Finality: A Scalable Protocol for Trustless, Permissionless Cryptocurrency Interoperability

Ho Lee Huo, Yoshi Kumatama
www.finality.network

Abstract. We propose a scalable second layer protocol allowing any willing party to build and operate a trustless and permissionless cryptocurrency exchange capable of processing millions of transactions per second. A root blockchain contract owned by the operating party securely holds user funds, which are tradable within its sidechain with instant finality. This design is secured by the operating party, periodically committing the merkle root of user accounts and proofs from the prior merkle root to the root blockchain contract. This second layer design serves as an extremely high-throughput, secure platform on which to build trustless, permissionless cryptocurrency exchanges. A future Finality root chain, with its own native consensus-generating Finality cryptocurrency, will connect Finality second layer protocols operating on various root chains to establish cross-chain interoperability.

1. Introduction

Cryptocurrency exchange has come to rely almost exclusively on trusted centralized exchanges that emulate the custodial financial institutions Bitcoin challenged [1]. While these centralized platforms provide high liquidity and transactional throughput for traders, they still suffer from the inherent weakness of the trust based model. They take custody of and monetize user funds, and abuse their power and influence to dictate coin listings appeasing the interests of their old-world business entities. Non-custodial peer-to-peer exchanges provide trustless escrow through on-chain order books and settlement, but fail to attract liquidity because market makers must bear gas costs to create, modify, and cancel orders. 0x and IDEX move the order book off-chain to cater to market makers, but 0x matches orders on-chain and requires settlement before allowing traders to reallocate their funds [2], and IDEX trusts a transaction processing arbiter to push queued matches to the root blockchain [3]. First layer protocols by design are unable to provide a high-throughput and trustless exchange platform that rivals centralized exchange trading.

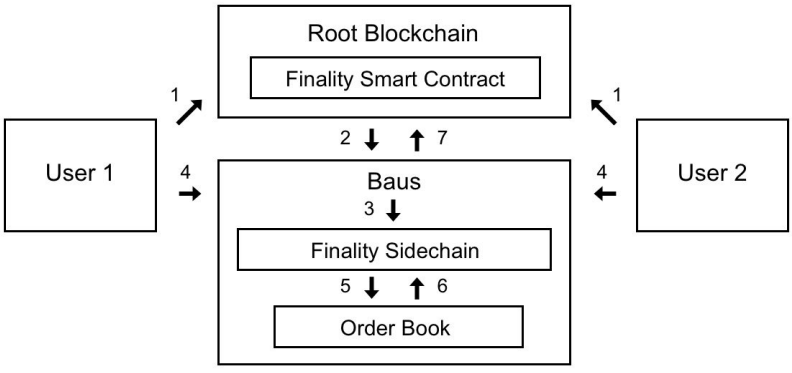
Second layer protocols promise greater scalability while maintaining the security of the root blockchain. Second layer decentralized exchange protocol Tesuji Plasma proposes More Viable Plasma specifications to group atomic UTXOs together in sidechain blocks to optimize speed [4-6]. However, the UTXO design falls short in providing an optimal structure for trading as partial fills result in constant splits of UTXOs. As the value of UTXOs become lower than the cost to exit them to the root blockchain,

these coins are essentially stuck in the second layer. Producing ownership proofs to exit each tiny UTXOs is computationally intensive.

What is needed is a scalable account-based second layer exchange protocol secured by the publishing of randomized merkle proofs rather than trust. Users may create, modify and cancel orders for no gas cost, and an operating party matches orders to update a ledger of account balances on a sidechain. Apart from on-chain deposits and withdrawals, only hashed trade transactions formed by two signed orders may amend the ledger. To ensure data is made available, the operating party periodically publishes a merkle root of the current ledger and a random selection of proofs from the prior ledger to the root blockchain. Users may challenge any publishing of this operating party, who forfeits a stake and reverts the sidechain to the last confirmed state given malfeasance. Malicious trade transactions and account balances are economically impractical to publish, which ensures the operating party is honest with near certain probability. Users funds are protected and may be withdrawn at any point from the root blockchain contract. In this paper, we propose an open design standard upon which any willing party may build a secure and scalable exchange where users are custodians of their own funds and may trade at millions of transactions per second.

