

A Lightweight Blockchain Consensus Protocol

Keir Finlow-Bates

keir@chainfrog.com

Abstract

A lightweight yet deterministic and objective consensus protocol would allow blockchain systems to be maintained and extended in a cheap and efficient manner, with even low powered computing devices such as smartphones, point-of-sales terminals and Internet-of-Things devices being able to participate equally in the creation and validation of blocks. In this paper we present such a consensus protocol based on the pre-announcement of miners identified through a hash of their public key and a unique address identifier, and a chain scoring system based on the proximity of the miner's hash to a hash of the preceding block, or a root hash of a Merkle tree produced from a number of preceding block hashes. This protocol should be reliable for permissioned and private blockchains, and may well be extensible to public blockchains.

Keywords: blockchain, consensus, proof-of-work, proof-of-stake

1. Introduction

Blockchains are built on consensus-forming protocols, which are used to determine which packaged block of data or transactions will be the next to be added to the end of the chain. Although effective, the most prevalent consensus protocol called proof-of-work (PoW) and used by Bitcoin [1], is an inefficient energy-consuming solution that has initiated a dedicated hashing hardware arms-race. Other protocols have been proposed and implemented, such as: proof-of-stake (PoS), proof-of-elapsed-time (PoET), and practical Byzantine fault tolerance (pBft). However, PoS relies on subjective consensus, PoET requires dedicated trusted execution environment hardware, and pBft is implemented through multiple rounds of participant voting, requiring messages in the network to be signed multiple times. This introduces a disk space overhead to store all the signatures. A more in-depth analysis

of the most common consensus protocol variants, PoW and PoS, and their respective strengths and weaknesses can be found at [2].

A consensus protocol must be objective, that is, any newcomer to the blockchain system must be able to independently determine from the blocks available which sequence of blocks forms the actual blockchain without requiring external guidance. It must also be deterministic: two entities with the same set of blocks must, through application of the protocol, produce the same chain and come to the same conclusion as to which block constitutes the end of the chain. The consensus protocol must be resistant to alteration, in that it should be impossible or prohibitively expensive to attempt to create and extend a fork off the main blockchain from a point earlier in the chain. And finally, when applied in a collaborative environment it should have a low computational overhead to reduce energy and hardware requirements.

As it stands, running a validator node (or “miner” in blockchain terminology) in order to maintain and extend a blockchain requires significant levels of some or all of: computational power, memory, energy and dedicated hardware. As a result it is unfeasible for Internet-of-Things devices, smartphones, point of sales terminals, and other lower-powered yet ubiquitous computer equipment to participate in blockchain maintenance. Mining on blockchains has become the preserve of large server farms in cheap-energy locations, and both the trustworthiness of the blockchain and the global environment have suffered accordingly.

We therefore propose a new blockchain consensus protocol that is neither computer-hardware specific nor energy intensive in order to deal with these known problems.

2. Miners

In order to be allowed to participate in the generation of proposed blocks to be added to the blockchain, under our new protocol a miner must pre-announce its intention to mine by submitting an announcement notification or transaction to the peer-to-peer network for inclusion in a block.

A waiting period of N blocks is then imposed, and it is only after that many blocks have been added past the initial notification block that a miner’s contributions may potentially be included in the blockchain. The parameter N can be adjusted for a specific blockchain implementation based on the expected robustness of the blockchain and estimated number of miners that will be participating, and also on the probability of faults or attacks. For

example, for a permissioned blockchain running on an internal network, N might be set quite low to 8, whereas for a public blockchain running on the Internet where occasional malicious activity may be likely, the value of N would be better set to something high such as 2048.

One specific miner is not bound by the waiting period, namely the miner announced in the first “genesis” block of the blockchain. It may start mining right away, in order to ensure that the chain can actually grow.

A miner notification message must contain the miner’s public key, which in the case of a blockchain with cryptocurrency rewards would be the address the rewards will be paid to, and a unique address identifier (UAI). A UAI may be an IPv4 or IPv6 network address, a MAC address, or even a domain name, depending on the specific implementation of the consensus protocol. The public key and UAI are concatenated and hashed with a standard cryptographic hash function such as SHA256, to produce a miner identification number (MIN). The hash function used must be one-way, collision-proof, and with an output that cannot be predicted in advance, i.e. a random oracle. A specific public key cannot be concatenated with multiple UAIs to produce multiple MINs for one miner. There should ideally be a cost associated with the UAI to prevent one miner generating a large number of MINs.

It is important that there is a standardized format or naming convention for the address type picked as the UAI. For example, for IPv6 addresses the format recommended in [3] could be used.

The MIN is used in the consensus protocol to calculate which miner has submitted the block, and this is discussed in greater detail below.

3. Chain scores

At its simplest, a consensus protocol consists of three actions:

- Defining a function which takes as its input a sequence or chain of blocks, and outputs a score for the chain,
- Examining each possible chain given the available proposed blocks, by feeding each chain to the function, and recording the score returned,
- Selecting the chain with the “best” score.

The second and third steps are relatively easy to implement, so the remaining requirement is to define the consensus protocol’s chain scoring func-

tion. For our function we propose a procedure based on the proximity of the miner's MIN to a hash calculated from one or more of the preceding blocks.

The new block must have a timestamp no earlier than $T - \delta$ where T is the average block generation time and δ is a window of block generation allowance. The aim is to produce blocks at a rate of about one every T seconds. Naturally the time limit is one-sided, to prevent the chain generation process grinding to a permanent halt if no miners are available for a period of time. The block to be selected is generated and signed by the miner with the MIN closest to one of:

- The hash value of the preceding block, or
- The hash value of the concatenation of the hash values of the previous K blocks for some value of K , or
- The root hash value of the Merkle tree[5] of the last 2^K blocks, for some value of K .

