PROPOSAL TO MODIFY SATOSHI CONSENSUS TO ENHANCE PROTECTION AGAINST 51% ATTACKS

A Penalty System For Delayed Block Submission

June 14, 2018

Alberto Garoffolo, Pier Stabilini, Robert Viglione, and Uri Stav

INTRODUCTION

The longest chain rule, or Satoshi Consensus, worked well in the relatively decentralized environment in which it was introduced in 2009. Mining resources have since concentrated and dropped in cost for lease, such that the original dominant strategy of playing by the rules no longer holds for all proof-of-work (PoW) blockchains that rely on the longest chain rule. As recent events have proven, in some circumstances it can be economically feasible to launch 51% attacks on operational public blockchain networks. This paper proposes a novel adjustment to Satoshi Consensus that makes it exponentially more costly, and hence unlikely, to launch such attacks for any proof-of-work mineable cryptocurrency system.

The common method for performing a double spend attack is as such:

- Execute a transaction T_1 sending coins from address *A* to an exchange address on the current public chain
- Privately mine a block on a parallel forked chain containing a transaction T_2 sending coins from address *A* to another address
- Wait for T_1 to be confirmed by the exchange and in the meantime continue to mine privately the parallel chain at a faster rate than the public chain
- Trade the confirmed coins on the exchange, then withdraw funds to private address *B*
- Broadcast to the network the private chain that is longer than the public one
- at this point, the network will adopt the attackers private chain as the new public chain (as it is longer) and miners will start mining on the newly reorganised public chain.
- T_1 is no longer valid and the attacker already used the coins fraudulently

To summarize, these attacks are possible because the system allows to “overwrite” the current node view of the blockchain history with the new one after a user accepted a specific transaction (e.g. after the Exchange waited for the confirmation time). This obeys Satoshi's principles given in the Bitcoin white paper and implemented by most of PoW cryptocurrencies. Block generation in PoW consensus is a stochastic process and honest miners can generate blocks in parallel on the same height (with probability depending on block generation time and network delays), for which Satoshi Consensus has a simple method to adjudicate by ultimately defaulting to the chain with the most accumulated work. Thus, pruning shorter branches with lower accumulated work has been an efficient way to keep the linear sequence of blocks across these distributed systems.

Satoshi's white paper is based on the assumption that a majority of computational power is controlled by honest nodes ("one-CPU-one-vote" principle), and within this assumption it is almost impossible to create any reasonable length adversarial branch to implement the double spend attack. Conventional heuristics have converged on the idea that it is sufficient to wait for several confirmation blocks to get very low probability of transaction cancellation.

Many things changed since publishing the Bitcoin white paper. Some of the most significant changes that jeopardize the longest chain rule, are the appearance of ASIC miners and other computation boost techniques that completely break the "one-CPU-one-vote" principle. Many cryptocurrencies share the same mining algorithm, but have extreme differences in hashrate, which allows for the computation of one cryptocurrency mining pool to potentially be used to attack another chain.

Thus, we see that Satoshi's assumption about honest majority of computational power and the impossibility of long adversarial branches may be broken in the modern environment, evidenced by the latest 51% attacks on several cryptocurrencies. We need a more advanced and more comprehensive system that satisfies modern conditions. The core PoW consensus requires corrections to provide secure services, while still allowing the possibility of honest miners generating conflicting blocks, and legitimate network delayed synchronization.

To achieve this goal we have identified a modification that significantly increases resources needed for a successful double spend attack (with no changes in honest hash rate) and shifts the cost of an attack to become economically senseless. Moreover, the increase of resources needed to make a successful attack will not require increased confirmation adjustments by receiving parties, including exchanges.