2. Design

Secured by a root blockchain ledger, Finality aligns natural incentives between users trading cryptocurrency, a Finality Sidechain Block (FSB) producer called a *Baus*, and incentivized participants acting as *Watchers* to facilitate the trustless and permissionless exchange of cryptocurrency. Watchers may be users or third party observers. The following diagram outlines the process of users depositing funds into the Finality Smart Contract (FSC) and submitting signed orders to the Baus, who maintains an off-chain order book whose data is transmitted to the network. Only matched orders produce trade transactions that are included in the Finality Sidechain within the merkle root the Baus publishes on the root blockchain:



1. User 1 deposits ETH and User 2 deposits FNY into FSC
2. Baus listens for deposits on Finality Smart Contract
3. Baus credits user accounts on the Finality Sidechain
4. User 1 submits buy order on FNY/ETH pair with unique hash and User 2 submits sell order on FNY/ETH pair with unique hash

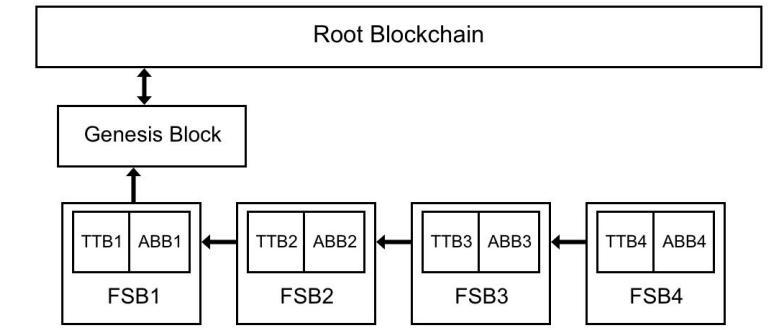
5. Baus verifies that User 1's ETH balance and User 2's FNY balance have sufficient funds and adds orders to order book
6. Trade is matched by Baus and added to the Finality Sidechain
7. Baus submits the merkle root of account balances to the root blockchain
8. Challenge period commences

2.1. Finality Smart Contract (FSC)

Each Finality Smart Contract features an operating Baus and address which receives user transactions for deposits, hibernates, withdrawals, and challenges. The FSC serves as a secure, trustless escrow for user funds as they trade by directly transmitting orders to the Baus. The Baus periodically publishes merkle root state changes and proofs to the FSC, and the FSC uses the seed of the root blockchain block creation times to to randomly request a proof of m user account balance proofs the Baus must submit alongside the following merkle root publishing.

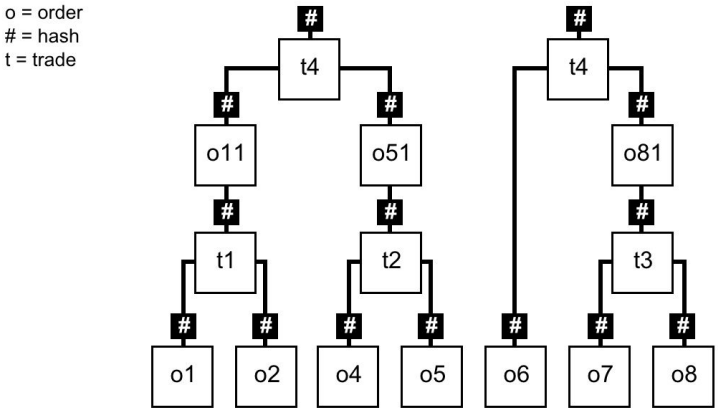
2.2. Finality Sidechain

A Finality Sidechain is operated by a Baus and derives its security from the FSC on the root blockchain. The Baus matches user trades and updates account balances to produce Finality Sidechain Blocks (FSBs), which consist of a Trade Tree Block (TTB) and an Account Balance Block (ABB). The FSBs form a sidechain linking back to the genesis block and constantly transmitted to the network:

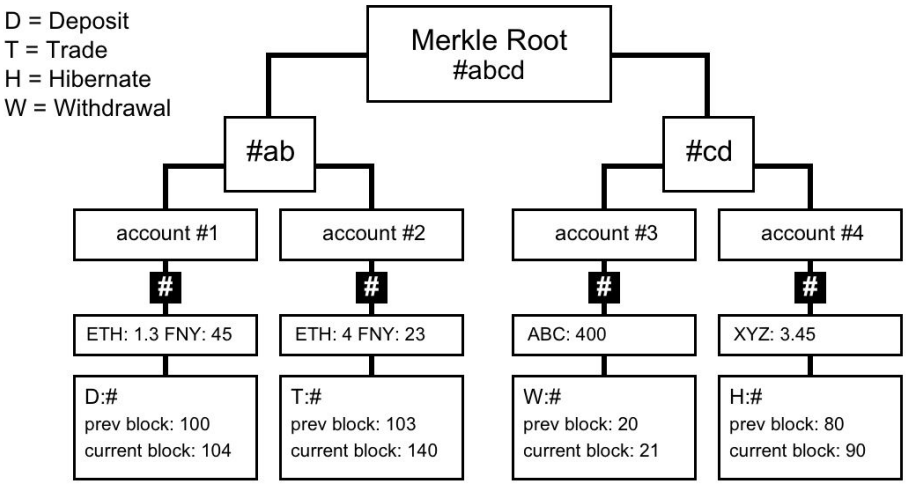


FSB = Finality Sidechain Block
TTB = Trade Tree Block
ABB = Account Balance Block

