æternity Hyperchains

Recycling power of blockchains

v1.2.0

Grzegorz Uriasz Radosław Rowicki Vjacheslav Brichkovskiy Dimitar Ivanov
Ulf Wiger

June 29, 2023

Abstract

Blockchains rely on decentralized consensus between miners. Several ways of achieving it in a safe, provable, and fair manner have emerged. The oldest approach dating back to Satoshi Nakamoto's whitepaper, known as Proof of Work (PoW) implemented in numerous cryptocurrencies is extremely expensive and leads to severe energy waste. Cheaper algorithms based on Proof of Stake (PoS) promise similar security to PoW, but are usually prone to exploitation. We propose a novel hybrid between PoW and PoS based on BitcoinNG, which combines advantages of both without acquiring their drawbacks.

1 Introduction

One of the most important problems that blockchains need to tackle to maintain a stable and safe infrastructure is the need for a decentralized consensus among the participants. This usually reduces to the existence of a point of agreement, from which everyone should be able to derive an objective and consistent truth about the state of the system. Most commonly, such point defines a certain description of the universe's history, or establishes the person responsible for dictating it for some time. There is a global tendency for depending on some random, fair and unpredictable event that allows for building up a trustless agreement on the current status of events.

Proof of Work (PoW) solutions are traditionally the most popular. They achieve network security by burning (usually empty) CPU cycles in order to randomly hand the power to users depending on their computational effort. This is a slow and costly process, which relies on having a vast and decentralized network of miners. On the other hand, Proof of Stake (PoS) is much more energy-efficient, but its implementation details matter greatly. It could be subject to the nothing-at-stake problem or stake grinding and thus eventually degrade to an inelegant PoW solution.

We propose a hybrid approach called hyperchains: PoS systems, which rely on existing PoW networks for providing security. The security of the PoW network itself is outside the scope of this document: it is up to the specific hyperchain setup to choose a secure PoW network since it can never be as secure as the PoW chain. It is worth noting that it ought to be possible to change the PoW system used by the hyperchain further down in its lifetime. From now on, we will refer to the PoW chain as a parent chain and the PoS-like system—as a child chain.

2 Existing solutions

In this section, we describe the existing approaches to the problem along with the problems they face and how they attempt to address them.

2.1 Proof of Work

Proof of Work (PoW) addresses the problem of decision-making by forcing the users (here called miners) to solve some hard computational puzzle to validate (here, mine) blocks[12]. The point is to make it hard to dominate the network by a single selfish entity. This approach works as long as nobody holds over 50% of the whole computational power, in which case they could fork the chain at any point and get ahead of the main history line. This is a serious issue since in most protocols the most difficult chain is considered the proper one. Therefore, one needs a lot of participants in the network to make it reasonably safe. Moreover, this approach leads to extreme waste of energy and huge costs—according to some measurements, the whole blockchain environment burns enough energy to power entire Denmark[10].

This idea does not scale well—it is almost impossible to create a public network from scratch that would not eventually be dominated by some malicious entity. A lot of existing serious blockchains suffer this problem[5]. On the other hand, a network becomes extremely secure once it is popular enough.

2.2 Proof of Stake

While PoW distributes the leadership based on computational power, PoS does it based on so-called stake, which in most cases means token supply, sometimes with additional tweaks[9, 1]. The idea is to create a leadership voting system which is activated periodically. Each time an election event occurs, the new leader is randomly selected from the stakeholders (called delegates). The chance of winning an election is proportional to the size of one's stake. This approach does not incur any noticeable energetic overhead and therefore is much more friendly to the environment. It also does not require users to have powerful computers to be able to have some involvement in decision-making.

However, PoS comes with some serious issues. First of all, there is the infamous "nothing at stake"[15] problem, which exploits the lack of any cost of the actual mining. In this case, there is no downside to staking several branches simultaneously in case of a fork. Some protocols address it by introducing a "slashing" mechanism to burn stake of the attacker[6]. However, that does not prevent the attack itself, but only decreases incentives to do it.

Ensuring that the source of entropy is distributed along the chain, makes all elections entirely deterministic and predictable leading to a strategy known commonly as "stake grinding," where the dishonest leader tries to rearrange the transactions to influence the result of the upcoming election.

