Saito: Solving the Collective Action Problems in Blockchain Design

David Lancashire, Richard Parris, Stephen Peterkins

January 10, 2020
v. 3.0.4

Abstract

Saito is a blockchain designed to solve the collective action problems that impede scaling in all proof-of-work and proof-of-stake blockchains. This whitepaper outlines the solution: coupling a circular ledger to a consensus mechanism that pays nodes for the collection and sharing of transaction fees. Fixing the underlying economic issues pushes Saito towards an optimal network structure while eliminating sybil-attacks, fee-recycling attacks, block-withholding attacks and the fifty-one percent attack.

Saito is a cryptocurrency designed for applications that need to send large amounts of data across a PKI network. It can be used to build decentralized versions of many data-heavy services, such as un-astroturfable Internet forums, social networks, pay-to-play websites, distributed key registries that are secure from MITM attacks, payment channels, and much more.

More generally, Saito is a solution to the problem of how to build a blockchain that preserves trustlessness at scale. The design corrects the collective action problems inherent in proof-of-work and proof-of-stake networks, permitting scalability to the point that underlying network hardware rather than economic constraints impose limits on blockchain growth. We believe the practical limit for a Saito blockchain today is in the order of 100 TB of data per day, and advances in routing capacity will push us to the petabyte level within a decade.

The next section describes briefly the economic problems that need to be solved in order to build a scalable blockchain. The following sections outlines how Saito solves these problems and describes an implementation of these methods.

1. THE PROBLEM

The problem with blockchain scaling is not at the network technology layer: at the time of writing, data centers around the world are implementing 400 Gbps network switches while 100 Gbps connections are becoming standard even in lower-tier colocation facilities. If we had the resources to pay for the necessary equipment there is nothing technically stopping us from building a blockchain that is as decentralized as the public Internet backbone.

What limits network growth is precisely the challenge of paying for the network. In the past, non-economists have waved away this limitation, claiming that as long as *someone* is earning money from the network they will pay all costs necessary to support it. But this is not true, for proof-of-work and proof-of-stake networks are affected by two fundamental market failures: a tragedy-of-the-commons problem that leads to blockchain collapse and a free-rider problem that leads to a systemic under-provision of unpaid network services. Neither problem is crippling at small scale, but they both grow increasingly incapacitating as bandwidth and storage costs rise.

The tragedy-of-the-commons issue is created by the existence of the permanent ledger, which encourages nodes to accept payment today for work that can be offloaded to others tomorrow. This incentive leads to bloated blockchains and more subtly to transaction mis-pricing, as users can pay fees that do not reflect the true cost of their transaction to the network. The fact that this is a fundamental problem is self-evident from the way Satoshi's solution is "not to care," an approach that stops being viable in networks that operate at serious economic scale.

Eliminating the tragedy-of-the-commons problem requires all nodes that add transactions to the blockchain bear the cost of processing those transactions for as long as they remain on the blockchain. In practice, this requires a market mechanism for accurately determining the price of on-chain data storage. It also requires eliminating blockchain creep or deferring fee collection so that payments are meted-out over time as the node continues to do the work required for payment. Our solution to this problem is described in Section 2.

The free-rider problem is more insidious. It arises in blockchains where payments are made for one specific type of work (such as mining or staking) at the expense of other necessary activities. This incentivizes participants to maximize their spending on paid work and minimize spending on anything else. In the blockchain space, this results in miners and stakers "free-riding" on those who do the unpaid work of collecting

fees or developing applications or otherwise supporting the user-facing network. Evolutionary pressures make this trap inescapable: any Bitcoin miner that spends a larger percent of its revenue on hashing than its more altruistic peers will eat away at their market share until they also defect.

In economics, the typical solution to the free-rider problem is to eliminate the property of "non-excludability" associated with any good or service: restricting its benefits to those who pay the costs of provision. In the blockchain space this is impossible to do without destroying the openness of the network: incentivizing network nodes to monetize transaction flows and block producers to hoard fees. Without a solution to this problem our choice is between a network that cannot scale because it cannot pay for network operations, or a network that scales but loses the openness, trustlessness and economic self-sufficiency that makes the blockchain a useful invention.

One solution to this problem is fixing the underlying incentive structure so that participants are incentivized to provide what the network actually needs. Because the blockchain requires a quantifiable cost-of-attack, this requires eliminating "mining" and "staking" and shifting to a different form of work that measures and pays nodes in proportion to the "value" they provide the network instead of the amount of hashing or staking they do.