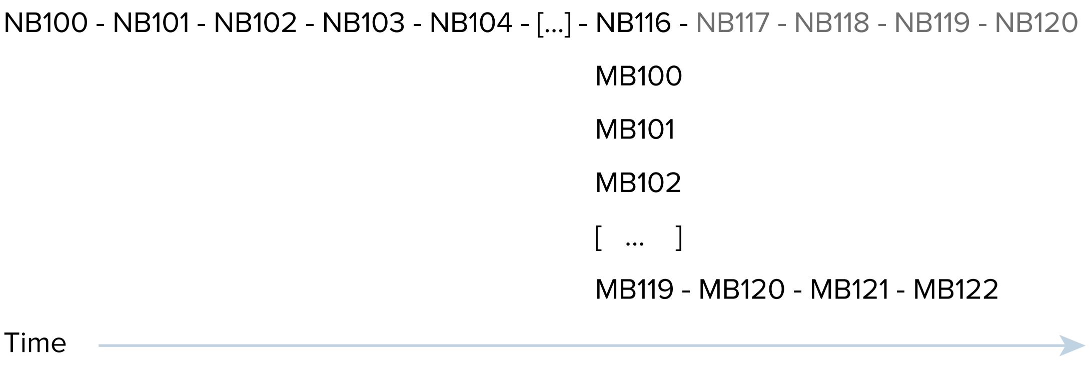
THE DELAYED BLOCK SUBMISSION PENALTY APPROACH

Considering that private mining is the source of a double spending attack, to make it less effective we introduce a penalty in the form of a block acceptance delay in relation to the amount of time the block has been hidden from the public network. Time being measured in block intervals, not temporally via timestamp.

For example, let's assume to have the following block reception scenario:

NB_i = Normal block

MB_i = Malicious block



NB100 - NB101 - NB102 - NB103 - NB104 - [...] - NB116 - NB117 - NB118 - NB119 - NB120

MB100

MB101

MB102

[...]

MB119 - MB120 - MB121 - MB122

Time

Figure 1

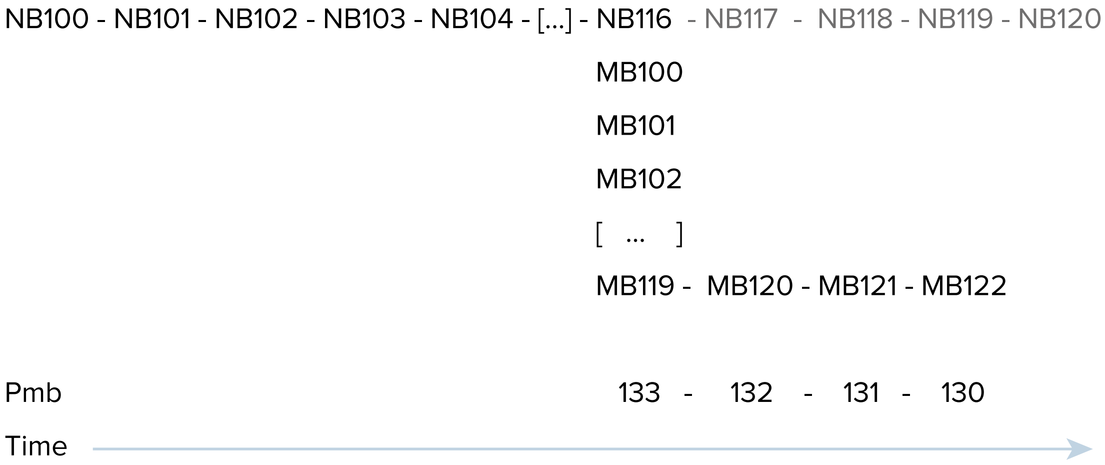
In the current system, blocks NB117, NB118, NB119, NB120 are not going to be mined because as soon as the network received block MB119, the malicious chain becomes the active one and the network will start mining from block MB119 abandoning the normal chain.

To prevent this, we introduce a fork acceptance delay related to the amount of time the fork has been hidden from the public network. This delay represents the number of blocks for which the adoption of the new parallel chain will be postponed. For an adversary it means that he will need to continue to mine the malicious fork even after revealing it and until the moment when the delay is finished.