Next issue is the "long-range attack"[16]. In the very beginning, the stake is scattered among a small group of delegates that together have full control over the chain. After some time, they can cooperate and start a concurrent chain diverging from the main one. This could lead to nasty frauds and would destabilize trust over the chain.

On the other hand, there are multiple approaches to deal with these problems. For instance, to prevent nothing-at-stake, the CASPER protocol introduces a "wrong voting penalty,", which punishes voters who support conflicting forks[3]. However, this solution is backed by a finality gadget, which is located on another blockchain anyway. NXT deals with long-range attack by forcefully finalizing all blocks that are older than 720 generations[14]. While this approach is stable for well established networks with high uptime, it introduces weak subjectivity since one still needs to trust some entity while entering the network for the first time or after a longer downtime. Ouroboros staking system has managed to reach solid security, but at the cost of very high complexity[8].

Although these are only few examples, the overall argument is that classical PoS comes with many problems, which, as they are being solved, eventually introduce new ones. This undermines reliability of many PoS systems, especially compared to mature PoWs.

3 Hyperchains design

We present a hybrid strategy that can benefit from stability of PoW solutions while offering scalability of PoS systems. A *hyperchain*, as originally described by Yanislav Malahov[11], is a special kind of blockchain that sticks to an already existing chain. They are going to be called child and parent chain, respectively.

The parent chain can be almost any arbitrary blockchain. In general, we want to use some big existing PoW based chains (at the time of writing, preferably Bitcoin or Litecoin) to reuse their burned work to

maintain the stability of the child chain. We also propose a PoS-like election system to choose leaders on the hyperchain. In this case, however, we employ a reliable and, most importantly, unpredictable source of randomness, which is the state of the parent chain. The idea is not absolutely novel, as there already exists research into this topic[2].

Following that, it feels natural to start a new election each time a (key)block is mined on the parent chain. The next leader shall be chosen depending on the hash of that block and selected with chances proportional to their stake. The selection algorithm is abstract over this document—it is up to the hyperchain to define the details.

We define a group of leadership candidates called "delegates." Each delegate needs to express their will of participation in the upcoming election by publishing a commitment transaction onto the parent chain. It is important that they clearly declare their view of the child chain and over which block they are going to compete. Therefore, the commitment must consist of:

- The subject of delegation on the child chain (most likely a public key)
- The block hash over which the delegate is going to build
- Signature of the delegate from the child chain

One of the concepts key to the commitment idea is to be able to rely on the parent chain's stability. We want to treat it as a rigid skeleton of the hyperchain, which can be achieved by proper block linking. The elected leader shall be required to publish the key block on the child chain with a cryptographic proof (referencing the parent) of their right to lead the upcoming generation and publish micro blocks.

One dilemma that rises at this point is whether the commitment should reference the latest key block or micro block of the child chain. Referencing a micro block may feel more transparent, but we believe that it would lead to massive forking (especially when some peers would not receive all blocks). One must also consider the parent network's throughput. It may be impossible to fit such a huge amount of commitments if we let delegates compete over each micro block.

The problem with referencing a key block is that the next leader could steal the transactions and post them in their micro blocks. This, however, can be resolved with a smarter feeing strategy. For instance, instead of giving the full fee to the miner, we can split it up and give the bigger part to the next leader who did include the previous leader's micro blocks in their continuation of the history, as it happens in the BitcoinNG[13]. Following their election, the new leader posts a key block on the child chain, which references the commitment on the parent, thus proving their right to lead the generation. Besides that, they need to reference the micro block from the previous generation, on which they want to mine.

4 Election process and staking mechanism

In this section, rather than prescribe specific mechanisms and rules, we propose solutions that would facilitate planning and implementation of the desired algorithms. It is not up to this document to specify the details.

