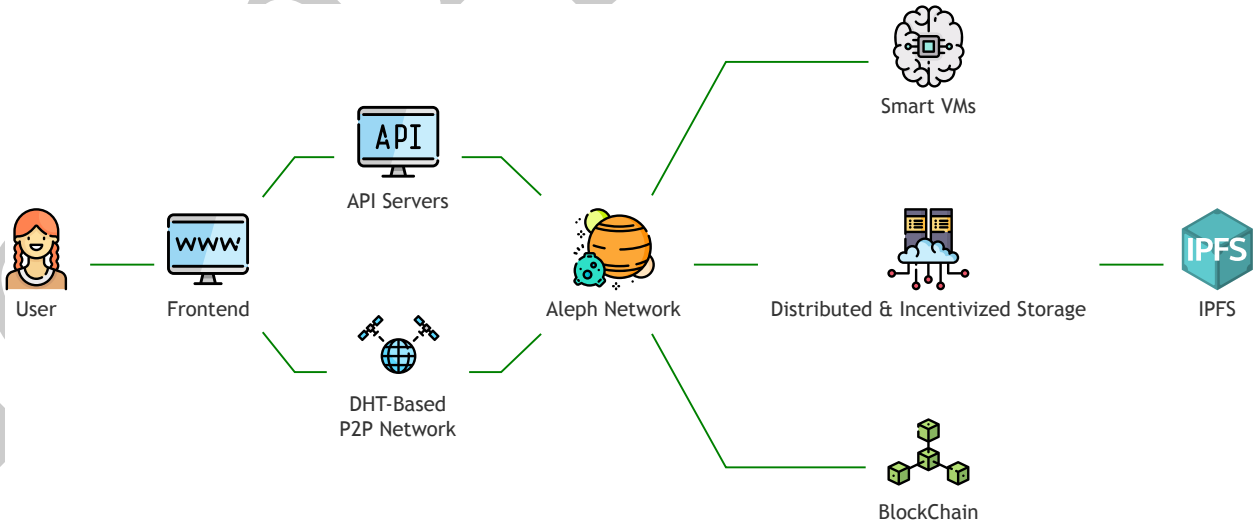
Agents will then commit the messages on the blockchain for the users, incentivized by the network token. They will spend the native chain asset, and will receive the token in exchange for their service, given by the next authorized agent, who will verify that the work has been well done by the previous agent, changing it's credit score on the fly.

This allows for free and instant interaction with dapps for users, with efficient batched insert on the blockchain for immutability.

Moreover, agents are running API servers that can be used by the dapps as entry points to the network.

There is a few main actions of the Aleph network:

- Instant (low-latency) global status update
- Bulk content write on blockchain to allow fee-less commits
- Pinning IPFS data (data replication)
- API server decentralization
- Virtual machines code execution and verification



3.1 Blockchains used by Aleph

The first underlying blockchain supported by Aleph is NULS. To allow easier port of existing dapps and ecosystems, later in the developement process, other blockchains can be supported like Ethereum, NEO and even the Bitcoin blockchain.

The address in the Aleph network is the underlying network of choice (NULS, Ethereum, NEO...) address used to sign messages. A user can post in his personal aggregate (key-value hash table linked to his primary address) his addresses on other networks. That way he can redeem or receive tokens from other blockchains.

If a user has a NULS account and an Ethereum account both linked together, a request for his aggregates (profile or settings for example), post (images, blog posts...) or any other linked content will return the same information from any of the two addresses.

3.2 Data storage

The hashes of the data are stored on chain, the data itself is stored encrypted or not on IPFS. Data is pinned by participating nodes.

Posting is done either via API by the dapps to an API node, or by IPFS directly if available on the browser.

Current state is stored on both blockchain (nodes are made of a blockchain explorer) and current received data that is not posted yet (queue, mempool).

Once the data signature is verified, and it is broadcasted to at least 2 nodes it is considered validated, and it will be included in the coming blocks.