The TTB is a set of binary trees whose leaves are matched orders and whose nodes are trade transactions that correspond to two updated accounts on the ABB. Each step of the TTB requires a hash as follows:



The ABB is a merkle tree whose leaves are user accounts containing all current coin balances and a hash of either a deposit, withdrawal, hibernate, or set of trades in a FSB. When a user completes a trade in the current TTB, previous trade hashes are replaced by *prev_block_number* and a hash of trade hashes from the current TTB are included as follows:



The following conditions must hold true for every FSB:

1. The sum of account balances in each ABB must reconcile with the sum of account balances of the previous ABB including net deposits, withdrawals and hibernates
2. The sum of user account balances debited by trades in the current TTB must reconcile with the sum of these user account balances in the previous ABB including fees transmitted to the Baus
3. All matched trade transaction nodes in the TTB must contain hashes that are referenced in the corresponding ABB
4. No user account contains a negative balance

3. Transactions

Users send deposits, hibernates, withdrawals, and challenges in on-chain transactions to the FSC, and submit orders directly to the Baus.

3.1. Deposits

A user initially deposits on the Finality Sidechain by sending ETH or any ERC20 to the FSC address. After the transaction reaches the Baus' root blockchain confirmation requirement, a Finality Account ID is created pointing to the user's deposit address and the Baus adds the user account within the ABB on the Finality Sidechain. Future deposits from the same root chain address to the FSC address credit this existing account. Users may not deposit a coin into an account in the process of a withdrawal for that coin.

3.2. Orders

A user must provide a proof of ownership of the adequate coin balance and a unique hash to place orders with the Baus. The order is broadcasted as:

$$(< order\_type, public\_address, token, quantity, price, base\_currency, nonce >, signature)$$

The Baus collects orders in a local order book, and if $price_{bid} \geq price_{trade} \geq price_{ask}$ the Baus matches the bid and ask orders to generate a trade. This creates a transaction node with a unique hash in the TTB and updates two user balances in the ABB. The trade transaction is broadcasted as:

$$(< trade\_status, unfilled\_quantity, unfilled\_order\_hash, token, trade\_quantity, trade\_price, buy\_order\_hash, sell\_order\_hash, fees\_in\_base\_currency, buy\_order\_user\_account\_updated\_values, sell\_order\_user\_account\_updated\_values >, baus\_signature)$$

Each trade transaction results in either a fill or a partial fill. In the case of a fill, $trade\_status, unfilled\_quantity$ is zero and $unfilled\_order\_hash$ is null. In the case of a partial trade, $unfilled\_quantity$ is the remaining order quantity from $partially\_filled\_order$. A partial order is broadcasted as:

$$(< order\_type, public\_address, token, quantity, price, y, base\_currency, prev\_trade\_hash, prev\_trade\_block\_number, original\_order\_hash, original\_order\_block\_number >, trader\_signature).$$

3.3. Hibernation

Hibernates are used to securely lock user funds within the FSC. A user initiates a hibernate by providing a proof of account balance, which features the $block\ number$ and hashes from the root to the $account\_info$ of the block. A user must cancel all open order to hibernate, which updates the account balance on the root blockchain. This secures the account balance from any updates if the user wishes to go offline for a period of time.

The Baus and Watchers are able to challenge this hibernate for t blocks with a proof of account balance challenge.

3.4. Withdrawals

A user initiates a withdrawal by providing a proof of account balance, which features the $block\ number$ and hashes from the root to the $account\_info$ of the block including all deposit, trade, hibernate, and withdrawal events in the user's merkle leaf. A submitted withdrawal mints a custom ERC 721 non-fungible token transmitted to the withdrawing account's address that represents that particular live withdrawal. The withdrawal must survive the challenge period of t blocks before the funds will be transmitted from the FSC to the address holding this non-fungible token. Users may trade this non-fungible token with other accounts as they wish, and the account holding the token at expiration time of the challenge period will receive the withdrawal amount. While the withdrawal challenge period is active, the user is not permitted to deposit the same coin. Each user has the ability to challenge again until the challenge period is over.

3.5. Challenges

Watchers may challenge false deposits, withdrawals, hibernates, orders, and trades for a time frame of t FSB's by submitting a challenge transaction to the FSC. This challenge period incentivizes Watchers to maintain the integrity of the Baus.

