PADLOCK
Protocol Allowing Decentralized Large thrOughput Coins, Kool

Ellis Frank

July 21, 2021

Abstract

PADLOCK is a cryptocurrency protocol that is designed to be decentralized, secure, and large throughput. It achieves this through sharded block verification and UTXO set, a novel consensensus mechanism referred to as "Proof of Staked Work", UTXO commitments, and UTXO expiry. When all of this is added together, it creates a cryptocurrency that is able to match, and vastly exceed Visa capacity.

1 Introduction

Most blockchains suffer from the scaling trilemma: decentralization, security, and scale; pick two. In order to remain decentralized, running a verifying node and mining on it should be easy. Running a verifying node allows the user to ensure consensus rules are being followed. Mining should be accesible to users as well, in order to ensure censorship-resistance, and egalitarian distribution of newly minted coins. However, this inversely proportional to scale. Scaling to a larger number of transactions puts a burden on mining and verifying nodes. This centralization opens the network to several attacks, such as the one describe by Vitalik Buterin [1], where the nodes and major miners of a blockchain hard fork and mint new coins. This paper attempts to demonstrate a protocol capable of scaling well remaining decentralized, and secure.

2 Sharded Block Verfication

Sharded verification is achieved by having each node only verify a random portion of each block. If the portion they verify is invalid, they will submit a fraud proof to the network, and that block will be dropped.

A fraud proof consists of data in that block that is invalid. Attached to the data must be various merkle proofs that proves the invalid data is actually apart of the block.

There is one type of fraud proof that would require verifying the entire block, which is calculating fees owed to the miner. In order to calculate this, a node would have to add up the fee from every transaction in the block, thus requiring verification of the entire block to prove fraud.

In order to fix this, there will be 256 sections of each block, and each section specifies the sum of the fees in each section. In order to prove that the total fee is wrong, only one 256th of the block needs to be revealed.

3 UTXO set sharding

The UTXO set will be split into 64 different shards. Each block will be split into 64 different equal sections, and each one will change a different UTXO shard. However, this leads to a problem; many of the transactions in each shard won't be apart of that nodes utxo set.

To solve this, the UTXO set will be stored as 64 separate UTreeXO [2] style accumulators. Each node will need to store the roots of every accumulator, which isn't likely to be over 500 bytes per accumulator. Every transaction input will need to contain a merkle proof to one of the accumulators.

3.1 Minimizing proof sizes

Proof sizes can be greatly reduced by merging merkle proofs. Each merkle proof supplies the prover with the locations of various hashes in the merkle tree.

$$4400 * 60 * 60 * 24 * 30 * 3 = 34,214,400,000 \text{ UTXOs}$$

$$2^{35} = 34,359,738,368 \text{ Merkle tree of height 35 required}$$

$$120 * 4400 = 528,000 \text{TXs in a block}$$

$$2^{19} = 524,288$$

Each transaction in the block **must** share at least 19 branches in the merkle tree, meaning that those 19 branches will only need to be stored once. This means every transaction except a few will be able to remove at least 19 hashes from their proofs, resulting in only 16 hashes being needed. Using 28 byte hashes, this adds an extra

448 bytes (assuming one input) per transactions, which is very manageable. This extra data is also in the right place. Transaction data isn't an ongoing cost, as old transaction data can be pruned without a loss in security, whereas the UTXO set is a permanent cost and cannot be pruned.

3.2 Verkle Tree Potential

Once the libraries and cryptography around verkle trees are more mature, merkle trees can be swapped for verkle trees, which would greatly reduce proof size.

4 Sharded Block Production Isn't Needed

Sharded block production won't be required until PADLOCK is experiencing large amounts of throughput. Producing blocks isn't the bottleneck with blockchains. Due to the need to sync, validating blocks should only take around 5-10% of a validators processing power. This is in part to protect from DDOS attacks, but mainly because of the time it takes to sync. If 100% of processing power is used, if a node goes offline, it won't be able to sync with the rest of the network, as it will always remain behind by the amount of time it was offline.