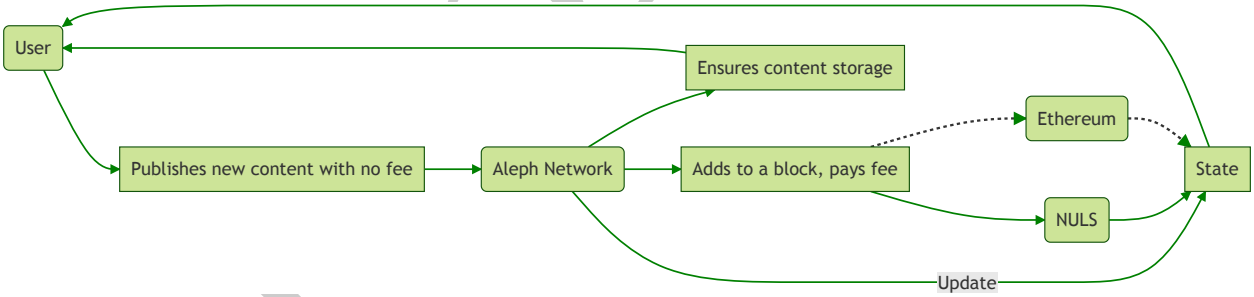


Figure 3.1: Data posting procedure

3.3 Data type

Table 3.1: Data types

Data Type	Description	Details
Aggregates	Key-value store linked to an account[1]	Each key can be updated separately, and its content is merged with previous. *Example*: user profile
Posts	Single data entry	Has a type field, an optional ref (reference) field that references another post or application-specific string. Can be amended with new posts with type "amend" and ref to the previous post. Data inside post content can be optionally encrypted with either owner key or a recipient key. *Example:* blog post, comment, picture gallery, video entry, new data point, new event...
VM State	State of a virtual machine	Fields depending on underlying engine (dockerized language-specific contracts or WASM for example)

3.4 Data exchange

The exchange of data between the dapps and the nodes is done either by pubsub (if available) or API posting, and the node will do the pubsub action on behalf of the user.

Using pubsub all nodes will get the user posts and actions.

Pubsub is using dht and ensuring all subscribed nodes will receive all users posts to get the current state.

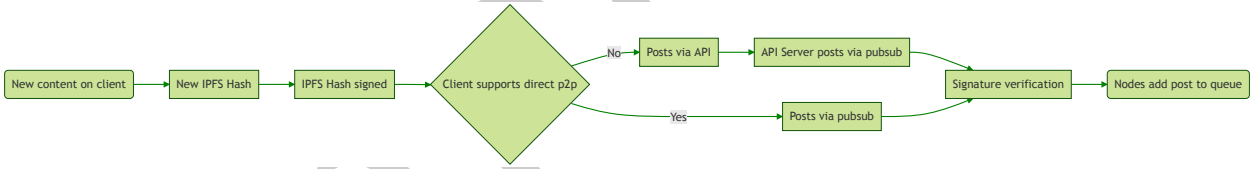


Figure 3.2: Data posting procedure

3.5 State

State encompasses both onchain committed data and uncommitted data received by pubsub. State can be recomputed by getting all TX from a user or containing messages signed by him, and adding the uncommitted messages.

- For smart contracts/VMs, it should be recomputed from last committed, signed and non revoked (no litigation) onchain state.
- For aggregates (user hash tables) from the last onchain commit for each key.
- For posts, from the original post plus all the amends. If last amend contains all the fields, original plus last amend is acceptable.

[1] Underlying blockchain of choice address and linked public key.

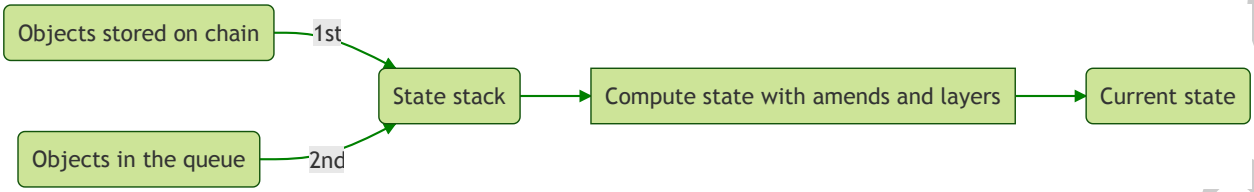


Figure 3.3: State composition

3.6 Virtual machines

Sometime called smart contracts, the virtual machines have a state that is commited as an object in the chain. All nodes don't need to keep all the VMs states, only the last one if no litigation happened.

