Long-term Storage in Decentralized Environments

Racin W. Nygaard
Supervision: Hein Meling, Roman Vitenberg, Leander Jehl, Veronica Estrada-Galiñanes
Department of Electrical Engineering and Computer Science, University of Stavanger

Motivation

Similar to how Ethereum is being proclaimed as the *World's Computer*[2], a decentralized storage solution can be thought of as the *World's Hard drive*. Serving files on demand and preserving the integrity of them is one of the most important tasks of a storage system. However, when the system is decentralized, a unique set of challenges emerges in order to provide the same guarantees.
Namely, how can we use a set of unreliable storage providers to provide persistent storage.

Background

Proof-of-Storage

Algorithm to check if the storage node has the data. Send probabilistic challenges to storage nodes to gain a measure of node availability and data integrity.

Swarm

Swarm[4] is a decentralized content distribution service and storage platform. It is a native base layer of the Ethereum web3 stack. Participants are incentivized by Ethereum to pool their bandwidth and storage resources together in order to best serve the network.

Manifest

The manifest itself matches to a unique Swarm hash and it typically declares the content which can retrieved through this hash. Each content entry contains its own unique Swarm hash, a string identifier and a content type field.

Alpha Entanglement Codes

A novel redundancy scheme[1] which creates a highly connected mesh of the data and parity blocks, resulting in a flexible and practical erasure codes with high fault tolerance. The code computes the XOR of two consecutive blocks in the mesh as more data is added.

Proof of Concept

To show the possibility of added redundancy, our team implemented alpha entanglement codes as a proxy service on top of Swarm during the Ethereum Hackathon in Madrid 2019. Our project won the 1st prize.
During the upload process, the original file is split into smaller chunks, which are entangled together in the proxy layer in order to create the parity chunks. Then all of the chunks are uploaded to Swarm as regular files, and the corresponding manifest is returned to the proxy. The manifest is then used to create a JSON mapping with the entangled files.

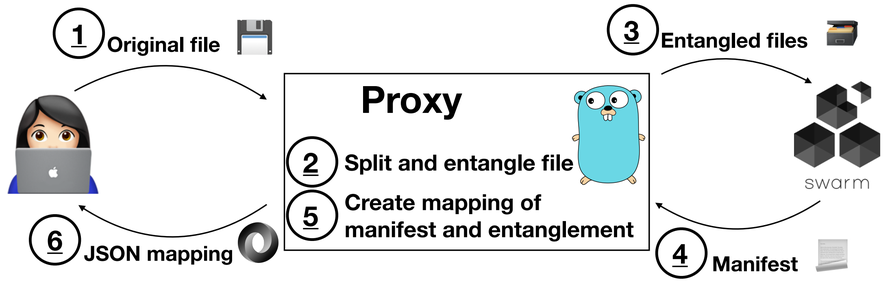

Fig. 1: Upload.

In our prototype, the end user is able to simulate different recovery scenarios by sending a list of unavailable data chunks. The proxy layer will then use the mapping to find the correct parity chunks needed for reconstruction. The chunks are then requested and retrieved from Swarm. Once all necessary chunks are retrieved, the proxy layer will reconstruct the original file and send it to the end user. Note that if some parity chunks are missing, it is possible to run steps 2, 3 and 4 again for reconstruction.

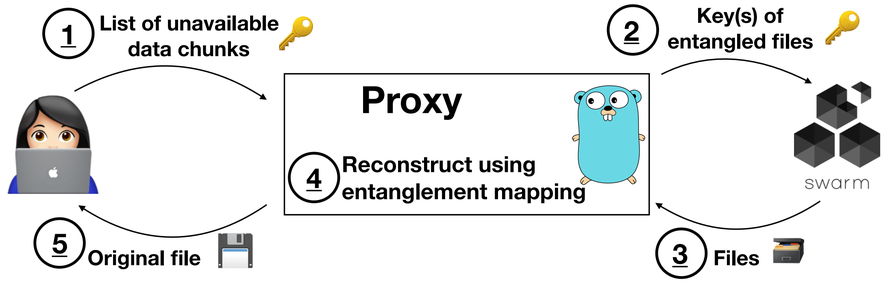

Fig. 2: Download.

Future Research

Going forward, I plan to continue to work on the research prototype in order to increase file and manifest redundancy in Swarm. A successful implementation will allow for running large scale evaluations. We plan to introduce entanglement in two phases, as follows:

1 Entanglements for Random Access

In the first phase, we place the task of repairing at the participants themselves. If a participant notices that a chunk is unavailable in the network, it is able to use two entangled chunks to repair.

2 Entanglements for Catastrophic Failure

A new role, *Insurer* is introduced into the network[3]. The insurer uses Proof-of-Storage to monitor the availability of chunks, and issues repairs if errors are detected. Down the line, it is our hope that this could be used to protect the Ethereum tries.

Acknowledgements

The project has funding from the Research Council of Norway and is developed by the University of Stavanger (UiS) with the partnership of the Arctic University of Norway (UiT).

References

[1] V. Estrada-Galiñanes et al. "Alpha Entanglement Codes: Practical Erasure Codes to Archive Data in Unreliable Environments". In: *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. June 2018, pp. 183–194. DOI: 10.1109/DSN.2018.00030.

[2] Ethereum Foundation. *Ethereum Introduction*. Available at `https://github.com/ethereum/wiki/wiki/Ethereum-introduction`.

[3] Viktor Tron, Aron Fischer, and Nick Johnson. *Smash-proof: Auditable storage for Swarm secured by masked audit secret hash*. Tech. rep. Technical report, Ethersphere, 2016.

[4] Viktor Trón et al. "Swap". In: *Swear and Swindle-Incentive System for Swarm* (2016).