OpenST Protocol White Paper

Benjamin Bollen, Nishith Shah, Lionello Lunesu, Sunil Khedar, Antoine Cote, Jason Banks, Jason Goldberg, Matt Chwierut, Brian Lio

Draft published for peer review* v0.8.3, last updated 5 November 2017

1 Introduction

Ethereum introduced to the blockchain toolset stateful accounts. The storage space associated with these accounts is write-protected by code, and we refer to these accounts as smart contracts[1] This general purpose capability of Ethereum has spurred a vast wave of innovations and a leading use-case of Ethereum has been the ability to create a token on top of Ethereum. For tokens on Ethereum, ERC20[2] has recently been adopted as the standard interface for a smart contract defining such a token. However in order to engineer utility into a token to incent greater uptake, several more challenges become apparent. Some of these challenges include latency, transaction fees, scale and privacy; for these challenges we attempt to contribute towards solutions in this technical whitepaper. Other challenges come from legal requirements and economic modeling to support a token economy; confronting these forms an integral part of the Simple Token project[3].

With Simple Token we set out to be pragmatic about the capabilities of existing decentralization technologies and look to find answers that solve for these problems today with a clear roadmap towards internet-scale performance. All the while we strive that all technical mechanisms are open and independently cryptographically verifiable.

OpenST operates as a non-profit and governs the development of the OpenST Protocol. In addition it performs high-level guardian tasks on the instance of the protocol that is associated with the Simple Token EIP20 token on public Ethereum. These guardian tasks are limited but necessary when a technical answer only is insufficient. A primary example can be the review of a new member company which desires to launch its own branded token within the OpenST platform.

First we establish a lexicon to help present the new patterns OpenST Protocol introduces in the token space. For a young token economy, speculative value of a token can easily drown out the intended utility. We describe how a utility token can

*for review contact review@simpletoken.org

[1]We use smart contract interchangeably for the code associated with an account, or the stateful instantiation of that code; the meaning should be clear from the context.

[2]See EIP20 (formerly ERC20)

[3]Find our thinking on governance, economic modeling and incentives in OST sidepapers.

be built on top of value assets that back it.

We introduce Simple Token as a freely tradable EIP20 token on Ethereum mainnet. Simple Token can be staked as a valuable crypto-asset to mint utility tokens. Furthermore Simple Token functions as the base token on the utility chains accounting for gas consumption.

We continue to outline how desired behavior can be enforced on the utility tokens so that the token serves the user transactions within an existing consumer application. We call such utility tokens branded tokens. We describe how user financial sovereignty is preserved on the blockchain as a user can choose to hard-exit the value of the branded tokens on Ethereum mainnet.

In the next section of the protocol we describe how rich interactions in consumer applications can be mapped to fundamental transactions on the blockchain by providing an API for developers to integrate OpenST into their application - making the development of a consumer tokenized economy simple.

While these last mechanisms are off-chain mechanisms, we consider them an integral part of the protocol to enable any third-party developer to build on top of the OpenST Platform. Additionally including them in the protocol enables open innovation and audits to realize best practices for minimizing correlatable data on chain of pseudo-anonymous accounts that could empower inferring personally identifiable data or application metrics. Future work in this part of the protocol layer as such includes technology to increase the noise-to-signal ratio on-chain, or reduce the signal by moving it off-chain with payment channels.

To conclude we describe how no trust needs to be placed in the validator pool of the utility chain, as in the case of chain halting, all ownership state can be carried back to the value chain.

We recapitulate the protocol at a high level using an explicit example and build on this example to illustrate how an end-user can use Simple Token on Ethereum to obtain and interact with a branded token.

We describe how OpenST Platform can be an open network of utility chains, serving different consumer applications and provided by third-parties.