3.6.1 Hash links

Each mutation to a vm state contains a relation to previous state (previous hash), if two data blocks relate to the same hash, the first to be commited on-chain will prevail (in case of data sync), to be received (in case of off-chain processing before block inclusion).

In the event where a new block is posted with a different history than what is computed off-chain, there is two possibilities:

- On-chain content is deemed bad (according to specific rules) and discarded in a new commited transaction, off chain content taking its place,
- Off-chain content is discarded and clients notified

3.6.2 VM State content

The VM state object is specific to the engine used (WASM, docker-like language specific VM, JVM...), but typically contains:

- Relation to previous state (object chain using hashes)
- Function called
- Arguments
- Function result
- New serialized state (can be a diff only for big states, in special cases)

3.6.3 Concurrency

While view-only functions that don't modify the state can be done concurrently, the write functions can't happen concurrently (or will end up in a fork). A node can send a pubsub message to broadcast it is working on/executing a "write" function on a specific VM, then once it's done, it will broadcast the new state to the other nodes in the same pubsub channel that will be able to take on from this point in history if they also have a write function to execute.

3.7 Cross-application data exchange and data ownership

All applications in the Aleph ecosystem are talking to the same data storage. Conventions exists and have to be documented on data structures to be used by applications.

That way applications can use the same user profiles (example nuls.space and nuls.world/social using the same profiles, and nuls.world/apps/vote using the profile and same post format than the formers for its content) and content sources if needed. A new developer can come and make a new frontend to all the existing data posted by users. Users own their data.

3.8 Moderation and requests for removal

3.8.1 Network and data moderation

Each application can have its own storage of black-listed content and addresses to handle its own moderation features (the application will then omit certain content or content creators from the user interface).

For illegal content, network will also have its own storage, synchronised between nodes of blacklisted content and addresses. Linked content hashes will be automatically unpinned from all nodes, leading to its destruction from the Aleph network.

3.8.2 Removal request

To comply with regulations (GPDR in particular), in a similar fashion to the illegal content data unpinning, a user can request certain hashes he posted to be removed, or even his full address data to be put into blacklist by signing a message using his public key.

3.9 Nodes

3.9.1 Clients

Simple clients to the network are either using the API to an API node or p2p pubsub connection to the network. If they are using the p2p system they can act like a full node and validate messages themselves (and run VM themselves if it's available for their platform, WASM working in web browsers).

They should be careful to verify the signatures of the received messages and states if they don't trust the API server or if they are connected directly to the network.

3.9.2 Packing nodes

The Packing nodes are those who validate the messages and submit their hashes to the underlying blockchains. They get rewarded for their action (token reward) or punished if they don't do their duty correctly.

If a packing node is found doing nefarious actions by the consensus, its address is blacklisted and his rewards (and/or deposit, this is to be defined, see Roadmap chapter) frozen.

Those nodes must be both API nodes and storage nodes too.

3.9.3 Storage nodes

The storage nodes pin incoming IPFS hashes to ensure distributed (geographically and accross multiple hosts to ensure the data won't go away) storage of the content.

Those nodes must be API nodes too, and can be, optionally packing nodes too.

They get rewarded in tokens for their duty.

4. Aleph Token

4.1 Rewards

Three main sources of reward and token creation are:

- Reward each signed message written to underlying blockchain.
- Reward storage of application data (pin items) and availability of API, both are mandatory for nodes.
- POCM NULS token locking in NULS underlying chain (see token distribution secion)

Of those rewards, a part will come from monetary creation, and a part from dApp owner incentive to prioritize their apps and their user fees.

If a dApp owner wants to get the benefit of the network without having to rely on third parties only, he can run a node of the network where he prioritize his dApp data (but can't ignore completely others, having a backup for his users on other nodes, while he gives the same allowance to other dApps).

4.2 Token use

Most node owners will accept free storage of dApp data, especially at start of the network. Once the data grows bigger, and having all the data on all the hosts starts being impractical, dApp owners can pay for storage and writes to blockchain of his dApp data.

Alternatively, application users can pay themselves for the storage of their data (example, big data volume like picture storage applications).

4.3 Token details

The first issuance of the Aleph utility token will be made as an NRC-20 token first, which will be able to be swapped for the native asset at a later date.