Figure 1: Hyperchains interface

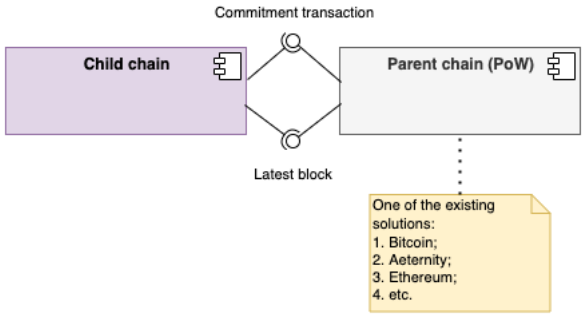
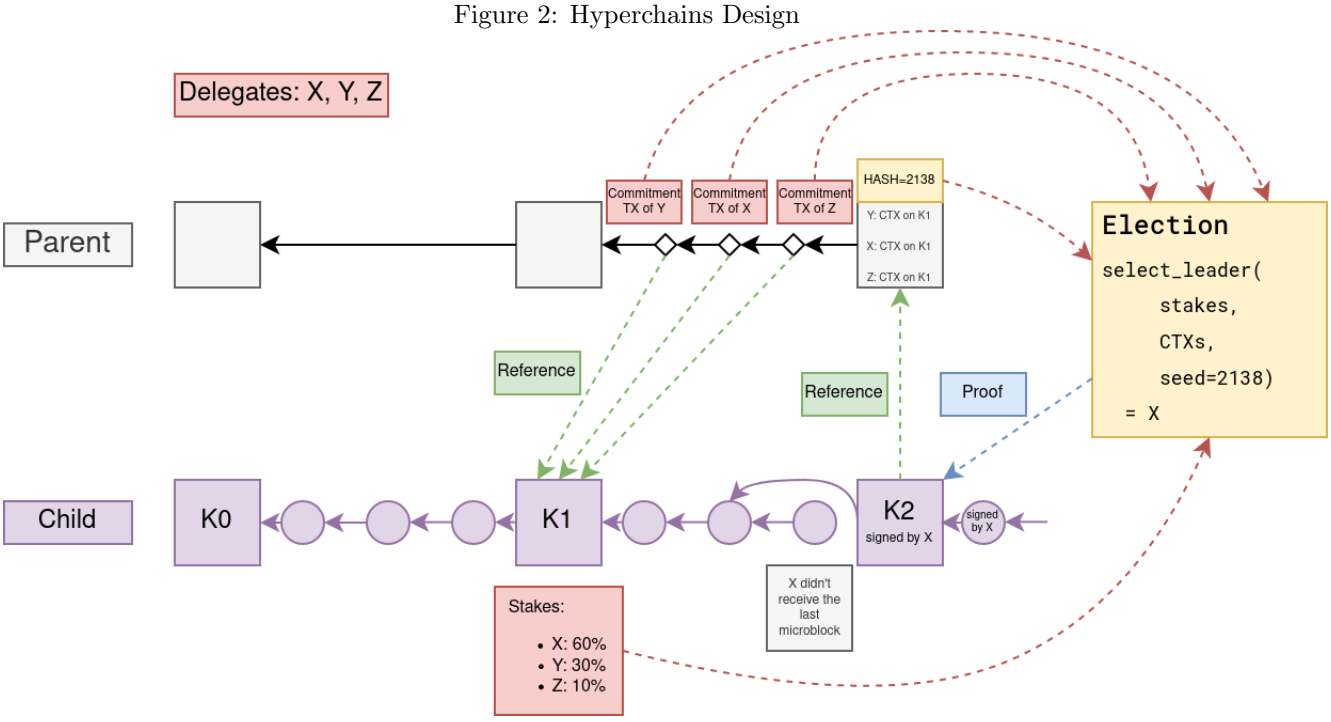


Figure 2: Hyperchains Design



The most convenient way to organize the election process is to create a smart contract on the hyperchain, which would manage stake and determine leaders. This contract shall be referenced in the protocol, and its interface should be specified there. It is advisable that the contract be adjusted by regular calls on the fly—this could prevent some protocol-level hard forking. We highly recommend introducing consensus changes with a decent delay to ensure that it will not break during a true hard fork.

We propose the following features of a staking contract:

- Withdrawing and depositing the stake
- Voting power calculation
- Leader election
- (optional) Controlled consensus changes

The system must implement some mechanisms to prevent abuse of the exposed interface, and it must be resistant to low responsiveness of the generation leader. Therefore, we highly recommend that some necessary calls (like election initiation or reward claiming) be applied automatically in each key block and be protocol-restricted in order to prevent arbitrary calls to them. The contract may optionally provide slashing punishments for misbehaving leaders.