Determining whether the first proposed schema using a hash of a single preceding block would suffice, or whether a more complicated schema involving hash chains or Merkle trees would be more robust remains a potential topic for future research. It is however likely in our opinion that some kind of hash chain offers more resilience as it links multiple blocks together.

In the unlikely event that two valid MINs are equidistant from the hash output of the mechanism chosen, the lower hash value wins.

Blocks can only contain data packages that predate the block timestamp, by at most $T * f$, where f is a multiplier (i.e. if a transaction is not included by $T * f$ it has become stale, cannot be included and should be resent).

A block value score is then computed using:

$$score = 2^{256} / |H - M| \tag{1}$$

or

$$score = 2^{256} - |H - M| \tag{2}$$

where H is the hash value returned from applying the hash function to the preceding block or blocks, and M is the miner's identification number.

Again, the choice of whether to use division or subtraction to determine the block score depends on a number of factors, such as the difficulty of performing long division on large numbers. A simplified method using only

the first s significant bits of the hashes for some small value of s might also improve the ease of coding an implementation of the process.

The complete score for a given chain of available blocks is the total sum of the score for each block in the chain. The score of each potential chain can therefore be computed, and the chain with the highest score wins and becomes the chosen blockchain.

The system is deterministic provided the key miners with announced MINs are running on the system. That is: given a block at the end of the chain, it is possible to determine the prime candidate for generating the next block. However, in practice some miners will drop out of mining over time. How to handle this is discussed in the next section.

4. Revoking miners and preventing flooding

Miners may occasionally become non-functioning, go off-line for a while, or even leave the blockchain entirely. If a miner with the closest MIN to the current chain score fails to report the next block because of this or some other reason, another miner should produce the next block to continue the chain. Therefore the subsequent closest miner can submit a block. An optional extension to the system is to allow this miner to include a MIN cancellation notice in the new proposed block for the closer miner that failed to report a block. This then prevents the miner with the canceled MIN from reporting future blocks until it has re-announced itself and the usual waiting period has passed.

Similarly, if the M closest miners fail to report a block, the $(M + 1)$ th miner can submit a block, including a cancellation notice for the other closer miners.

Miners should build a table of active MINs by parsing the blockchain and updating their records on every block, in order to ensure they have an active snapshot of potential miner activity for current mining.

A related issue is the flooding of the peer-to-peer network with proposed blocks by miners that do not feasibly have a chance of submitting a successful block. As there is no real cost associated with submitting a block, many miners will produce and submit one every block round. If not handled properly, this could lead to the network becoming flooded with useless block proposals. In particular, miners with MINs far from the chain score will be generating blocks containing a substantial number of cancellation notifications.

The easiest method for dealing with this is for peers to simply drop any received proposed block that is submitted by a miner with a MIN that is significantly low on the list of potentially successful miners. Over time it should be possible to build up a statistical predictor to determine the chance of any given miner submitting a successful block, and a punishment scheme could be implemented, for example to exclude miners for a time proportional to their MIN distance from the preceding chain score if they repeatedly submit blocks that stand no chance of being accepted.

5. Stake grinding and early chain attacks

An issue which PoS suffers from, and which at first glance our consensus protocol might appear to be vulnerable to is stake grinding [4]. In PoS an attacker scans the entire blockchain to determine a point where their stake produces a better score for producing a subsequent block than the currently accepted successor. The attacker starts building a chain fork from that earlier point in the chain. Similarly, in our protocol an attacker could look for a block with a successor hash that is closer to their announced MIN than that of the currently accepted miner's MIN.

However, due to the miner announcement waiting period the attacker can subsequently only use their own previously announced miners to grow their fork, and possibly the resources of a small number of neutral miners who happen to only have synchronized their blockchain to the fork point at that precise time in history.

Furthermore, the attacker cannot add miner announcements with fake miners within the blocks of the forked chain to improve the chance of growing that chain, because adding each notification will change the hash value of the block containing the announcement, removing the advantage gained by having a miner with a more proximal MIN number.

6. Conclusion

We have proposed a consensus building system for blockchains that does not rely on performing large numbers of wasted calculations, elaborate voting schemes, and does not depend on amassing substantial quantities of cryptocurrency in order to participate in maintaining and extending a blockchain system. We started with an announcement system to ensure that miners

have to register in order to mine on the blockchain, and to prevent the generation of spurious mining accounts in order to manipulate the odds of being the next entity to generate a successful block. We added a scoring system based on the proximity of any given miner's identity number to a hash of one or more preceding blocks in order to determine a score for any proposed successor block, and presented several scoring schemes for determining the value of a given chain of blocks, allowing third parties to independently and objectively decide which chain is the most successful.

A number of different block hashing procedures and chain scoring methods were presented as potential candidates in order to optimize the consensus protocol implementation, along with parameters that require fine-tuning and empirical investigation in order to determine which configuration and parameterization would be best for any given blockchain system.

- [1] Nakamoto, S., *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2008, retrieved on 12 Oct 2016 from <https://bitcoin.org/bitcoin.pdf>
- [2] Bitfury Group, *Proof of Stake versus Proof of Work*, 2015, retrieved on 12 Oct 2016 from <http://bitfury.com/content/5-white-papers-research/pos-vs-pow-1.0.2.pdf>
- [3] Kawamura, S. and Kawashima, M., IETF RFC5952, *A Recommendation for IPv6 Address Text Representation*, 2010, retrieved on 12 Oct 2016 from <https://tools.ietf.org/html/rfc5952>
- [4] Poelstra, A., *On Stake and Consensus*, 2015, retrieved on 12 Oct 2016 from <https://download.wpsoftware.net/bitcoin/pos.pdf>
- [5] Merkle, R. C., *US4309569: Method of providing digital signatures*, 1979