If technically possible (part under research):

- the switch between nrc-20 and native aleph token will be available in both directions.
- the token will also be made available as an erc-20 token

4.4 Initial token distribution

Exact details will be announced later on, here is the planned token distribution (can be subject to changes).

To reward the NULS community and ecosystem, a big part of the supply will be airdropped to NULS token holders.

1,000,000,000 (1 billion) tokens will be issued initially.

Of those :

- 150M will be airdropped (most likely to NULS token holders, details will be anounced through official channels),
- 250M will be reserved for an extra incentive on POCM mining program
- 600M to the Aleph team (who will use this for bootstrap period rewards, might be able to sell some for development funding, allocate a part for a community or foundation fund or any other use it might deem necessary)

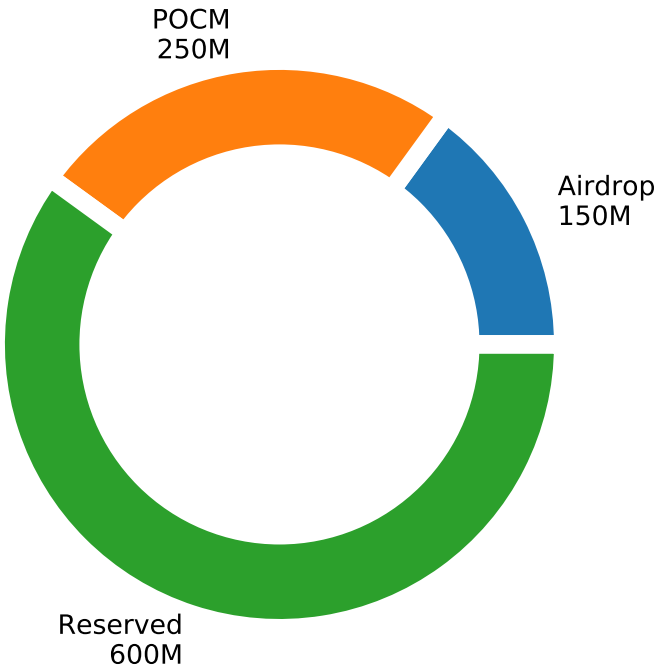


Figure 4.1: Token Distribution

4.4.1 NULS staking program (POCM)

In addition to a regular airdrop, NULS holders can lock a part of their NULS tokens to get Aleph tokens.

While their tokens are locked, they will receive Aleph tokens based on the NULS amount they "staked"/"locked".

Those tokens will come from an initial pool of 250M tokens, locked for that purpose. Once that pool dries up (after approximately 2-3 years), the reward will be lowered and the tokens minted using this program will be taken from the Supply Evolution.

4.5 Supply evolution

Once the full network is ready (see roadmap chapter) and through the described rewards the token supply will approximately grow by 10% the first year, then the inflation rate will decrease by 2% each year until it reaches 4% per year and stays at that value.

Table 4.1: Supply per year

	1st year	2nd year	3rd year	4th year	5th year	6th year
inflation	10%	8%	6%	4%	4%	4%
tokens minted	100M	88M	71.28M	50.37M	52.38M	54.48M
total supply eoy	1100M	1188M	1259M	1309M	1362M	1416M

While the part devoted to the team and developement may seem high on issuance, the inflation rate makes it lower than the rest of the circulating supply during the 3rd year.