While the OpenST Protocol and Platform concerns only application logic - and OpenST will focus on implementing it as Ethereum smart contracts - and off-chain technology, it is still worthwhile to detail our considerations with respect to the available chain technology upon which the OpenST Platform can execute. We discuss these architectural requirements lastly and how OpenST can contribute to existing projects in this area. This concludes the OpenST Platform as the second part of this paper.

Lastly we put forward the roadmap for OpenST.

2 Related Work

OpenST has been born out of the chasm between two worlds. One side holds the promise of open payment networks and financial sovereignty of users in a digital world. On the other side sit millions of potential users for whom the technical adoption curve is steep, and businesses who are looking to tokenize, but who need to

work within existing regulations and tax law. Challenges currently facing cryptocurrencies and applications hoping to leverage tokenization include user-experience, economic and legal constraints and the opportunities that can be unlocked with the right technical solution. While the OpenST protocol white paper may read at times like a scalability proposal - and obviously a scalable architecture is a requirement - it is not, nor do we intend it to become that. We set out to build on and contribute to the work done by great teams to present a protocol that helps bridge cutting-edge blockchain technology with mainstream consumer applications.

Interledger Protocol - Interledger uses a two-phased escrow on two ledgers where a connector has funds on both ledgers and functions as a market maker between the two chains; additionally allowing for a graph of connectors to transfer across multiple hops. We drew inspiration from the Interledger protocol, but solved for a different problem: we look to mint a new token representation on a utility chain of the value staked on a value chain.

Tendermint - Tendermint is a leading protocol for cryptographically sealed Byzantine fault-tolerant, Proof-of-Stake consensus. It is used / implemented by Ethermint, Parity and Hyperledger Burrow.

Cosmos - Cosmos by Tendermint is a framework for interoperability between blockchains. It has pioneered the concept of InterBlockchain Communication (IBC), which we apply here in a specific context for transferring proofs of utility between Ethereum and a utility chain. Cosmos as a network also provides structure between different chains. We explore for utility chains to be natively compatible as Cosmos zones (when running Tendermint consensus).

Lightning / Raiden - work on payment channels has both inspired the OpenST protocol, and also forms a natural complement to OpenST. Raiden can transfer Simple Token between different utility chains (and Ethereum mainnet). Payment channels specifically can be hosted by member companies to scale the volume of transactions for their branded token off the utility chain. In this sense payment channels also aid in protecting user privacy and help shield a member companys inner metrics of its application to the outside world, while being open and accountable with their branded tokens.

Plasma - while we didnt learn about Plasma until it was published, we are enthused that OpenST can be seen as a very specific application of some of the ideas that are also abstractly proposed in the Plasma white paper. We welcome the alignment and look forward to mutual contributions.

Polkadot - Polkadot is a protocol for heterogeneous general state transfer between multiple chains. With OpenST we aim to solve a particular problem, but see benefits to strengthening the guarantees against Byzantine validators across utility chains.

Casper - as OpenST spans out over many utility chains with all value staked on Ethereum, all work to strengthen and scale Ethereum greatly benefits the ecosystem, and as a result OpenST.

3 A Protocol for Simple Tokens

3.1 Staking Value for Utility

The OpenST Protocol establishes a bridge between two differently purposed blockchains. A *blockchain*, which is required in order to hold cryptographically secured valuable assets; and a *utility blockchain*, which has utility tokens in favor of which the assets are held on the value blockchain. The utility chain needs to support lower transaction fees and lower transaction confirmation times than the value chain. We put forward this condition seeing as one desired outcome of the OpenST Protocol is to enable micro-transactions within mainstream consumer applications using the utility tokens.

We present the discussion for two Ethereum-based chains, but note that this is not a requirement for the value chain.

Through the lens of the Simple Token project and the Simple Token EIP20 token on public Ethereum, the value chain in our discussion is planned to be Ethereum mainnet. For the utility chain we consider an Ethereum-based chain with a Byzantine fault-tolerant consensus engine which seals blocks cryptographically, as examples Proof-of-Authority[4] or Proof-of-Stake consensus engines.

