Twenty Percent and a Few Days – Optimising a Bitcoin Majority Attack

Ansgar Fehnker[1](✉) and Kaylash Chaudhary[2]

[1] Formal Methods and Tools Group, University Twente, Enschede, The Netherlands
ansgar.fehnker@utwente.nl

[2] School of Computing, Information, and Mathematical Sciences, University of the South Pacific, Suva, Fiji

Abstract. Bitcoin is a distributed online payment system that organises transactions into blocks. The size of blocks is limited to 1 megabyte, which also limits the number of transactions per second that can be confirmed. This year several attempts have been made to create a fork or a split that removes this restriction. One such alternative is Bitcoin Unlimited (BTU). Proponents of BTU have suggested to use a type of majority attack to force other Bitcoin miners to adopt BTU.

In this paper we model this attack in Uppaal, and analyse how long it will take for an attack to succeed, depending on the share the attacker has of the total network, and the so-called confirmation depth. The analysis shows that with a share of 20% an attack will be successful within a few days. This paper also looks at the effect of increasing the confirmation depth as a countermeasure.

1 Introduction

In circulation since 2009 [9], Bitcoin is the most popular digital currency. Bitcoin is managed by a peer-to-peer network. Every peer keeps a record of all transactions in a public ledger. Transactions are organised into separate blocks, all of which are linked to their immediate predecessor, forming a chain. The protocol uses a proof-of-work solution to induce a unique order on blocks, a process known as mining. As the difficulty of mining increased over the years, peers started working together in so-called pools.

Bitcoin has a block limit of 1 megabyte, which limits the number of confirmations to 3 transactions per second. This year has seen coin splits such as *Bitcoin Cash*, or forks, such as *SegWit*, aimed at this limitation. Proponents of one alternative, Bitcoin Unlimited (BTU), have suggested to use a type of majority attack to force adoption of BTU [11]. We will refer to this attack as the *Andresen* attack, after the former lead developer of Bitcoin who proposed the attack. Current lead developers of Bitcoin have argued that an attack on the main fork is a waste of computing resources [10]. That assessment will however depend on how many computing resources are required, and for how long.

A. Dutle et al. (Eds.): NFM 2018, LNCS 10811, pp. 157–163, 2018.
https://doi.org/10.1007/978-3-319-77935-5_11

This paper uses statistical model checking with Uppaal [7] to analyse the success chance, depending on the size of a mining pool. We use a parameter sweep to identify an optimised strategy, and then use this strategy for further analysis of potential counter measures.

Beukema [3] developed a formal model for double spending and corrupt peers. Andrychowicz et. al. modeled Bitcoin contracts with timed automata [2], and verified that an honest party can not lose Bitcoins. Herrmann considered the implementation, evaluation and detection of double-spending attacks [8]. This related work did not include blockchain forking, which is the focus of [6].

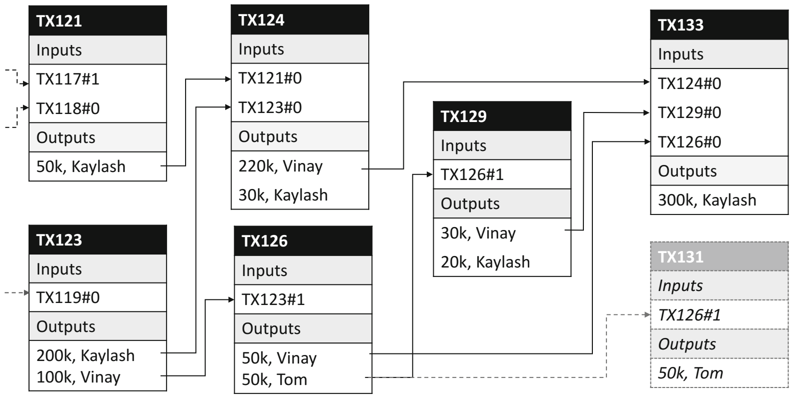


Fig. 1. Transaction graph

2 The Bitcoin Protocol

Bitcoin is a decentralised peer-to-peer electronic cash system, without central trusted authority [9]. Public/private key cryptography ensures the validity of transitions, while a so-called *mining* process determines the order of transactions.

Transactions. There are two types of transactions: coin-base and regular transactions. In this paper we consider regular transactions, transferring existing Bitcoins from one user to another.

