Swarm Incentive MVP

Rationale

This document describes the MVP (Minimal Viable Product) for the incentivization track of Swarm.

Swarm as an incentivized decentralized P2P storage and communication infrastructure depends on the cooperative behavior of its nodes for proper emergent network operation. In Swarm's design, such cooperative behavior is incentivized, while freeriding and abuse of network resources are disincentivized.

During the Swarm Orange Summit 2019 in Madrid, a dedicated track has been set up to allocate resources in order to implement a basic MVP for incentivization in Swarm.

In this document, the scope and deliverables of this MVP are defined, as well as roles, responsibilities, and the roadmap.

Relevant resources

Swarm homepage	https://swarm-gateways.net/bzz:/theswarm.eth/
Original incentives paper	https://swarm-gateways.net/bzz:/theswarm.eth/ethersphere/orange-papers/1
SW3 paper	https://www.overleaf.com/project/59c298da5b3a6e3b7ec80d56
User stories	https://github.com/ethersphere/user-stories/issues
Incentivization EPIC	https://github.com/ethersphere/user-stories/issues/14
Swarm code base	https://github.com/ethersphere/swarm
Sw3 smart contracts	https://github.com/ethersphere/swap-swear-and-swindle

Status

As of today, 5th of June 2019, the Swarm codebase has a basic p2p accounting layer implemented. This takes care of accounting between each individual peer; every node maintains a balance with each peer.

An RPC endpoint to query a node's balance with a peer is implemented.

Also, a set of smart contracts has been implemented, reflecting what has been specified in the [SW3 Paper].

MVP Description

The goal of the MVP is to provide a first implementation of incentivization.

Only storage layer incentivization is required (no full-fledged PSS incentivization).
The MVP is required to not only do the accounting but actually also address settlement.

No price discovery mechanism will be implemented for the MVP. It will work (for convenience) out of a fixed price set in the code. No price negotiation is being implemented.

The code should allow parametrizing the following variables:

- (Fixed) price for messages
  The architecture so far for accounting is based on a data structure which allows setting prices per message type. These prices should be able to be parameterized so that it is possible to experiment with different prices.
- Payment threshold
  [SW3 paper] describes how there is a payment threshold. When reached (due to overly tilts in a node's balance with a specific peer), the settlement via cheques is kicked off. The paper describes the details of that exchange.
- Peer-ignore threshold
  In [SW3 paper], this threshold is defined as "disconnect threshold". However, it has been proposed to not drop the peer, but to just stop exchanging with it anymore - to ignore it. We should question if just ignoring is actually justified, as it means keeping connections open and thus possibly wasting resources; a disconnect may still be the better option. The MVP shall give clarity on this question.

Requirements for incentives have been announced from the adaptive nodes track and for basic spam protection via **postal stamps** for both storage (syncinc) and messaging https://swarmresear.ch/t/proof-of-burn-for-spam-protection/14.
These requirements are all currently in **research** stage and not implementation-ready. With their own deadline in mind, the Swarm incentive MVP track can **not commit** to include them into the deliverables. However, acknowledging that there is uncertainty about the implicated workload and if additional resources may be required and potentially made available, this should be discussed at the time the requirements are formalized, and then decisions are made about when to implement these, with the explicit freedom to declare them out of scope for this MVP.

User Stories

The Swarm team gathers requirements in the form of user stories (see [User stories]).
For incentives though, strictly speaking, the benefit of using user stories is limited. As described earlier, incentives are meant to mold emergent network behavior as a whole, and not to fulfill user-triggered use cases. It may be helpful to understand incentives as rather non-functional requirements, comparable to ones like scalability or security, rather than user story based requirements.

Nevertheless, a number of user stories can be expressed.

- An overarching epic (in agile development terminology) has been defined at [Incentivization EPIC].
- See balances with other peers https://github.com/ethersphere/user-stories/issues/20
- Automatic settlement: https://github.com/ethersphere/user-stories/issues/21
- See file retrieval price https://github.com/ethersphere/user-stories/issues/22
- Public gateway incentivisation https://github.com/ethersphere/user-stories/issues/23
- Storage node incentivisation https://github.com/ethersphere/user-stories/issues/24

Tasks

User stories are high-level descriptions of work packages. In order to accomplish these user stories, lower-level fine-grained tasks need to be defined.