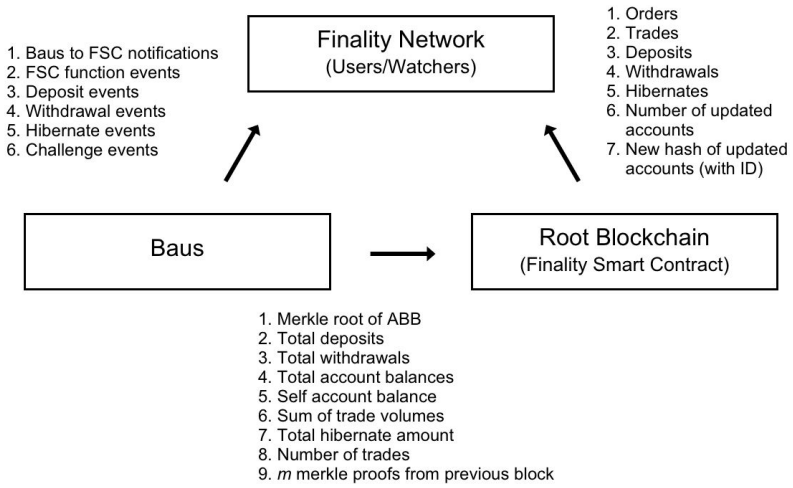
- **Proof of Account Balance -** If the Baus submits a user deposit, hibernate, or withdrawal that does not correspond sufficient funds in their account balance, a Watcher may submit a proof of account balance by providing the $account\_info$ of a newer block, and the hashes from the root to the $account\_info$.
- **Proof of Trade -** If the condition $price_{bid} \geq price_{trade} \geq price_{ask}$ does not hold true for a trade, a Watcher may submit a proof of trade challenge by providing the block number and hashes from the root to the $account\_info$ that contains the $trade\_hash$ along with the $trade\_info$.
- **Proof of Order -** If the Baus includes a trade linked to a user order that is not valid, a Watcher may submit a proof of order challenge by providing the user account merkle leaf containing the invalid order.
- **Proof of Duplicate Order -** If the Baus attempts to re-trade an already completed order, a Watcher may submit a proof of duplicate order challenge by providing the user account merkle leaf referencing the $block\_info$ and $trade\_hash$ of the completed trade linked to the original order. This challenge includes the order nonce, which points to that order's unique hash.
- **Proof of Residual Order -** Each partial fill creates a new residual order in the order book. A malicious Baus may create an invalid partial order or double trade the partial order to economically benefit a colluding user account. In both cases, any Watcher may submit a proof of residual order challenge by providing the $block\_info$ and $trade\_hash$ linked to the residual order.
- **Proof of Duplicate Residual Order -** If the Baus attempts to re-trade an already completed residual order, a Watcher may submit a proof of residual order challenge by providing the user account merkle leaf referencing the $block\_info$

and $trade\_hash$ of the completed trade linked to the original residual order. This challenge includes the order nonce, which points to that order's unique hash.

Given a successful challenge, the malicious FSB and all subsequent FSBs are rejected, and the Baus forfeits its stake on the root blockchain to the successful Watcher. User account balances revert to the last confirmed state on the root blockchain, and user funds are withdrawable given adequate proof of ownership submissions.

4. Incentives

As users submit orders to be matched by the Baus, incentives built into the Finality ecosystem facilitate the flow of information between the Baus, the root blockchain (FSC), and the Finalty Network (Watchers) as follows:



4.1. The Baus

The Baus receives signed user orders to maintain an off-chain order book, match trades, produce FSBs, and periodically publish #1-9 in the diagram above to the FSC on the root blockchain. The FSC requests a #9 merkle proof of a random m of n accounts from the prior state publishing [8]. If the Baus maliciously debits a colluding account and credits another, at least two account balances are compromised yielding a probability, $p\,(m,n,t)$ of identifying at least one false merkle leaf, where t is the number of sidechain blocks allowed for a challenge period:

$$p\,(m,n,t)\ \geq\ 1-\left(\frac{(n-m)\times(n-m-1)}{n\times(n-1)}\right)^{t}$$

$(refer\ to\ \ 7.1.\ Proof\ in\ Appendix)$

The Baus may select m, n and t to its discretion, yet is incentivized to maximize $p\,(m,n,t)$ to give traders confidence their funds are safe. Given the Baus' $p\,(m,n,t)$ and the $\#tradesofuser$ during a particular sidechain block, the $proof\ size$ and net $proof\ cost$ to publish on the root blockchain is calculated as:

$$proof\ size\ = m[MHlog_2(n) + AB + AA + AI + PB + UAB + (TH + TB) \times (\#tradesofuser)]$$
$$proof\ cost\ =\ proof\ size\ \times (gas\ price\ per\ byte)$$