[4]Proof-of-Authority is not Byzantine fault-tolerant, but it can still be considered useful in the context of utility chains.

3.2 Establishing a Bridge

To establish a channel between two chains, we require both chains to have a light client smart contract on each chain tracking the blocks on the other chain. In several configurations prior or ongoing work already exists.

When we take into consideration the specifics of these chains then we can consider specific light client contracts which relieve the responsibility of a central party on mutually committing the latest state hashes on the other chain.

As Ethereum mainnet operates a Nakamoto consensus engine (as the value chain) there is a soft requirement to set a threshold for the number of block confirmations to wait for until a state transition on Ethereum is considered finalised. If the utility chain is cryptographically validated by a known set of validators then those validators can each report the block hashes they have seen on public Ethereum and a block hash is considered final when consensus among the validators is reached.

In the opposite direction, to report the latest block hash of the utility chain to the value chain, the logic involves no subjective threshold parameter when the utility chain is cryptographically sealed by a known set of validators as a complete light client can be implemented as a smart contract on Ethereum.

In general a value chain is a Proof-of-Work generated chain (for the near future at least), while a utility chain for efficiency reasons should be assumed to be cryptographically sealed. It is therefore worthwhile to note that this asymmetry is by design: any halting or Byzantine failure of

the utility chain can be proven by any user on the value chain to forcefully recover the staked assets on the value chain after a sufficiently long grace period should the utility chain fail to recover. This way users are assured to always be able to recover their original assets on the value chain. On the other hand a value chain can hard-fork at which point the utility chain can evaluate out of band whether to track both or one of the forked value chains going forward.

As a net result of having complementary light client tracking contracts on both chains, the smart contracts have at their disposition knowledge of the state root of the other chain[5]. Such transactions committing the blocks allow consequentially for state from the other chain to be asserted as true only if a Merkle-proof for those state variables can be proven against the committed root hash.

Committing the root hash of the alternate chain allows for any party to present statements of what is true on that chain, removing the need for trusted oracles or trusted parties. A second benefit of this approach is that it strengthens immutability of either chain as the latest blocks are anchored into an independent chain. In particular if the utility chain is cryptographically sealed, anchoring the latest block on regular intervals into a (Proof-of-Work) value chain prevents any of the validators from rewriting the block history.

[5]What we outlined here is in effect an application of the concept of InterBlockchain Communication (IBC) as proposed by Tendermint in the Cosmos Whitepaper.

3.3 Two Sides of the Same Token

Tokens form a natural basis upon which to build functional sharding. Tokens, like smart contracts, are contained within the blockchain they are defined on. Unlike smart contracts, tokens have a universal metric for coarse graining, namely their total supply, and the extrinsic property of their market valuation.

If we consider the total supply of a token then all forms of monetary transactions leave the total supply of tokens unchanged. Whether it concerns a send transaction, an escrow, or even an Interledger protocol exchange across different ledgers, within the chain defining the tokens, the total supply is unaffected by these transactions.

By grouping value on a value chain we can denominate that value as a new token, a utility token, and transactions of the utility token do not change the total amount of grouped value. If the utility token would be defined on the same blockchain, then we would not have gained additional transaction throughput capacity. However, by defining this utility token on a new chain, the utility chain, transactions of the utility token need not concern the value chain, and we have a logical model for functional sharding of transactions.

When a user adds crypto-assets on the value chain to the grouped value, she should expect to receive an equivalent amount of utility tokens on the utility chain. While we call this process minting utility tokens, no value is created, only a new representation of that original value is created while the value assets themselves are locked.

In the reverse process, if she can prove

ownership of utility tokens and intent to return them in favor of an equivalent stake of the grouped value, then our user can do so at any point. As the utility tokens are backed for the full value at any time, all users can run on the bank and they would all recover their full value on the value chain.