5 Security

This hybrid solution allows us to use the parent chain as a reliable protector against most of the attacks that target the PoS systems[4]. In this section, we assume that the parent chain is well secured, and every user has stable access to it.

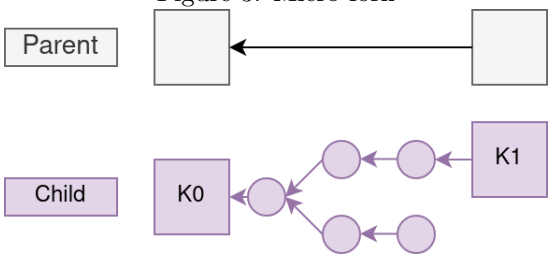
5.1 Nothing at stake

This type of attack splits into two cases: micro forks and generational forks.

5.1.1 Micro forks

A malicious leader may produce blocks without any cost. They are free to create conflicting branches within a single generation. This case is very similar to BitcoinNG's[7]. However, this kind of forking is not dangerous in hyperchains. It may introduce some mess initially but becomes resolved instantly with the next key block.

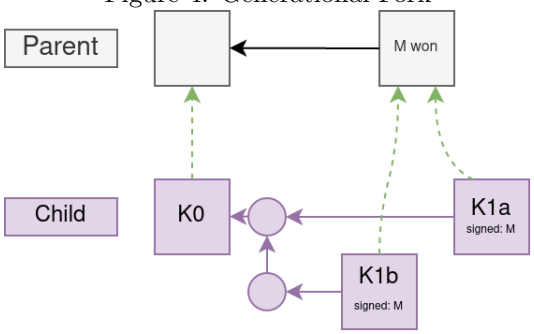
Figure 3: Micro fork



5.1.2 Generational forks

This happens when there are multiple key blocks produced by a leader, most likely as a consequence of a micro fork described above.

Figure 4: Generational Fork



The $K1a$, $K1b$, etc. are key blocks on the child chain emitted by a malicious leader over different micro blocks. This effectively splits the network, as it becomes unclear to which of the forks the delegates should commit.

Similarly to PoW chains, this should be solved by the rest of the network. Eventually some peers should decide on one of the forks, breaking the balance in the weight function proposed below. After that happens the "heavier" chain shall be deemed the right one. To enforce making a decision, for every candidate only a single commitment can be counted (most reasonably, the last one in the parent's block).

A weight function should reflect the total staking power of those who committed to a fork. The exact logic shall be defined in a smart contract. A simple example would be the sum of all stakes that were backed by a commitment.

5.2 Lazy leader

A leader who does not produce a key block is called "lazy". In a case of a lazy leader, any delegate can produce a key block on behalf of them. Subsequently, the network is most likely split in the same way as in a generational fork. Therefore, such a situation shall be resolved the same way. This also means it makes sense to produce key blocks even while not being a leader.

In a case in which a lazy leader submits a delayed key block, it is up to the network whether to consider it. One may argue that a syncing peer would not have a proof that the leader was ignored due to their laziness, but it is not different from blacklisting. This behavior is available in almost every PoW and PoS blockchain and in general does not cause threat, therefore we do not consider it an issue as well.

5.3 Stake grinding

Since the RNG depends exclusively on the key block hash on a PoW chain, it is impossible to predict its outcomes. One could try to mine the parent chain in a special way, but it would require so much computational power that in most cases it would be easier to take control by a 51% attack.

5.4 Long range attack

While it is still possible to perform a long range attack, it would be impossible to do it secretly and without preparation since the very beginning. The commitments guarantee that the information of delegates is stored on an immutable chain, and one would need to announce their will of mining suspicious blocks during the entire period of the attack. This would quickly expose the intention of the attacker and let the others prepare for a possible upcoming attack (by blacklisting them, for example).

5.5 Overloading parent chain

Healthy commitment mechanism is key for fair election on a hyperchain. If some delegates get prevented from publishing commitment transactions on the parent their chances of becoming a leader will vanish regardless of their stake. This is a vulnerable point of the system. On most blockchains there is a limitation on the number/total size of transactions in a single block. A malicious leader could send numerous commitments to fill the entire block leaving no place for the others.