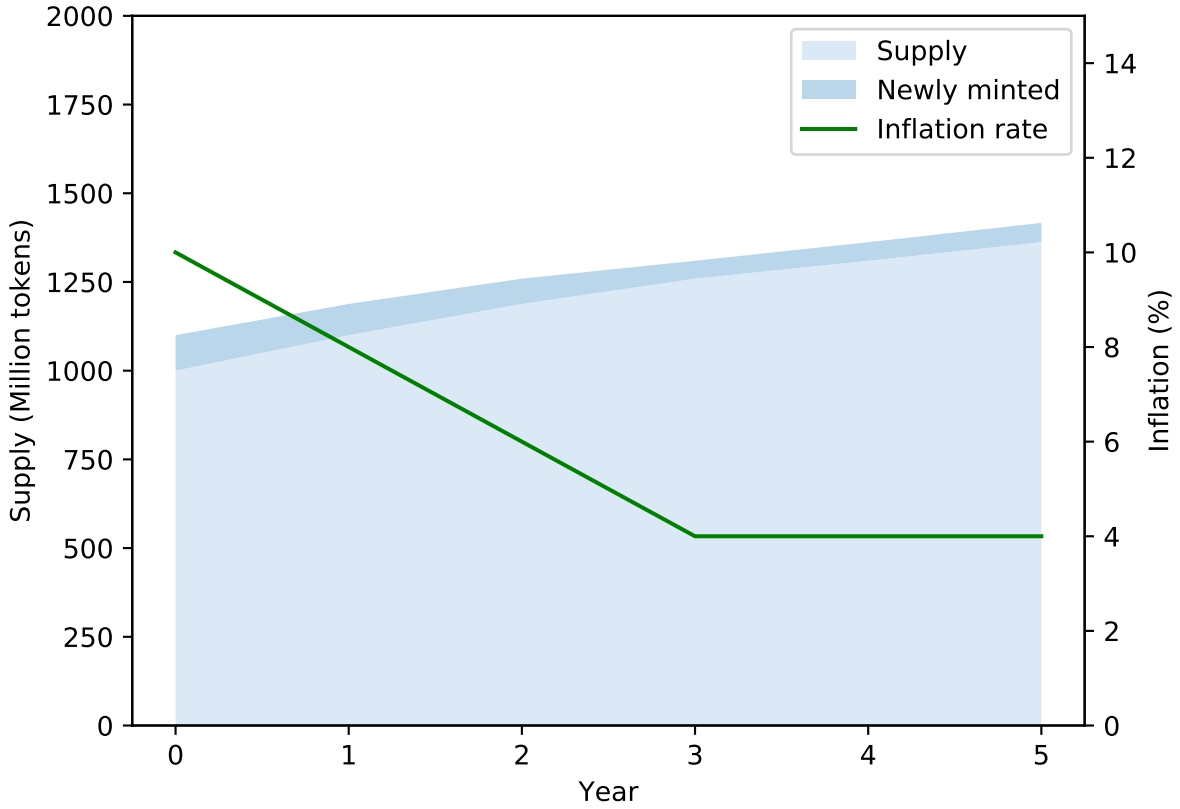


Figure 4.2: Supply evolution

5. Roadmap

5.1 Genesis (NULS Only)

In the first step of the project, the token will live on the NULS blockchain as an NRC-20 token. Rewards to the nodes of the network will be given from the Aleph reserve funds.

The only address format and public keys supported on the network will be the NULS addresses.

At the start of this period, a proof of concept will be ready and applications can be developed on Aleph, supporting all but cross-chain and virtual machines.

5.2 Cross-chain

The cross-chain infrastructure will be developed at this milestone, allowing other kind of addresses to be used on the Aleph network (Ethereum first).

5.3 Virtual machines

The "Smart Apps" virtual machine will be developed for this milestone, allowing the Aleph native token to live in a virtual machine by itself.

A token swap will take place from the nrc-20 (and erc-20 if ever issued) token to the native token.

5.4 Network nodes and economy

If technically possible, other chains token (aleph token as nrc-20, erc-20 or nep-5 for example) will be able to be swapped from and to the Aleph native token.

The network nodes will now receive their token rewards corresponding to the plan described in the Token chapter.

5.5 Specialized nodes

The priorization feature will now be developed and all nodes won't store all the data or run all the VMs. Some nodes can focus on a few dApps for their owners while still being able to help the whole network (and their dApp content can still be accessible to others).

5.6 Future

There is a lot of possibilities for the future (not defined yet):

- Decentralized exchanges and encapsulation of other assets
- Use of Aleph as a monetary 2nd layer (like lightning or plasma)
- ...

References

"Coinmarketcap." 2019. Accessed February 27. https://coinmarketcap.com/all/views/all/.

Drake, Kyle. 2015. "HTTP Is Obsolete. It's Time for the Distributed, Permanent Web." September 8. https://ipfs.io/ipfs/QmNhFJjGcMPqpuYfxL62VVB9528NXqDNMFXiqN5bgFYiZ1/its-time-for-the-permanent-web.html.