We identify the following tasks for the implementation of the MVP:

- Define and implement thresholds for
    - Payment
    - Peer-ignore (in [SW3 paper defined as “disconnect threshold”) https://github.com/ethersphere/swarm/issues/1440
- Allow parametrization of threshold and prices to allow experimentation https://github.com/ethersphere/swarm/issues/1441
- Generate go-bindings from the existing smart contracts https://github.com/ethersphere/swarm/issues/1442
- Integrate the go-bindings with the wider code base. This essentially integrates incentives into the swarm node code. https://github.com/ethersphere/swarm/issues/1443
- Implement cheque exchange protocol https://github.com/ethersphere/swarm/issues/1444
- Implement an RPC endpoint to query cheque status on a node https://github.com/ethersphere/swarm/issues/1445
- Complete automatic settlement https://github.com/ethersphere/swarm/issues/1446
- Write a command-line tool to query a node’s balance with other peers https://github.com/ethersphere/swarm/issues/1447
- Write a command-line tool to query cheque status on a node https://github.com/ethersphere/swarm/issues/1473
- Define useful information requests for node operators https://github.com/ethersphere/swarm/issues/1474
- Define useful information to monitor/debug/verify correct Swap functioning (MVP) https://github.com/ethersphere/swarm/issues/1475
- Implement grafana dashboards to visualize accounting metrics (these metrics already exist in the code) https://github.com/ethersphere/swarm/issues/1476
- Revisit existing swap tests and update them https://github.com/ethersphere/swarm/issues/1477
- Define simulation scenarios https://github.com/ethersphere/swarm/issues/1478
- Implement simulations https://github.com/ethersphere/swarm/issues/1479
- Run simulations, observe, analyze https://github.com/ethersphere/swarm/issues/1480
- Create/update Swap documentation https://github.com/ethersphere/swarm/issues/1481
- Design, configure and deploy test cluster / network https://github.com/ethersphere/swarm/issues/1482

- [sc] Drop submitCheque and make sure this is safe for the owner https://github.com/ethersphere/swap-swear-and-swindle/issues/25
- [sc] Documentation update https://github.com/ethersphere/swap-swear-and-swindle/issues/26
- [sc] Get rid of custom encode functions where possible https://github.com/ethersphere/swap-swear-and-swindle/issues/27
- [sc] EthSignTypedData or other signing mechanism https://github.com/ethersphere/swap-swear-and-swindle/issues/28
- [sc] Replace bytes memory with structs where possible https://github.com/ethersphere/swap-swear-and-swindle/issues/29
- [sc] Make sw3 tests more readable https://github.com/ethersphere/swap-swear-and-swindle/issues/30
- [sc] Deploy smart contracts https://github.com/ethersphere/swap-swear-and-swindle/issues/31

- https://github.com/ethersphere/swarm/issues/1502

Acceptance Criteria

- Code review validates the code (developed in a separate branch) and labels it as Swap-1.0
- Swap-enabled nodes are deployed on an independent network.
- Incentivization on that network is ON by default
- Balances can be queried at any time
- Thresholds have been validated through testing/simulations and have reasonable values
- Operation of nodes without Swap enabled is not affected

Planning

Medium time-frame planning https://github.com/orgs/ethersphere/projects/8
Sprint planning https://github.com/orgs/ethersphere/projects/9

Responsibilities

Marcelo Ortelli	Go development Smart contract development
Diego Masini	Research Occasional Go development
Ralph Pichler	Smart contracts development
Rinke Hendriksen	Smart contracts development
Vojtech Simetka	Coordination IOV Labs Requirements Planning
Tim Bansemer	Coordination with other tracks
Fabio Barone	Engineering lead Planning Go development

Roadmap

There is a rough high level project planning at https://docs.google.com/spreadsheets/d/1eMg5ZvC8rfKgsjLMoM9sw9BFpDM96ThS4dXbzmvWvL0/

The above spreadsheet tries to map out in time how we will implement the tasks and user stories described in this document.

However, this planning of course can only be considered a high-level estimate. Software development should be tackled iteratively (widely accepted best practice). There are a few risks and uncertainties involved in this project:

- Theoretical and economical questions which have not yet been resolved through research. A list of the currently known questions has been elaborated at https://docs.google.com/document/d/1H09c1aTViwi5axKn5Ff9-fDbdCAJHaLQq8xpJ9n23rA. This in fact does not even allow to fully specify the requirements for this MVP at this time.

- The very experimental nature of an incentivized distributed p2p storage platform which in this form has never been tried before.
- Development of parallel tracks affecting this MVP (specifically Swarm's Core Track)
- A team with people from different backgrounds and skills which has never worked together before.
- The distributed locations and time zone difference of team members.

From this point of view, it is difficult to have a realistic and binding time estimate into the future. The above high level spreadsheet suggests a time frame of about **18 weeks.** Start date should be considered Monday, 17. June 2019 (discounting this MVP setup phase).

In any case development will be planned and monitored via sprints iteratively. See Planning for the resources used to maintain this planning.