However, for producing blocks in a system with sharded block verification, this isn't an issue, as the verification time will be significantly lower than the time it takes to produce blocks. Bitcoin ABC has been shown to be able to validate 3000 transactions per second well running 3 nodes on a 4 core desktop at 100% cpu usage [4].

Sharded block production would need to be implemented once throughput starts to be in excess of 5000 transactions per second.

5 UTXO Expiry

Well it is possible to fully verify the blockchain with only the roots of the accumulators, users wanting to spend their UTXOs will need to aquire the proofs from somewhere. In order to facilitate this, nodes will have the option to store proofs for one or many of the UTXO shards.

However, some UTXOs are sent to invalid addresses, or aren't going to be spent for a long time. So that nodes won't have to store these sorts of UTXO proofs, UTXOs will be unspendable after 3 months, and nodes won't have to store their proofs anymore. These UTXOs can be reintroduced by providing a merkle proof that they were apart of the UTXO set before it was pruned.

6 Fast Syncing

Because the roots of the accumulators are stored in every block, a node can start syncing from any point in the blockchain, rather than syncing from the very beginning. This is secure assuming that the blockchain was valid up until the point when the node started syncing.

7 Proof of Staked Work

The goal of Proof of Staked Work is to provide a consensus mechanism that doesn't require constant energy expenditure, and is easy to participate in. Proof of Work is undesirable because it requires constant energy expenditure. Proof of Staked Work mitigates this, by requiring participants to prove they have computing power very infrequently. In a general form, in order to become a block producer, you must first register by proving you have computational power. What makes this different than normal proof of work, is that block producers don't need to prove they have computing power every time they make a block, and there isn't a need for constant energy expenditure like in proof of work.

7.1 Registration

To be registered as a block producer, you must mine a valid registration block. Registration blocks happen once every 50 blocks. A registration block doesn't contain any actual transaction data, but rather just proofs of work.

Many block producer candidates can pool together as well. They do this by having the registration block contain a list of all of their addresses, then a list of nonces, with the nonce producer's address appended to the nonce. This has a capacity for 50 nonces. The cumulative difficulty of every nonce is added, and must reach a certain threshold. The difficulty target is set to target 60 seconds. The mining algorithm is RandomX. RandomX is most efficiently mined on traditional CPUs, which makes it much easier for people to participate, as they don't need a good GPU, or ASIC.

Every registration block, existing block producers score will be reduced by 0.1%. This is done to incentivise existing block producers to try and mine new registration blocks.

7.2 Flow of Block Creation

First, a block producer is chosen via a verifiable delay function. This function is simply a proof of work on the previous block header. A nonce will be hashed with the previous block header. If the resulting hash doesn't meet the difficulty target, the nonce will be incremented and the process repeated until a valid hash is found. The difficulty target is designed to target 30 seconds. Once a valid hash is found, the last 8 bytes are represented as an unsigned 64 bit integer. The next block producer is chosen by checking which block producer's range the hash is in.

The block producer will collect transactions from the mempool, and then add them to their block. The block producer will then have to complete a proof of work on their block to prevent spam. The difficulty of this proof of work adjusted every block to reach an average of 60 seconds per block. Once this is done, they broadcast their block, and the network starts the verifiable delay function to select the next block producer.

In the event the selected block producer is offline, then nodes will continue to increment the nonce in the verifiable delay function until a valid hash is found. That new hash will select the next block producer.

7.3 Consensus

The valid blockchain is considered the *heaviest* chain. Normal blocks will possess one weight unit. However, in the event where there was a second block producer selected, if the first block producer, does provide a block, the second block, and the following 5 after it, will be worth 0.8 weight units.

7.4 Incentives

In order to incentivise being a block producer, they will periodically receive a reward based on their score. The total amount given in each period will be fixed (i.e. the inflation doesn't increase when there are more block producers). This is how new supply gets put into circulation.