Each transaction has one or more transaction inputs and one or more outputs. An input is a reference to an output of a previous transaction. It proves that the senders possess the Bitcoins they claim to have. The transaction output specifies an amount and a recipient.

Figure 1 gives an example of a transaction graph. Transaction TX124 has two inputs; transactions to user Kaylash worth 250k, and two outputs: 220k to user Vinay, and 30k to user Kaylash[1]. The first output is in turn an input of transaction TX133, and thus spent. The second has not been used and is thus unspent.

[1] The amount is expressed in Satoshi (1 BTC is 100 000 000 Satoshi).

To guard against double spending each output can only be used once. TX129 and TX131 cannot both be part of the transition graph, since both spend the second output of TX126. To impose an order on transactions – to decide which transaction came first – the Bitcoin uses a so-called *blockchain*.

Blockchain. A block contains a set of transactions, a header, and the hash of the predecessor block. Transactions in the same block are considered to have happened at the same time. Transactions in different blocks are ordered using the predecessor relation between blocks. Transactions are only confirmed if they appear in some block; unconfirmed transactions are kept in the *transaction pool*. A peer selects transactions from the transaction pool for a new block at the end of the block chain, provided it completes a so-called *proof-of-work*.

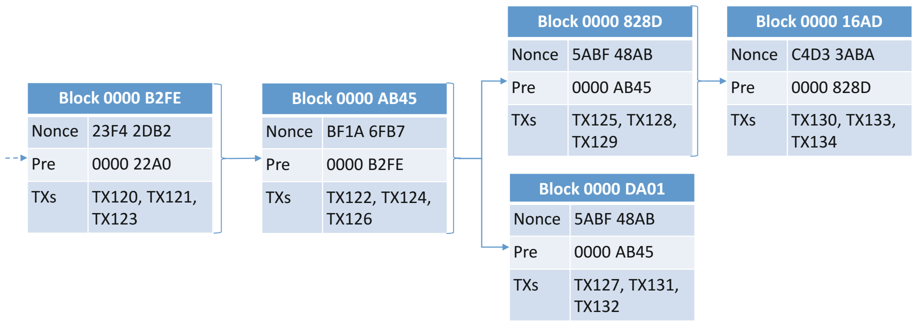


Fig. 2. Blockchain

For the *proof-of-work* the node randomly selects a *nonce*, which will become part of the block header. The hash of the block (including this nonce) is calculated and the result is compared with the *target value*, set by the Bicoin network. If the hash is lower than the target value, the proof-of-work is completed; otherwise the node repeats this process. Once a nonce is found, the block becomes valid and is broadcasted to the network. This process is called *mining*.

With different *miners* working on different blocks, the blockchain may have forks, i.e. that two blocks are created and broadcasted over the network simultaneously. Some peers might receive the first block first, and others the second. Peers continue with the block they received first.

Transactions in the longest chain are considered confirmed. Other transactions are added back to the transaction pool, and can be used to build new blocks. Miners will usually extend the longest chain, because they will only be rewarded for blocks in the longest chain. There is a non-negligible probability that a fork of the longest chain will become the longest. The distance of a block to the end of the chain is called the confirmation depth. It is advised to only consider transactions in blocks with confirmation depth 6 or more settled [4].

Figure 2 depicts a blockchain that includes the transaction of Fig. 1. Transaction *TX126* is included in *Block 0000 AB45*, and TX129, which uses an output of 50k for Tom from *TX126*, is in *Block 0000 828D*. *TX131* cannot be included in this block, or any of its successors, since any output can only be used once. *TX131* could however be included in a fork of *Block 0000 AB45*. If this fork becomes the longest, *TX131* would be considered valid, instead of *TX129*. The 50k would have gone to Tom, instead of Vinay and Kaylash. However, for this to happen the miners would have to outcompete the rest of the network which will extend the longest chain ending in *Block 0000 16AD*.

Hash-Rate. The network *hash-rate* (hashes per second) is a measure for the processing power of the Bitcoin network. The *target value* is adapted every 2016 blocks, to achieve a desired confirmation time. In 2017 the confirmation time was about 12 min [1].