This requires us to find a new way to measure value to pay nodes in proportion to the amount they provide to the network. We propose deriving our measure of "work" from the transaction fees that users pay. The work of routing transaction fees into the network is the work that our network must encourage. Honest nodes can then be induced to do this work profitably. Attackers can be put into a situation where attacking the network has a quantifiable cost of attack. The security mechanism described in Section 3 outlines a technical method of implementing such a system.

2. FIXING THE TRAGEDY OF THE COMMONS

Saito solves blockchain creep by allowing the nodes in the network to delete the oldest blocks in the ledger at predictable intervals ("epochs"). Epoch length is specified in the consensus code. An extreme case - a blockchain designed to handle global email traffic - may have an epoch as short as 24 hours.

Saito specifies that once a block falls out of the current epoch, its unspent transaction outputs (UTXO) are no longer spendable. But any UTXO from that set which contains enough tokens to pay a rebroadcasting fee must be reincluded in the very next block. Block producers do this by creating special "automatic transaction rebroadcasting" (ATR) transactions that include the original transaction data, but have entirely fresh and thus newly-spendable UTXO. After two epochs block producers may delete all block data, although the 32-byte header hash may be retained to prove the connection with the original genesis block.

The ATR mechanism fixes the tragedy of the commons problem completely, while making it impossible for a blockchain to grow so big that it collapses. The key is ensuring the "rebroadcasting fee" paid by ATR transactions is a positive multiple of the average fee paid by new transactions. As the blockchain grows and there is less space for new transactions in each block, market competition pushes up the fees paid by new transactions, forcing up the fees paid by older transactions in the process. The market reaches equilibrium where old data is removed from the chain at the same pace that new data is added.

Market discovery of the true cost of blockchain processing is a side-effect of this incentive structure, which works by eliminating the incentive block producers have to add unprofitable data to the chain. This mechanism fixes the tragedy-of-the-commons problem at the incentive level, and prevents subtle forms of free-riding commonly found in other chains (deleting on-chain data, refusing to store or share historical blocks). These problems are eliminated because nodes that do not store the whole blockchain are incapable of producing new blocks, and old data must be rebroadcast for payments to be issued.

While this avoids the problem of our blockchain growing too large for network nodes to store, and ensures space on the blockchain can be priced accurately even as storage times approach infinity, fixing the tragedy of the commons problem does not get money to the nodes in the peer-to-peer network that are paying for all of the miscellaneous activities that keep the network operating. To solve this problem we need a new consensus mechanism.

3. ELIMINATING FREE-RIDING

In Saito any node can create a block at any time provided it includes enough "routing work". The amount of "routing work" required to produce a block depends on how quickly a block follows its predecessor: consensus rules increase the value immediately after a block is found and decrease it gradually until it reaches zero. Since block producers will issue blocks as soon as it becomes profitable, the pace of block production is determined by the overall amount of "routing work" generated by the network.

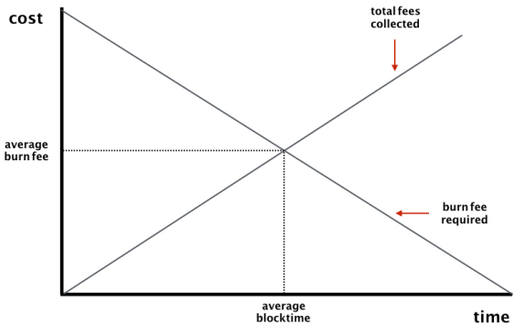

Figure 1: The Burn Fee Curve

Saito derives routing work from the value of the transaction fee paid by each transaction. Using this measure of work to produce blocks makes attacking the network expensive, since any increase in the pace of block production requires attackers to pay more in transaction fees than the network is actually collecting. It can be seen from Figure 2 that it is impossible for attackers to produce blocks at a faster pace than the main chain unless they have access to a larger pool of transaction fees.

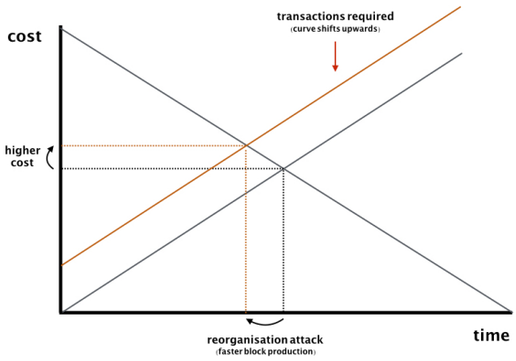

Figure 2: Good Actor Burn Fee Costs...

