This post was written collaboratively by@adlerjohn, @matt, @mikerah, and @villanuevawill (ordered alphabetically by last name).

Abstract

We present a novel methodology for constructing trustless two-way bridges between a main chain and a side chain. By halting the side chain's progress permanently, we no longer need tight synchrony requirements and long withdrawal delays even in the optimistic case. Instead, withdrawals are only allowed after the side chain has been halted for a long period of time. Once the side chain is halted, atomic swaps can be used to withdraw funds immediately for use, through a liquidity provider network. A new side chain can be launched in tandem to halting the old one to accept and begin processing swapped funds.

Background

Prerequisite Reading

- Fraud and Data Availability Proofs: Maximising Light Client Security and Scaling Blockchains with Dishonest Majorities
- Building Scalable Decentralized Payment Systems
- Minimal Viable Merged Consensus
- On-Chain Non-Interactive Data Availability Proofs
- Simple Fast Withdrawals

Problems with Existing Proposals

Current side chain proposals have several problems that we resolve in this post. We will go through some of the more common proposals in this section:

Merge-Mined Side Chains

Merged mining involves re-using proofs of work by allowing miners to mine two or more chains simultaneously. This is used for systems like Rootstock as a way of bootstrapping the launch of a side chain.

Unfortunately, merged mining only provides security against external hashrate, it does not provide security in the general case. Additionally, it does not

in any way enable a trustless two-way bridge of funds (i.e.

, moving coins/tokens from/to the main chain and side chain without trust in a third party).

Federated Peg Side Chains

Federated pegs are two-way pegs in which the gatekeepers of the backwards peg are trusted third parties, known as a federation

. The federation cannot release funds from the side chain to the main chain without a majority of its members agreeing to this through a multisig. Even though the federation doesn't directly control the actual funds, users can have their funds stuck in the side chain indefinitely until a majority decides to unlock the funds. Users are therefore at the behest of the federation.

They are usually used when the main chain doesn't have a lot of expressivity in its smart contract functionality e.g.

, Bitcoin Script. In order to effectively lock and unlock funds between the side chain and the main chain, proofs need to be verified on-chain, which may require extensive functionality. This is how systems like Liquid and Rootstock work.

There is another type of federated two-way peg system calledDrivechain, in which the federation is an implicit subset of the main chain's miners. Even though enrollment in the federation is permissionless in this case, we still have the same problems as explained in the previous paragraph, namely that a majority of miners can steal or lock funds arbitrarily.

Plasma Chains

Plasma Cash, inspired by the earlierCoinWitness, uses a coin-based data model (unlike Bitcoin's UTXO data model or Ethereum's accounts data model). This has the nice feature that all coins on the side chain are known to the main chain—similar to opened channels. The trustless two-way peg is maintained through the use of complex exit games, which are

necessary for the peg to be maintained no matter what happens on the side chain, including operators withholding blocks or producing invalid blocks.

The issue that arises with these exit games is that they are interactive and must be performed on a live, ever-changing side chain. Therefore they need to account for possibly fraudulent exits of spent coins and invalid spends, which ends up preventing the effective use of watchtowers and requires that users hold on to state (coin history). This forces users into rigid synchrony requirements and necessitates that every user run a full node.

Two-Way Bridge Through Halting

Assumptions and Requirements

This post uses the term “side chain,” though it can be equivalently substituted with “execution environment with delayed state execution,” as we shall show in a later post.

Funds are deposited to the side chain via a one-way bridge (i.e.

, funds are locked in the contract with no immediate way of retrieving them). Execution on the side chain proceeds as it would normally. How this is done is mostly orthogonal, though we require side chain block headers be available on-chain and a solution to both invalid blocks and unavailable data. Invalid blocks can be resolved through validity proofs or fraud proofs (though fraud proofs are preferred), while unavailable data can be resolved by either posting all data on-chain as calldata

on request/all the time or by using non-interactive on-chain data availability proofs.

