

July 17, lining up the MVP: Tardigrade.js a Front end Dashboard

Taylor Hulsmans

July 2022

Abstract

Another good week developing the pure vision of CAW, which seems to have survived a major liquidity unlock today, which saw the dev overseeing the funds relock them for Oct 24 and Jan 2 respectively. Most of my time was spent beginning to characterize the JS API- Tardigrade.js and piecing the vertical slice together. A Dashboard front end, also for the purposes of testing the npm package is also starting to be coloured in. It is hoped that this place can provide an area to manage orbit-db instances from multiple tardigrade instances, a sort of hub of what ones node is hosting.



0.1 Scaling Ethereum through social methods

At its heart, Tardigrade utilizes specialized forms of state channels to achieve efficiency. What makes the project innovative is that it does not try to provide liveness with cryptoeconomic miner/watcher incentives, security with fraud proofs or finality with challenge windows. Instead, the beneficiaries of a message chain has an incentive to pin on IPFS and distribute it over pubsub- ergo

liveness is now user driven. Security is proxied for reputation derived from the social graph by which the state channels derive initial state, it too is now user driven. Transaction finality is not given first class citizenship, as long as a signature chain is processed before the deadline, state update is attempted-Tardigrade just does the best it can. In short, Tardigrade doesn't try to be the secure and efficient scaling solution, it tries to be the perfect insecure and efficient scaling solution. Tardigrade isn't for sending millions trustlessly to a user across the world, its for sending cents inconsequentially to many people, and hopefully, as the social network grows, sending tens of cents.

0.2 Tardigrade.js

While technically all horizontally integratable and independent of each other, the smart contracts, the IPFS pubsub, and the orbit db have to work together to produce the fluid user experience. As such, I've opted to compose Tardigrade.js, an npm package that can be used to integrate any project with Tardigrade. From creating and managing orbitdb databases for message chains, internally keeping used nonces and batching duplicate micropayments, to creating and adding users to the social graph, Tardigrade.js provides an easy to use API for web browsers and nodejs applications.

0.3 Dashboard

Given that Tardigrade is publish-subscribe, users will be hosting their own content through IPFS, knowing what one is hosting than, or pinning, alongside any other platform specific information will be helpful. The dashboard is a react front end that aids in this process, which also serves as a place to test the npm package