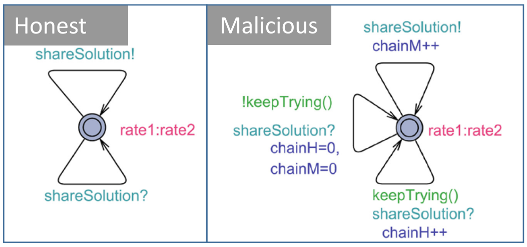
The hash-rate of a pool is relative to hash-rate of the network. A hash-rate of $r \in [0, 1]$, means that the pool alone would find 1 block in $12/r$ min. At the time of writing, November 2017, the largest pool *AntPool* had a hash-rate of 18%. These numbers are subject to fluctuation; in 2014 pool *Ghash.io* came close to a hash rate of 50% [5].

Andresen Attack. Andresen proposed that BTU miners could create a fork, mine it in secret until it reaches a length of 11, and then publish the fork at once [11]. Previously confirmed transactions in the honest fork would become unconfirmed, and Bitcoin miners would loose their reward for extending the previously longest fork. The aim is to undermine the trust in classic Bitcoin. The next sections investigate at what hash-rate of the malicious pool this attack becomes feasible.

3 Model and Strategies

The attack is modelled as a race between the honest pool and the malicious pool. This model is simplified by the fact the malicious pool will work exclusively on its fork, while the honest pools work exclusively on their fork, the publicly known blockchain. The model does not have to take into account network delays or concurrent mining on the same fork.

The model, to the right, contains two integer variables, `chainH` and `chainM`, to measure the length of the honest and malicious fork. Both pools announce solutions on channel `shareSolution`, with `rate1:rate2`.



If the malicious pools detects that the honest pool found a block it either continues the current attack, i.e. with the current fork, or abandons this attempt, and starts a new fork. Continuing is modelled as incrementing `chainH`. Starting a new fork is modelled by resetting `chainH` and `chainM` to zero.

The strategy deciding to continue or abandon an attack is defined by an array `int threshold[7]`. The race will continue if `chainH < threshold[chainM]`, and will be abandoned otherwise. The attack is successful if the malicious fork reaches the confirmation depth 6, **and** if at the same time the length of the malicious fork exceeds the length of the honest fork. This is expressed as the following Uppaal property:

```
Pr[<=1000000] (<> chainM>=6 and chainM>chainH)          (1)
```

4 Analysis

The analysis systematically sweeps through parameters for `int threshold[7]`. It considers values from 0 to 10, with a difference of at most 4 between `chainM` and `threshold[chainM]`. This leaves 12597 arrays to consider for an optimised strategy. The analysis computed the expected attack duration for hash-rates from 10% to 50% for each of these 12597 candidates. It identified an *optimised* strategy, defined by threshold array $[0, 2, 4, 5, 7, 8, 9]$, which is the rounded average of the top 1% of parameter values. This strategy means that the malicious pool should continue with the current attempt, even if it trails in the race, as long as it found a few blocks itself.

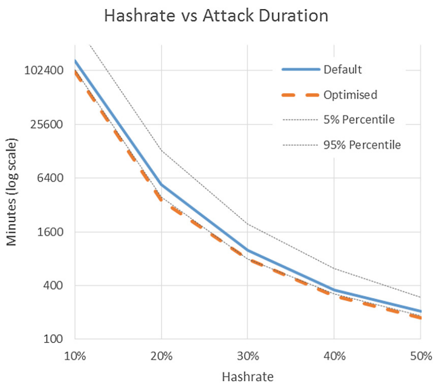


Fig. 3. Duration of an attack for different hash-rates for the default and optimised strategy.

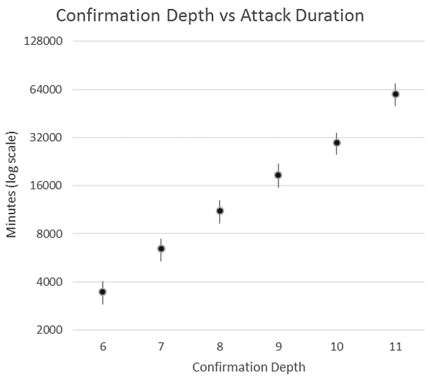


Fig. 4. Duration of attack for different confirmation depths. Results for 20% hash-rate.