To prevent attackers from stealing routing work, Saito has its routing nodes cryptographically sign transactions as they propagate inwards towards block producers. Our consensus rules specify that the amount of "routing work" a transaction provides drops with each hop that the transaction takes across the network, and that transactions provide zero routing work to nodes that are not included in this routing path. The "work" the network is measuring is essentially the efficient collection and sharing of inbound network fees.

We note that as long as there is no payment for block production this system offers comparable security to Bitcoin: the cost of a chain-reorganization can always be quantified. Attackers need to spend their own money to attack the chain, allowing users who require significant guarantees against non-reversibility to wait an appropriate number of confirmations. As a bonus, the network can automatically regulate the amount of "routing work" needed for block production to keep blocktime constant as transaction volume grows, so that security scales with fee-volume.

The major problem with this approach lies in the consequences of requiring the network to burn capital to produce blocks:

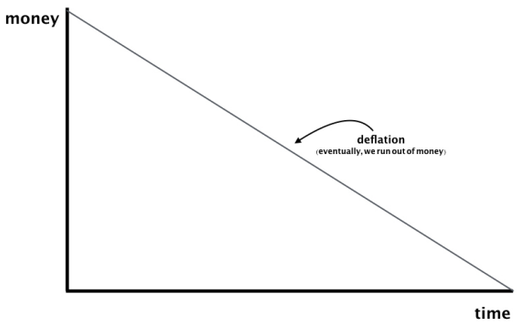

Figure 3: Deflation of Burn Fee Over Time

Avoiding a deflationary crash requires us to inject tokens back into our network, without making attacks profitable. Saito cannot follow in the footsteps of Bitcoin, and simply give the fees directly to block producers: this would destroy our ability to derive "routing work" from fees since attackers could simply recycle the payment from one block into the routing work needed produce the next! Dividing the payment is a better solution, but as long as block-producers have any influence over who gets paid a savvy attacker could sybil the network or conduct grinding attacks that target the token-issuing mechanism.

So, the problem that Saito solves is different from proof-of-work and proof-of-stake blockchains. In Bitcoin-class networks, security is achieved by making it difficult to produce blocks, and the network gives control over fee distribution to the block producer.[1] In Saito networks our first-order problem changes into securing the payment mechanism: ensuring that payments are proportional to work *independently of who produces the actual blocks*. Our goal is creating a mechanism where honest nodes are paid for operating the network regardless of whether they produce bocks. And where attackers always lose money attacking the network because they must necessarily transfer money to honest nodes as a condition of producing the longest-chain.

Ensuring that there is always a quantifiable cost to attacking the system involves returning transactions fees to the network through a process that cannot be gamed by any of the players in the network without spending far more money on the attack than they stand to benefit from collecting the payments. We call this method the "golden ticket" system.

4. THE GOLDEN TICKET

[1]Many core problems with the proof-of-work and proof-of-stake mechanisms stem from this decision. Leaving aside the fifty-one percent attack, note the way these networks use supply-side constraints in external markets (i.e. inelastic supply curve for hashpower or capital) to impose a "cost constraint" on attackers. Not only does this design remove any ability for the blockchain to regulate its own security, but the profits generated from sourcing work necessarily and inevitably commoditize and flatten the supply-curve for the work-factor in the external market.

Whenever a node produces a block, it may collect the difference between the amount of "routing work" (transaction fees) included in the block and the amount of routing work required for block production. No other payments are made.

Unlocking the payments requires the network to solve a computational puzzle we call the "golden ticket". This puzzle requires knowledge of the block hash to solve. Miners on the network listen for blocks as they are produced and begin hashing in search of a solution and propagating those solutions back into the network as fee-paying transactions.

We specify that only one solution may be included in any block, and the solution must be included in the very next block to be considered valid. If these conditions are violated or if a "golden ticket" is not solved, the funds that were not paid out in the previous block are not allocated. They fall backwards into the blockchain and eventually fall off the circular blockchain, at which point the lost funds will be recollected and eventually included in a future block reward.

Should a solution be found in time, the unallocated fees are released to the network; split between the miner that found the solution and a random node in the peer-to-peer routing network. The winning routing node is selected using a random variable sourced from the miner solution, with each routing node's chance of winning normalized to be proportional to the amount of "routing work" it contributed to the block being solved.