For example, let's consider the following delay function *DF*. To define *DF* we first introduce *BDi* which effectively represents block reception delay defined as a difference between current main chain height and height of the received block (e.g. BDmb100=16, BDmb101=15, ..., BDmb115=1, BDmb116=0, BDmb117=-1, BDmb118=-1) or -1 in case the height of the received block is greater than the current main chain height.

The delay function *DF* in this case for the whole forked chain will be represented as the SUM of BDmb[i] values, where *i* represents indices of the blocks in the forked chain.

In the following example on Figure 2, the delay DF(MB100,...,MB119) = 16+15+14+13+12+....1+0-1-1-1 = 136 - 3 = 133 blocks. So the adversary will need to keep mining his fork after it became public for another 133 blocks until it will be accepted by other nodes.



NB100 - NB101 - NB102 - NB103 - NB104 - [...] - NB116 - NB117 - NB118 - NB119 - NB120

MB100

MB101

MB102

[...]

MB119 - MB120 - MB121 - MB122

Pmb 133 - 132 - 131 - 130

Time

Figure 2

With such a delay function, if we assume for example to adopt a 20 block confirmation time, a 21 block delay punishment would be 21 * (21 + 1) / 2 = 231, then the minimum number of blocks to be mined to perform the attack would be 231 + 21 = 252 blocks

Please note that the confirmation time cannot start until a fork is in progress.

Proper tuning of the delay function will complicate an attack to the point of infeasibility since it will require significantly more resources and will also give an additional opportunity to react on the fork before its final adoption. The network will learn of the contentious fork and it will have to reduce the full delay function (*DF* = 0) before the network defaults to the chain as truth. During this period, network participants, such as exchanges, can freeze potentially fraudulent deposits until the issue has been resolved, e.g. either the attempted fraudulent chain is abandoned, or it has successfully driven *DF* → 0 through brute force. Honest miners continue to add to the active chain.

The delay function may also consider current mining difficulty (*d*) to provide better protection for coins with low hashrate (e.g. increasing delay by a factor inversely proportional to the current difficulty). Such an augmenting factor to *DF' = DF * f(d)* can be ignored by setting f(d) = 1, but any sensible functional form can be substituted in that has the properties of low impact for the range of honest simultaneous block contention, then scales exponentially outside any reasonable range for which we can assign dishonest motives.

Although the introduced feature complicates the whole specter of mining attacks (those executed by mining private parallel forks), it does not prevent the network from converging in case of a natural split (e.g. when some parts of the world are completely segregated from each other for some time). The period of convergence will be longer than the pure longest chain rule, but honest miners will work on converging chains by driving $DF \rightarrow 0$.

Note that if an adversary mines his chain publicly the delay will not be applied, but in this case everyone can see the fork and is able to take preventive measures (e.g. exchanges will increase confirmation periods, etc.).

CONCLUSION

The operating environment for cryptocurrency systems has changed significantly from its origins in 2009 when mining power was far more decentralized. Satoshi Consensus, or the longest chain rule, worked well to adjudicate natural chain forks by simply deferring to the chain with the most accumulated work. Both technical limitations and economic incentives combined to render the longest chain rule the dominant strategy for any miner, whether honest or nefarious. This is no longer the case and public blockchains need to upgrade consensus rules to make it far more costly to succeed with double spending.

This proposal gives one such method that has a simple form that is, itself, quite promising, but which can be generalized with a scaling function to make it both technically infeasible and economically disastrous to attempt double spending. The method permits adjudication of honest miners simultaneously solving blocks, as well as legitimate network fractures that resolve over time. No attack vector should ever be considered permanently neutralized, but this method certainly renders one common method far less likely. Additional research into layered defense strategies, such as introducing interval block notarization schemes on top of this penalty system could make the system even more secure and should be investigated.

ACKNOWLEDGEMENTS

SPECIAL THANKS

to the team and community for contributing:

@cronic, Peter Stewart, @ultimateblockage,
Rosario Pabst, Rowan Stone, Gustavo Fialho and Vitalik Demin.

Creatives by Lucy Wang, Linda Baksija and Marko Orčić.