This strategy hands the staking value to parent's tokens in some sense. The difference is that in this case the financial effort made to boost one's chances of being elected is disposable (as long as the attacker did not mine that block—but then they could just reject other commitments). At the time of writing the costs of performing such an attack are huge on most blockchains, and it is almost impossible to sustain it for a longer time. Note that in order to take full control over just a single generation the attacker would need to invest funds equal to significantly extended transaction fee multiplied by the network throughput hoping that every other commitment will get superseded by *every* of their spam transaction. If it is not enough, performing an election every $k_{>1}$th block would greatly amplify the required effort.

5.6 No commitments

It may be the case that no one publishes a commitment transaction to a key block, or that a miner on the parent chain does not include any commitment. We propose that the smart contract responsible for handling elections should decide on what happens in that case. For example, a "lazy leader" scenario may be utilized; the previous leader may take over; or the election may be reevaluated with the candidates from the previous generation.

6 Derived issues

While the presented idea seems to cover the majority of cases, some scenarios may break the system's stability. However, depending on the chosen parent chain and the tweaks of inner protocol, their impact can be limited to an acceptable risk.

6.1 Forks of the parent

Whatever happens to the parent, the similar shall happen to the child. The child chain is entirely vulnerable to forks of the parent chain, and it is quite hard to agree about on which branch to continue. Most likely, the hyperchain would fork as well. On the other hand, if the validators manage to decide on one branch, it would be technically possible to jump into the other if the chosen one becomes less attractive.

6.2 Attacks on the parent

The hyperchain can never be more secure than the parent. If somebody succeeds in a 51% attack on the parent chain, they will also take control of the child chain. Therefore, the choice of the parent should be made with regard to its security.

6.3 Finalization time

Since there is no single correct strategy on how to react to forks, the finalization time shall not be shorter on a hyperchain than on the parent chain. If the parent key block gets rolled back, so will all leader elections.

6.4 Stake collusion

While stake distribution may look healthy on the surface, stakers can actually collude rather than act as self-serving actors. As a consequence, there may emerge a group of delegates whose perceived best interest would be to join a cartel and unfairly influence the blockchain. This issue, however, is common to all capital-based PoS systems where power is handed by the value staked in. Although we do not propose any direct solution to this, a properly tailored staking contract may address this issue.

7 Conclusion

The proposed solution makes it possible to conveniently create simple, customizable, and most importantly, secure blockchains. The idea of reusing the existing mining power of other networks allows not only small chains to stay safe, but also prevents massive energy waste and, therefore, is much more friendly to the environment. Hyperchains can be utilized in extremely flexible ways. They can be created and maintained almost for free and in case of network overload, they can serve as parent chains for other nested child chains. We leave many implementation details up to the hyperchains creators, because we find them highly use-case dependent.

References

[1] Bentov, I., Gabizon, A., and Mizrahi, A. Cryptocurrencies without proof of work.

[2] Bonneau, J., Clark, J., and Goldfeder, S. On bitcoin as a public randomness source.

[3] Buterin, V., and Griffith, V. Casper the friendly finality gadget.

[4] DEIRMENTZOGLOU, E., PAPAKYRIAKOPOULOS, G., and PATSAKIS, C. A survey on long-range attacks forproof of stake protocols.

[5] Dickman, T. Pow 51% attack cost.

[6] Edgington, B. A technical handbook on ethereum's move to proof of stake and beyond.

[7] Eyal, I., Gencer, A. E., Sirer, E. G., and van Renesse, R. Bitcoin-ng: A scalable blockchain protocol.

[8] Kiayias, A., Russell, A., David, B., and Oliynykov, R. Ouroboros: A provably secure proof-of-stake blockchain protocol.

[9] King, S., and Nadal, S. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake.

[10] Lee, T. B. Bitcoin's insane energy consumption, explained, 2017.

[11] Malahov, Y. G. Hyperchains — secure, cheap & scalable blockchain technology for everyone, 2016.

[12] Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system.

[13] Niu, J., Wang, Z., Gai, F., and Feng, C. Incentive analysis of bitcoin-ng, revisited.

[14] Nxt community. Nxt whitepaper, 2016.

[15] Sharma, A. Understanding proof of stake through it's flaws. part 2 — 'nothing's at stake', 2018.

[16] Sharma, A. Understanding proof of stake through it's flaws. part 3 — 'long range attacks', 2018.