We call the division of payment between miners and routers the "paysplit" of the network. It is set to 0.5 by default (half to miners, half to routers) but can be made adjustable as described in the section below. The golden ticket system can be visualized as follows:

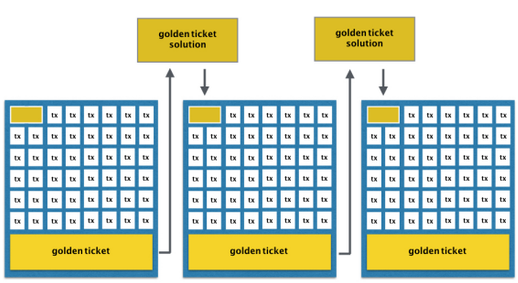

Figure 4: The Golden Ticket System

This system has several major advantages over proof-of-work and proof-of-stake mechanisms. The most important is that Saito explicitly distributes fees to the nodes that service users, collect transactions and produce blocks, and does so in proportion to the amount of value that these actors provide to the overall network. Network nodes compete for access to lucrative inbound transaction flow, and will happily fund whatever development activities are needed to get users on the network.

This is a fundamental shift. Whereas other blockchains explicitly define which activities have value, Saito lets the users on the edge of the network decide what constitutes value, and infers who deserves payment from their price-signaling. The Saito method incentivizes the efficient delivery of value to users. It is also the only way to guarantee that a self-sufficient network can remain open at scale, since paying for fee-collection is the only way to ensure the economic independence of the consensus mechanism.

The Saito consensus mechanism is also "twice as secure" as its proof-of-work and proof-of-stake counterpoints. Honest nodes route transactions to block producers and earn fees in exchange. But attackers are thrown into a catch-22: they not to spend the same amount of fees as the honest network to produce a chain that can compete with the longest-chain, but must also match 100 percent of mining output to find enough golden ticket solutions to recapture their funds.

The basic version of the Saito system achieves 100 percent fee-security, eliminating the fifty-one percent attack completely. Section 6 describes a modification to this mechanism that pushes security above 100 percent and guarantees that attackers will lose money in all circumstances. Regardless of which implementation is used, the economic problems created by mechanisms that rely on external supply-curves vanish: mining serves as a pure cost function instead of a difficulty function and the blockchain remains secure even if the supply curve for hashpower becomes perfectly flat.

5. ADVANCED SECURITY - PAYSPLIT

There are several modifications to the paysplit mechanism that can be used to increase attack costs. While the version of Saito being launched for MAINNET does not include this mechanism, it is possible to add a dynamic voting system to Saito that can allow paysplit to float dynamically. This section describes an improvement that allows for a floating paysplit under certain assumptions about the rationality of the network.

An implementation of this system modifies blocks so that they include a vote to increase, decrease or hold constant the paysplit of the network. Golden ticket solutions are then modified so that they contain similar vote on the difficulty of the golden-ticket production function. The consensus variables of the network are updated to reflect both votes when and only when golden tickets are solved and included in the blockchain.

Adjusting paysplit like this can change the distribution of fees between routing nodes and miners in real-time. This allows the network to reach

an optimal equilibrium rather than an arbitrary one. To prevent the resulting equilibrium from reflecting only the preferences of the routing and mining nodes, we recommend letting the users on the network tag their transactions with an optimal paysplit vote as well: should a user-originated transaction contain such a vote, it may only be included in a block that votes in the same direction. Users who take sides in the ongoing struggle between routers and miners thus sacrifice the reliability and speed of transaction confirmation, but gain marginal influence over how the network allocates fees. Equally, users gain power as they can withhold their fees from nodes voting differently to themselves.

Under conditions where network participants exhibit bounded rationality, this mechanism pushes paysplit to the point where the security provided is optimal for all participants given the cost of additional fee collection. Toqueville-compacts secure this equilibrium point: any two players in the tripartite network structure (routers, miners, users) may team-up to shift the paysplit of the network back to the desired ideal. While we leave research into this mechanism for the future, a useful thought experiment is exploring how the security of this three-player system degrades to offer only bitcoin-level security as the paysplit approaches extreme values.

6. ADVANCED SECURITY - POWSPLIT

It is possible to increase attack costs beyond 100 percent of available returns through a "powsplit" mechanism. Note that in the normal Saito implementation with a fixed paysplit, the network auto-adjusts mining difficulty so that one solution is found per block, on average. Since miners cannot control the speed at which solutions are found, network difficulty may end up being lower on average than needed for optimal security.