(*refer to* 7.4. *Data Size in Appendix for abbreviation names and byte sizes*)

The *merkle size* and *merkle cost* to publish on the root blockchain are calculated as:

$$merkle\ size\ =\ (MR + CB + TD + TW + BAB + TC + TV + HV)$$
$$merkle\ cost\ =\ merkle\ size\ \times (gas\ price\ per\ byte)$$

The Baus may choose to charge either percentage or flat fees based on a function, δ, of the root blockchain *gas price*, the trading activity during that block, and the amount of live accounts in its data structure. Based on fee revenue and the publishing costs on the root blockchain, the Baus generates a function, $\Delta B_{est,\,i}$, to determine the amount of root blockchain blocks between publishing times for i^{th} sidechain block:

$$\Delta B_{est,\,i} \geq \frac{(merkle\ cost + proof\ cost + computation\ cost)}{T_{f_{est,\,i}} \times (avg\ trading\ fee)}$$

where,
$$T_{f_{est,\,i}} = (\alpha \sum_{k=i-N}^{k=i-1} (1-\alpha)^{i-k-1} T_{f_k})$$

where, $T_{f_k} = \frac{total\ trades}{total\ root\ blockchain\ blocks}$ *in* k^{th} *sidechain block and* α *is trade frequency estimator*

(*see proof in Appendix* 7.2.)

Given that there is $i = 1$ malicious account publishing in the FSB, it is derived that:

$$\frac{m}{n} \leq 1 - (1-p)^{\frac{1}{t}}$$

This determines the number of m merkle proofs necessary to identify this malicious publishing with probability p (*see graph in Appendix*).

Given proven malicious publishing of state changes or withholding of data to the network, the Baus forfeits all of its stake to the challenging Watcher. All following blocks including the malicious one are discarded. User account balances revert to the last checkpoint, defined as the user account balances at the point of expiration of the current challenge period. Funds will be withdrawable given an adequate proof of ownership submission to the root blockchain.

4.2. Watchers

Watchers are incentivized to maintain their own image of the Finality Sidechain because they earn fees for providing instant access to funds on the root blockchain for user withdrawals. To provide a user instant root blockchain liquidity during a withdrawal challenge period with near zero risk, a Watcher confirms the validity of the request by examining its own image of the user's trade and account balance history. When the Watcher confirms the data matches, it sends a transaction of the requested funds minus a fee to the user's address and inherits the non-fungible token representing the user's withdrawal from the FSC. These funds are unlocked and transmitted to the Watcher's root blockchain address after the challenge period is over in t FSBs from the $block\_number$ of the request. Because Watchers already maintain a highly detailed data structure of the Finality Sidechain to provide instant withdrawal liquidity, they also are positioned to identify potential Baus malfeasance to win the bounty of the Baus' stake given malicious publishing.

4.3. Users

While the Baus publishes randomized merkle proofs to the root blockchain to ensure data availability and Watchers scrutinize the Baus' Finality Sidechain to ensure probabilistic security, ordinary users may also maintain their own account merkle branch to validate their trades and account balances personally. This will increase the probability that users challenge any malicious publishing of their trade history and account balance by the Baus. Users may come online once per t sidechain blocks to ensure the validity of the Finality Sidechain holds.

4.4. Finality (FNY) Cryptocurrency

The Finality (FNY) Cryptocurrency is initially built on the ERC20 standard and may be used as one of the trading pairs of a Finality exchange. Finality will distribute 100,000,000,000 FNY to the community in a Massive Randomized Airdrop (M-RAD). In the future, the ERC-20 FNY will be convertible 1:1 with a new FNY cryptocurrency that will be usable as a gas currency to operate and publish on the Finality Root Blockchain.

5. Root Blockchain

In the future, the Finality Protocol will move to an interoperable Finality Root Blockchain that optimizes for the publishing of stage changes for decentralized exchanges. Bauses use the same second layer protocol outlined in this paper to publish FSBs, but will be committing their merkle proof along with FSB data to the Finality Root Blockchain using FNY as a gas currency and base trading pair. This root blockchain will be secured by a proof of work consensus layer incentivizing miners paid in FNY for their work, and will trustlessly interact with FSCs built on neighboring blockchains through Simplified Payment Verification (SPV) integration.

5.1. Simplified Payment Verification (SPV)

Simplified Payment Verifications (SPVs) provide a means of confirming the validity of transactions on any blockchain without the necessity of running a full node. Each FSC

built on a neighboring smart contract enabled blockchain contains an SPV that confirms deposits to credit Finality Sidechain user accounts. This enables a Baus to provide interoperable trading pairs on its Finality Sidechain, and users may deposit into a linked FSC on a neighboring root blockchain to trade on its Finality Sidechain.