Figure 3 compares the duration of the optimised strategy with the default strategy that abandons an attempt if the honest fork exceeds the malicious fork (array $[0, 1, 2, 3, 4, 5, 6]$). At a hash-rate of 10% the optimised strategy has an expected duration of 100203 min (69d:14h:03m), at hash-rate 20% this is 3582 min (2d:11h:42m). The default strategy is expected to take 134023 min (93d:1h:43m) and 5403 min (3d:18h:03m) respectively. At a hash-rate of 20% (or more) the Andresen attack appears to be feasible.

One counter measure would be to increase the confirmation depth. Figure 4 shows for a hash-rate of 20% that increasing the confirmation size by one increases the estimated duration on average by 77%. Note that the scale is logarithmic. For confirmation depth 10 this means that the expected duration is 29600 minutes (20d:10h:20m), and for depth 11 – as mentioned initially by Andresen – it increases to 60024 minutes (41d:16h:24m). Increasing confirmation depth increases the time to confirm transactions. A confirmation depth of 10 instead of 6, for example, means that it will cost 80% more time.

All results have been computed with Uppaal 4.1.19, at confidence level 0.99.

5 Conclusion

Analysis with Uppaal SMC of the Andersen attack shows that it does not require the majority hash-rate to succeed. A hash-rate of 20% would yield a success within a few days, sufficient to construct a malicious fork of length 6, something that would be unprecedented. Since the malicious pool tries to catch up from behind if it trails, the malicious fork will even have length 7 or more in 11% of the cases. Classic Bitcoin miners could adopt a larger confirmation depth, but that would affect the entire network and the efficiency of the currency as a whole. To mount a counterattack classic miners would actually have to adopt BTU, which is the declared aim of the initial attack.

The analysis does not depend on particularities of BTU. Similar considerations can be made for other coin forks. Bitcoin protocol developer Mark Corallo argued in [10] that an attack on the main fork is unlikely, given that resources could be better used for mining the main fork. Considering that some of the players have or are near the required hash-rate, and that optimising the strategy reduces the cost of an attack significantly, may change this economic argument. The cost of an attack may weigh up against long term benefits of setting a new standard, or short term benefits of intentionally upsetting the market.

Uppaal SMC's input language made it easy to model the attack, and its specification language was sufficiently expressive to analyse the model. Using these we were able to do a parameter sweep to optimise a strategy for the Andresen attack. The optimised strategy significantly reduces the expected duration of an attack, and thus resources required for it. The model, with additional analysis results are made available on http://wwwhome.ewi.utwente.nl/~fehnkera/V17/.

References

1. Bitcoin charts (2017). https://blockchain.info/charts
2. Andrychowicz, M., Dziembowski, S., Malinowski, D., Mazurek, Ł.: Modeling bitcoin contracts by timed automata. In: Legay, A., Bozga, M. (eds.) FORMATS 2014. LNCS, vol. 8711, pp. 7–22. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10512-3_2
3. Beukema, W.: Formalising the bitcoin protocol. In: 21th Twente Student Conference on IT (2014)

4. Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J.A., Felten, E.W.: Research perspectives and challenges for bitcoin and cryptocurrencies. IACR Cryptology ePrint Archive 2015, 261 (2015)
5. Cawrey, D.: Are 51% Attacks a Real Threat to Bitcoin? CoinDesk, June 2014. https://www.coindesk.com/51-attacks-real-threat-Bitcoin/
6. Chaudhary, K., Fehnker, A., van de Pol, J., Stoelinga, M.: Modeling and verification of the bitcoin protocol. In: MARS 2015. EPTCS (2015)
7. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B.: STTT. Uppaal SMC tutorial **17**(4), 397–415 (2015)
8. Herrmann, M.: Implementation, evaluation and detection of a doublespend-attack on Bitcoin. Master's thesis, Master Thesis ETH Zürich, April 2012
9. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2009). http://www.Bitcoin.org
10. Shin, L.: What Will Happen At The Time Of The Bitcoin Hard Fork? Forbes, October 2017. https://www.forbes.com/sites/laurashin/2017/10/31/what-will-happen-at-the-time-of-the-bitcoin-hard-fork/
11. Van Wirdum, A.: Bitcoin Unlimited Miners May Be Preparing a 51% Attack on Bitcoin. Bitcoin Magazine, March 2017. https://Bitcoinmagazine.com/articles/Bitcoin-unlimited-miners-may-be-preparing-51-attack-Bitcoin/