Tenfold Protocol – Light Paper

Introduction

The biggest hurdle for blockchain and DApp adoption is scalability. Existing blockchains are plagued by slow confirmation speed, low transaction throughput, and high gas cost.

We scale Ethereum and other blockchains with a novel approach: Tenfold Protocol, a layer-2 scaling solution for securely maintaining a state machine (i.e,. a deterministic database) off-chain while reading its state on-chain. It's compatible with any blockchain that supports smart contracts, including Ethereum.

By leveraging Tenfold Protocol, developers can build large, sophisticated blockchain applications that are not possible to run fully on-chain due to scalability issues. Over time this should lead to more robust, diverse DApps running efficiently and therefore more scalable use cases for blockchain ecosystems.

The design of Tenfold stems from the insight that blockchain transactions fundamentally fall into two categories:

- Transactions whose value are unbounded. Examples include ETH transfers: there's no upper limit (other than that imposed by the token supply) as to how much ETH can be transferred in a transaction.
- Transactions whose value are bounded. Examples include updating a CryptoKitty's breeding cooldown: while breeding cooldown is a valuable attribute, its value is conceivably limited.

While unbounded transactions should be processed on the blockchain for maximal security, Tenfold enables DApps to process bounded transactions off-chain, thus vastly improving performance and reducing gas costs.

Tenfold Protocol uses a cryptoeconomic approach to handle bounded transactions while remaining compatible with other technologies such as Plasma for handling unbounded transactions.

A Case Study

To make the use case of Tenfold more apparent, we will examine HyperDragons, a blockchain game that is deploying Tenfold into production. At the time of writing, it's a top-5 blockchain game in terms of daily active users (DAUs) and transaction volume.

HyperDragons consists of many mini-games, one of which is a battle game where players pit their dragons against one another. The battle is fairly complex: dragons may cast skills and spells on their opponents.

To ensure fairness and transparency, the battle currently takes place fully on-chain. As a result, it currently costs up to hundreds of dollars to execute a single battle. Therefore the developer had to limit the amount of battles to 3 per day, while each battle can only accommodate 32 users.

With Tenfold, battles will be executed fully off-chain, while results of the battle will be securely recorded on-chain. The results are needed on-chain because they are used for further decision-making: for instance, a winning dragon may acquire a badge on-chain, which augments its value.

The new version of HyperDragons using Tenfold will accommodate up to 2,048 characters in each battle, for a practically unlimited number of games a day.

Architecture

From a high-level, Tenfold consists of three major components:

State registry: each state registry securely records the state of a blockchain application. **Validator network**: a network of validators connected through a P2P broadcast network and a decentralized filesystem; the validators are responsible for securely updating the state registry. **Liquidity market**: a market where token holders can lend their tokens to validators for staking.

We will walk through the components and how they are tied together in the following sections.

State Registry

The state registry is fundamentally a **token-curated registry** (TCR). A TCR is essentially an on-chain database where each update goes through a stake-based voting process to decide whether it should be applied. In particular, if someone submits an invalid update which then fails the poll, they will lose tokens to the validators who correctly voted against the update.

In our system, each blockchain application (e.g., a game) will have its own state registry, which stores the state relevant to the application. For instance, in the battle game we described above, the state registry may store the match results.

Validator Network

In Tenfold, validators are nodes that monitor the state registry and ensure that only correct updates are applied to the state registry. When someone attempts to update the state registry by submitting a **state proposal**, the validators vote on whether the update should be applied or not.

Here we have two questions to answer:

- How does a validator know if an update is correct or not?
- Why does a validator have the incentive to vote correctly, or to vote at all?

Deciding if an Update is Correct

To ensure that a validator can figure out if a given update is correct, we leverage the concept of a **state machine**. In computer science, a state machine is a program that, given the same sequence of inputs and the same starting state, always arrives at the same output. In other words, it's a deterministic program.

In Tenfold, each blockchain application is modeled as a state machine. A validator can download the state machine via a decentralized, content-addressed filesystem. The inputs to the state machine are broadcast through an authenticated (i.e., signed) P2P network. In the current iteration of Tenfold, we are using IPFS for the decentralized filesystem and secure scuttlebutt for the P2P broadcast network, respectively.