5.2. Sidechain Auction Applications

In addition to the decentralized atomic exchange of fungible digital assets, the Finality Root Blockchain may support the publishing of Finality sidechains specialized for scalable auctioning, including any digital ad space bidding or non-fungible token bidding. The same type of publishing would occur upon the root blockchain as outlined in this paper, but the exchange would be an amount of a fungible digital asset in return for a non-fungible digital asset.

6. Security / Attack Vectors

Users manage the private keys associated with their deposit addresses and may withdrawal from Finality at any time with adequate proof of ownership submission, eliminating the risk of a hacker obtaining access to an exchange wallet. Although they may pose a threat to other second layer protocols, the following attack vectors are mitigated with our Finality Sidechain design:

6.1. Sidechain Monitoring

One of the paramount risks associated with off-chain protocols is the requirement of users to come online at least once per t sidechain blocks challenge period in order to protect their account from malicious publishing of the operating entity. This requirement is not necessary with Finality because the Baus must submit the randomized merkle proofs of m of n sidechain blocks from the prior submission for every new on-chain state change, which will with almost certain probability identify any malicious publishing from the prior submissions.

6.2. Root Blockchain Congestion

In the scenario of a congested root blockchain yielding high gas transaction fees, the Baus will decrease the frequency of root chain publishing based on its $\Delta B\ (merkle\ cost,\ proof\ cost,\ avg\ trading\ fee)$ function. This results in the chellege period of t FSBs to be longer in terms of root blockchain block. Users still are able to withdrawal their funds securely.

6.3. Collusion

A FSB producing Baus may attempt to collude with a user account by matching an invalid trade between two accounts, which steals funds from one of the accounts. Given the probability, $p \geq 1 - (\frac{(n-m)\times(n-m-1)}{n\times(n-1)})^t$, of randomly identifying at least one of the two compromised user account balances, the Baus has an almost certain probability of identifying the malicious activity, which would result in the forfeit of its entire stake.

Furthermore, The random publishing of merkle proofs to the root blockchain and the incentive for Watchers to scrutinize the integrity of the Finality Sidechain ensures that user funds are safe from Baus collusion with a user account.

6.4. Data Availability

The availability of sidechain data is necessary in order for third parties to challenge and user funds to remain safe. If the data is unavailable for some trade or account balance, a Watcher can request the data to be revealed on the root blockchain [7]. The Baus must broadcast the requested information on the root blockchain and close the reveal request in the next block submission. The following checklist ensures that data is made available:

1. The Baus broadcasts the events on every trade at no cost, but the Baus has the power to make data unavailable.
2. The Baus provides APIs to retrieve the data at no cost, but the Baus has the power to make data unavailable.
3. A Watcher request peers to share the data at no cost but works only if the data is available with the peers.
4. If the Baus withholds data, a Watcher may request for the data to be revealed by the Baus on the root blockchain. This costs gas to both the Baus and Watcher, but data availability is guaranteed because the FSC requires the Baus to include it on the next publishing.

6.5. Sybil Attack

The Baus may attack the network by creating a large number of accounts in order to increase the n value of account to potentially decrease the probability of a malicious account publishing is detected by the random proof. Since $p\ (m,\ n,\ t)\ \geq\ 1-(\frac{(n-m)\times(n-m-1)}{n\times(n-1)})^t$ is a constant value this risk is mitigated because the Baus will be required to submit more proofs per publishing. Additionally, for the Baus or any attacking party to create a large number of new accounts, on-chain transactions are required to initialize any new accounts, making this attack vector unfavorable.

7. Appendix

7.1. Proof - $\mathbb{P}(m,n,t)$

$$\because\ \mathbb{P}(m,\ n,\ t) = \sum_{\gamma=1}^{\gamma=n-m} \frac{{}^{n-\gamma}C_m}{{}^{n}C_m} \times \mathbb{P}(mal=\gamma)$$

$$= Probability\ of\ BAUS\ getting\ away\ with\ at\ least\ 1\ and\ at\ most\ \gamma\ malicious\ accounts$$

$$where\ \mathbb{P}(mal=\gamma) = Probability\ of\ \gamma\ accounts\ being\ compromised\ by\ BAUS$$

$$also,\ \mathbb{P}\ (m,\ n,\ \gamma)\ = \frac{{}^{n-\gamma}C_m}{{}^{n}C_m}$$

$= \textit{Probability of BAUS getting away with } \gamma \textit{ compromised accounts}$

$\therefore \mathbb{P}(m,\ n,\ \gamma,\ t) = \left(\frac{{}^{n-\gamma}C_m}{{}^{n}C_m}\right)^t$