A "powsplit" approach eliminates this problem by targeting mining difficulty so that one solution is found every N blocks on average. When such a solution is included in the blockchain, if the previous block did not contain a golden ticket, the random variable used to pick the winning routing node for the previous block is hashed again to select a winner from a table of stakers for the unsolved block which preceded it. An upper limit to backwards recursion may be applied for practical purposes, as the circular blockchain will recapture any funds that are not paid out.

To become stakers in the network, users broadcast a transaction containing a specially-formatted UTXO. These UTXO are added to a list of "pending stakers" on their inclusion in a block. Once the current staking table has been fully paid-out, all pending staking UTXO are moved into the current staking table. To avoid throttling attacks on the staking mechanism it is wise to not permit stakers to withdraw or spend their staking UTXO until they have received payment.

The percentage of network revenue allocated to staking nodes should be their proportional share of the amount of fees paid into the treasury by the staking mechanism during the previous round. Limits may be put on the size of the staking pool to induce competition between stakers if desirable or prevent users from spamming the staking mechanism in the hope of dissuading honest stakers from participating. In normal situations the looping blockchain and ATR mechanism will prevent stakers from launching spam attacks as multiple UTXO will all pay rebroadcast fees.

To ensure the system works, block producers who rebroadcast UTXO must now indicate in their ATR transactions whether the specific outputs are active in the current or pending staking pool. A hash representation of the state of the staking table also be included in every block in the form of a commitment to allow nodes to verify the accuracy of the staking table, but the ATR rebroadcast mechanism will theoretically allow all nodes to reconstruct the state of the staking pool within one epoch at most.

The cost of attacking the network now rises above 100 percent. Losses are guaranteed to attackers by adjusting mining difficulty upwards if two blocks containing golden tickets are found in a row and downwards if two blocks without golden tickets are found consecutively. A similar punitive cost throttles the staking payout if two blocks without golden tickets are found in a row (an ever-increasing amount of the staking revenue is withheld). We encourage those interested in how this affects attack costs to consult our mathematical papers on the mechanism.

7. ADDITIONAL NOTES ON NETWORK SECURITY

Saito's design incidentally solves several long-standing problems of note. Hoarding attacks are minimized because nodes that participate in transaction routing maximize revenue by finding the most efficient routing path into the network. Competition encourages the sharing of fees rather than the hoarding of fees. The availability of routing information in blocks also allows participants to check that their peers are faithfully propagating their transactions.

Given that adding hops to any routing path necessarily reduces the profitability of every other node in the path, the sybil problem is also solved. Blocks provide the information needed for participants to identify and eliminate sybils in their peer-to-peer networks. Evolutionary pressures ensure that they do: weaker nodes which permit themselves to be sybilled and refuse to opti-

mize their transaction paths will find themselves driven out of the network by competitive pressures over time.

The routing network also serves a unique defensive mechanism. As routing nodes can raise the cost of attacks to attackers in real-time by refusing to route transactions to them, forcing an increased reliance by the attacker on their own wallet to fund block production. Because nodes must participate in the P2P network to harvest transactions also defends Saito against subtle attacks like monetized transaction flows and closed-access routing networks which benefit their participants in other networks while undermining the profitability of nodes in the peer-to-peer network.

As a final observation, we note that the "scalability trilemma" does not exist in the Saito design, something that is self-evident given the multiple obvious configurations of the network in which redirecting fees from miners to routing nodes may simultaneously increase the throughput, decentralization and security of the network.

7. SUMMARY

The fundamental problems affecting blockchain scaling are economic. Saito fixes them, allowing us to build a massively-scalable blockchain which achieves its scalability not through algorithmic tweaks to existing technologies, but by eliminating the incentive problems that prevent money from flowing to the bottlenecks in the network.

Those who pour over the technical details of our network will find embedded in it at least seven major innovations in blockchain technology: the transient blockchain, the burn fee, the golden ticket system, paysplit and powsplit, dynamic work penalties with history tracking, a secure multi-party voting mechanism, and the chain of cryptographic signatures that permits the blockchain to identify and reward productive nodes in the routing network.

Provisional patent protection has been secured on these techniques but we welcome contact from other blockchain projects looking to incorporate one or several of these methods in their own networks. We encourage readers to visit our website (https://saito.tech) which includes an interface for the working network, a roadmap outlining future development plans, and tutorials that can help anyone get started building Saito applications **today**.