As a result, each validator is able to independently maintain a copy of the blockchain application and thus can decide if a given update correctly reflects the state of the application or not.

Incentives for Voting Correctly

Before addressing incentives, we need to clarify that in Tenfold there are two kinds of tokens:

Application tokens (AT): the tokens issued by the application developer, i.e., the user of Tenfold. Application tokens are used in the stake-based voting process for securely updating the state registry. In this section when we refer to "tokens," we are referring to application tokens. **Protocol tokens** (PT): the tokens issued by the developer of Tenfold (us). We will explain the utility of protocol tokens in the next section.

The incentives for token holders to vote correctly have been heavily explored in TCR literature; at a high level, these insights include:

- If token holders do not vote correctly, invalid updates will be applied.
- If invalid updates are applied, the application will cease to be useful and users will drop out.
- If users drop out, the demand for the tokens drops, since the tokens are only useful within the application. Furthermore, the market's confidence in the token will falter, which results in further decrease in token demand.
- If the demand for the tokens drops, the price of the token drops.
- Since token holders do not want the price of the tokens to drop, they are therefore incentivized to vote correctly.

Liquidity Market

If we closely examine the system we described so far, we will notice one crippling weakness: there is a mismatch between application token (AT) holders and validators. AT holders may not be able to validate because they lack the necessary computation, storage, and network resources; whereas people with the necessary resources may not be able to validate because they lack AT.

Without a sufficient number of active validators, the security of the system cannot be enforced. Therefore, we solve this problem by building a liquidity market, i.e., a market where AT holders may lend their AT to validators for staking. The token economics of the liquidity market will be explained in detail in a future document.

Comparisons with Other Solutions

Alternative blockchains (e.g. EOS, DFINITY, NEO)

Tenfold is a **layer-2** scaling solution, meaning that it's an off-chain solution that builds on existing blockchains. It's completely agnostic to the underlying blockchain as long as the blockchain supports general smart contract capabilities. Therefore, it's compatible with Ethereum as well as alternative blockchains such as EOS, DFINITY, NEO, etc.

Plasma

Plasma, in its current form, only handles the secure transfer of crypto assets. It cannot be used to execute general state updates. As a result, its use so far has mostly been limited to decentralized exchanges.

Tenfold is completely compatible with Plasma; our goal is to eventually build a unified platform that supports both Tenfold and Plasma, so developers can update state *and* move assets off-chain.

State Channels

State channels are a secure way to execute general computations off-chain; however, they have a couple well-known limitations:

- The set of participants has to be fixed. For new users to join or for existing users to leave, on-chain transactions are required.
- The communication overhead is `O(N^2)`, since every operation has to be signed by every participant. As a result, each state channel can only accommodate a very limited set of users.

- State channels need to be implemented in smart contract languages such as Solidity, since disputes must be settled on-chain. In contrast, Tenfold allows application logic to be implemented in arbitrary languages.

Furthermore, Tenfold works with any general state machine written in any language, whereas state channels must be implemented in smart contract languages since disputes may eventually need to be settled on-chain.

Perhaps most importantly, while some projects have used state channels in one form or another, **generalized state channels**, or state channels that can be used to implement arbitrary state machines, are still very much a work-in-progress both from a theoretical and an engineering perspective. It's still months, if not years, away from production usage.

State channels do have advantages over Tenfold, however. Notably, state channels are **cryptographically secure** whereas Tenfold is **cryptoeconomically secure**. That is, the security of state channels relies on nothing more than math, whereas the security of Tenfold depends on the majority of nodes acting in an economically rational way, i.e., in their best self-interest.

Truebit

Truebit is a computation marketplace where one can post a (potentially very heavy) computation task and receive the result on-chain. Since the marketplace itself is on-chain, there's significant latency in posting tasks and receiving results, which means Truebit is not well-suited for interactive applications such as games.

Furthermore, Truebit suffers the same problem as generalized state channels in that it's still very much a work-in-progress and very far from production usage.