$= \textit{Probability of BAUS getting away in t consecutive blocks with } \gamma \textit{ compromised accounts}$

$\therefore \textit{Probability of BAUS getting caught} = P(m,\ n,\ \gamma,\ t) = 1 - \left(\frac{{}^{n-\gamma}C_m}{{}^{n}C_m}\right)^t$

$P(m,\ n,\ 1,\ t) = 1 - \left(\frac{{}^{n-1}C_m}{{}^{n}C_m}\right)^t$

$P(m,\ n,\ 1,\ t) = 1 - \left(1 - \frac{m}{n}\right)^t$ and $P(m,\ n,\ 2,\ t) = 1 - \left(\frac{(n-m)\times(n-m-1)}{n\times(n-1)}\right)^t$

7.2. Proof - ΔB

$\because \textit{Profit}_{\beta_i} = \textit{avg trading fee} \times \textit{total trades}_{\beta_i} - (\textit{proof cost} + \textit{merkle cost} + \textit{computation cost})_{\beta_i}$

where $\textit{proof cost} = \textit{proof size} \times (\textit{gas price per byte})$
$\textit{proof size} = m[MH \cdot \log_2(n) + AB + AA + AI + PB + UAB + (TH + TB) \times (\#\textit{tradesofuser})]$
$\textit{merkle cost} = \textit{merkle size} \times (\textit{gas price per byte})$
$\textit{merkle size} = (MR + CB + TD + TW + BAB + TC + TV + HV)$

$\therefore \textit{total trades}_{\beta_i} = (\textit{Profit}_{\beta_i} + (\textit{proof cost} + \textit{merkle cost} + \textit{computation cost})_{\beta_i}) \,/\, \textit{avg trading fee}$

$\textit{also},\ T_{f_i} = \frac{\textit{total trades}_{\beta_i}}{\Delta B_i}$

where $T_{f_i} = \textit{trade frequency in block } \beta_i$
$\Delta B_i = \textit{root blockchain block at the time of } \beta_i - \textit{root blockchain block at the time of } \beta_{i-1}$
$\beta_i = \textit{sidechain block}$

$\therefore \Delta B_i = \frac{\textit{total trades}_{\beta_i}}{T_{f_i}}$

$\Delta B_{est,\,i} = \frac{\textit{total trades}_{\beta_i}}{T_{f_{est,\,i}}}$

where $\Delta B_{est,\,i} = \textit{estimated value of } \Delta B \textit{ for } \beta_i$
$T_{f_{est,\,i}} = \textit{estimated value of } T_f \textit{ for } \beta_i$

The estimated T_f *can be deduced from exponential weighted average*

$$T_{f_{est,\,i}} = \alpha \cdot T_{f_{i-1}} + (1-\alpha) \cdot T_{f_{est,\,i-1}}$$

where α *is trade frequency estimator. Higher* α *discounts older frequencies faster.*

$$\therefore T_{f_{est,\,i}} = \alpha\,[T_{f_{i-1}} + (1-\alpha)T_{f_{i-2}} + (1-\alpha)^2 T_{f_{i-3}} \cdots + (1-\alpha)^{N-1} T_{f_{i-N}}]$$

$$T_{f_{est,\,i}} = \alpha \sum_{k=i-N}^{k=i-1} (1-\alpha)^{i-k-1} T_{f_k}$$

$$\Delta B_{est,\,i} = \frac{total\ trades_{\beta_i}}{\alpha \sum_{k=i-N}^{k=i-1} (1-\alpha)^{i-k-1} T_{f_k}}$$

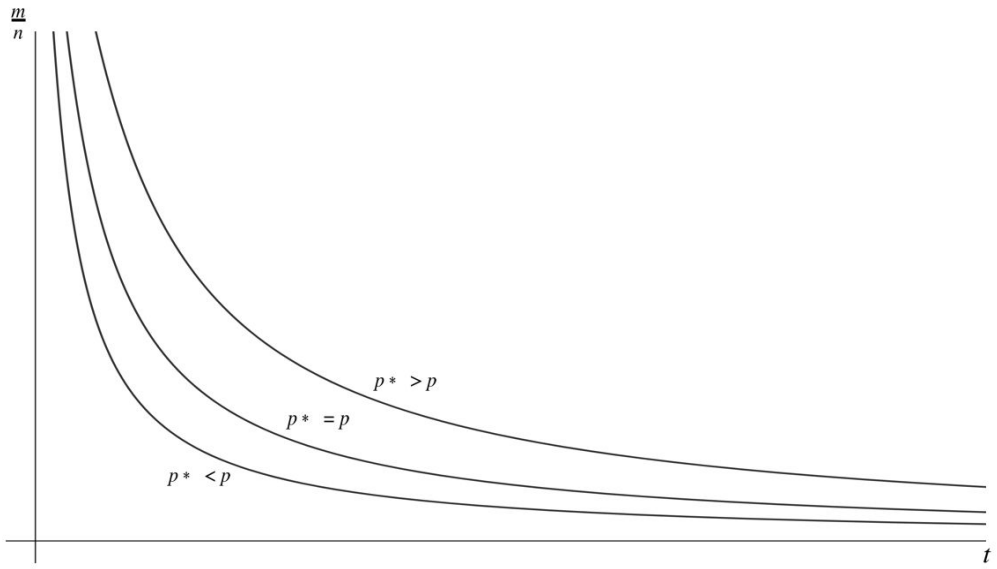
$$\Delta B_{est,\,i} = \frac{Profit_{\beta_i} + (proof\ cost + merkle\ cost + computation\ cost)_{\beta_i}}{(\alpha \sum_{k=i-N}^{k=i-1} (1-\alpha)^{i-k-1} T_{f_k}) \times avg\ trading\ fee}$$