7.5 Reduced Block Reward

Because block production takes much less energy than it would in Proof of Work, block producer's reward can be safely reduced without a drop in security. This showcases the largest benefit of Proof of Staked Work: it produces the same security as proof of work (security measured by the cost of acquiring all the hardware it would take to control 51% of the hash rate), well requiring much less energy expenditure.

7.6 Attacks

7.6.1 Stake Grinding

Stake Grinding is an attack that can be done on certain implementations of proof of stake. In these implementations, the next block producer is chosen with a verifiable random function (VRF) seeded with previous blockchain data. The issue with this design is that a block producer could simply change block parameters until the VRF produces a result that picks themselves as the next block producer. This design is immune to this, because generating a new block requires a proof of work, so generating blocks is slow. On top of this, there is the verifiable delay function, which would slow down an attacker even more. In the time it would take them to find block parameters that benefit them, another block producer would have been chosen.

7.6.2 Nothing at Stake

The Nothing at Stake problem happens when there is a fork in a Proof of Stake blockchain. Stakers are incentivised to stake on both chains at once, in case one chain gets ahead of another, and they are selected as block producer on the forked chain. In order to dis-incentivise this, block producers can include a fraud proof consisting of a chain of block headers from forked chains in order to prove certain block producers were extending forked chains. This will result in those block producers score dropping by 80%. This incentivises all block producers to remain on the same chain, to reduce the risk of their score going down.

8 Privacy

Privacy is first achieved through pseudonymity. Users can also generate new addresses for every transaction. However, this still has some pitfalls. If someone receives a transaction and they know the sender, they are able to see their balance. To combat this, PADLOCK uses confidential transactions. Confidential transactions use homomorphic encryption to hide transaction amounts, well still allowing them to be verified for zero balance sums [3]

9 Throughput Estimate

There are a total of 64 shards. With a low amount of nodes, in order to remain secure, nodes will want to validate multiple, or all, of the shards. However, with a large amount of validating nodes, it becomes increasingly likely that every shard in a block will be validated by one of the nodes, therefore making it secure to validate a lower number of shards. Ideally, nodes will only need to validate one shard. Assuming similar performance to Bitcoin ABC (which is likely, considering it is based on Bitcoin Core which doesn't have very much parallelization, as well as less database requests because of UTreeXO), at 100% load, a typical 4 core desktop can handle at least 3000 transactions per second. In order to make syncing quick, no more than 10% of the processing power should be used to validate blocks. This would allow one node to process 300 transactions per second. With 64 shards, that's 19200 transactions per second. However, before that point, sharded block production should be implemented.

10 Adaptive Block Size Cap

There will be a cap on block size in order to prevent spam attacks. However, this cap shouldn't be a hard coded constant. Rather, it should be adaptive, so that a potentially contentious hard fork isn't required every time the throughput cap starts to be met. The block size cap for PADLOCK is calculated by multiplying the median block size of the past two months of blocks by two. In order for an attacker to increase the cap to unweildly amounts, they'd have to produce half of the blocks in the two month time period.

11 Conclusion

In this paper, we have outlined the general structure for how the PADLOCK blockchain will work. It allows for very high throughput well remaining secure and decentralized thanks to sharding, requires little energy expenditure, and keeps users privacy by hiding transaction amounts. The description offered here is subject to change as more research is done, and as implementation is done.

References

Buterin, V. (2021). The limits to blockchain scalability. https://vitalik.ca/general/2021/05/23/scaling.html

Dryja, T. (2019). Utreexo: A dynamic hash-based accumulator optimized for the bitcoin utxo set. https://eprint.iacr.org/2019/611.pdf

Maxwell, G. (n.d.). Confidential transactions - investigation. https://elementsproject.org/features/confidential-transactions/investigation

Toomim, J. (2019). 3,000 tx/sec on a bitcoin cash throughput benchmark. Retrieved July 20, 2021, from https://www.youtube.com/watch?v=j5UvgfWVnYg