Any data model can be used for the side chain, including a UTXO data model (like Bitcoin), an accounts data model (like Ethereum), or a coin-based data model (like Plasma Cash, and indeed the scheme presented here can be used to augment Plasma constructions). The execution model may also be extended to generalized state execution via both arbitrary predicates and programs.

Two-Way Bridge

We note that the majority of complexity, cost, and safety assumptions of exit games for side chains to allow for trustless two-way bridges (i.e.

, Plasma) stems from the fact that validity must be enforced at every exit attempt through an interactive game, in perpetuity, within a very tight window of time. This makes these schemes especially vulnerable to chain-congestion attacks, and significantly limits their safe capacity.

We can reduce this complexity almost entirely by simply disallowing withdrawals until after the side chain has halted

and a sufficiently long period of time has passed (say, a few months—a system parameter). Until that time, anyone may post a fraud proof or data unavailability proof to invalidate any number of side chain blocks. After that time, withdrawals are allowed with a non-interactive inclusion proof against the state of the latest valid side chain block (i.e.

, the side chain’s canonical tip).

Fraud/data unavailability proofs may be posted earlier, but aren’t necessary to guarantee the safety of funds on the side chain (i.e.

, guarantee that funds cannot be stolen). This is an implementation detail of the particular consensus protocol used to progress the side chain, as is its fork choice rule.

The synchrony requirements of this scheme are much weaker than those of previously-proposed layer-2 techniques: actors in the system only need to come online to validate data during the very long period of time after halt, and only once. In essence, rather than requiring the side chain to be continuously implicitly validated, the final state of the side chain is only implicitly validated once.

Halting the Side Chain

The biggest outstanding question is now “who gets to decide when to stop the side chain?” A multisig federation may have too much control and griefing potential, or may never halt the side chain. Stake-based voting is manipulable and may also result in the side chain never halting.

As it turns out, the simple answer is the most elegant: the side chain simply halts after a predetermined fixed number of blocks. This also has the nice effect of discouraging ossification of the side chain: users will withdraw their funds then deposit them into a new side chain, potentially with improved specs or fewer security holes. This can be abstracted out at the wallet level, so is not a significant UX blocker.

Fast Withdrawals: Liquidity Providers

In order to avoid withdrawal delays, liquidity providers can purchase ownership of funds on the side chain via an atomic swap. This is similar to tokenized fast exits proposed for Plasma, though no funds are being withdrawn. Liquidity providers will take the associated risk of fraud and cost of capital lockup, and pass them on to the user for a profit in exchange for liquidity. By running a side chain full node however, they will be able to make this purchase with high confidence in the validity of the underlying asset.

Only a single liquidity provider is required for fast withdrawals, but not to enforce the trustless two-way bridge. Being a liquidity provider is entirely permissionless, only requiring capital. Liquidity providers are incentivized to ensure they are swapping valid funds, and therefore ensure the side chain tip they are swapping against is valid, by whatever means necessary.

Fast Resuming: Staggered Side Chain Genesis

While we now have a way for users to quickly withdraw their funds to be used thanks to liquidity providers, user funds are now on the main chain—which may be slow, expensive, congested, or involve understanding the complexities of gas fees. To resolve this we can launch a new

side chain, potentially with new rules (similar to a hard fork), immediately after the old side chain is halted.

The new side chain can accept deposits right away, and so users can atomic swap funds from the old side chain then deposit them to the new side chain seamlessly (with UI integration). The new side chain can also accept batch withdrawals-into-deposits after the long period of time to post fraud proofs has passed for the old side chain.

With proper wallet integration and a permissionless liquidity provider network, this allows users in almost all cases to not even know they’re using a particular side chain, or that their funds have migrated.

Future Work

In a future post we will show how this line of thinking can be extended to enable a wide assortment of execution environments with delayed state execution on Eth 2.0 in sharded context.