$$\therefore\ for\ Profit_{\beta_i} \geq 0$$

$$\Delta B_{est,\,i} \geq \frac{(proof\ cost + merkle\ cost + computation\ cost)_{\beta_i}}{(\alpha \sum_{k=i-N}^{k=i-1} (1-\alpha)^{i-k-1} T_{f_k}) \times avg\ trading\ fee}$$

7.3. Computation

For $\gamma = 1$, we may use $\frac{m}{n} \leq 1-(1-p)^{\frac{1}{t}}$ to visualize the relationship between t and $\frac{m}{n}$ given various probabilities, $p* = p$, $p* > p$, and $p* < p$:



Let's analyze an example Baus with:

$n = 1,000,000$ *live accounts*
$m = 1,500$ *merkle leaf proofs*
$t = 6,000$ *FSB challenge period*
50 *coins*
100 *trades per active user*
$100,000$ *total trades*

A malicious account update published by a Baus would have at most a 1 in 66,553,888, or a 0.000001503% chance of successfully cheating the network.

7.4. Data Size

Abbreviation	Name	Data Type	Size (bytes)
MR	Merkle Root	*bytes32*	32
CB	Current Block	*uint64*	8
TD	Total Deposits	*decimal [ ]*	$16 \times (\#coins)$
TW	Total Withdrawals	*decimal [ ]*	$16 \times (\#coins)$
BAB	Baus Account Balance	*decimal [ ]*	$16 \times (\#coins)$
TC	Trades Count	*uint64*	8
TV	Trade Volume	*decimal [ ]*	$16 \times (\#coins)$
HV	Hibernate Volume	*decimal [ ]*	$16 \times (\#coins)$
MH	Merkle Hash	*bytes32*	32
AB	Account Balance	*decimal [ ]*	$16 \times (\#coins)$
AA	Account Address	*address*	20
AI	Account ID	*uint64*	8
PB	Previous Block	*uint64*	8
UAB	Updated Account Block	*uint64*	8
TH	Trade Hash	*bytes32*	32
TB	Trade Block	*bytes [ ]*	200

7.5. Terms

Abbreviation	Name
FSB	Finality Sidechain Block
FSC	Finality Smart Contract
TTB	Trade Tree Block
ABB	Account Balance Block
M-RAD	Massive Randomized Airdrop

8. References

[1] Satoshi Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” https://bitcoin.org/bitcoin.pdf, 2008

[2] Will Warren, Amir Bandeali, “0x: An open protocol for decentralized exchange on the Ethereum blockchain,” https://0xproject.com/pdfs/0x_white_paper.pdf, 2017

[3] Aurora Labs, “IDEX: A Real-Time and High-Throughput Ethereum Smart Contract Exchange,” https://idex.market/static/IDEX-Whitepaper-V0.7.5.pdf, 2017

[4] Vitalik Buterin, “Minimal Viable Plasma,” https://ethresear.ch/t/minimal-viable-plasma/426, 2018

[5] pdobacz, “More Viable Plasma,”https://github.com/omisego/elixir-omg/blob/develop/docs/morevp.md, 2018

[6] Paul Peregud, “Tesuji Plasma Blockchain Design,”https://github.com/omisego/elixir-omg/blob/develop/docs/tesuji_blockchain_design.md, 2018

[7] Rami Khalil, Arthur Gervais, “NOCUST – A Non-Custodial 2nd-Layer Financial Intermediary,” https://eprint.iacr.org/2018/642.pdf, 2018

[8] Jieyi Long, “Off-chain Plasma state validation with on-chain smart contract,” https://ethresear.ch/t/off-chain-plasma-state-validation-with-on-chain-smart-contract/2813, 2018