Discrete event simulation of different selfish mining strategies in Bitcoin

Master Thesis Proposal

Advisor: Edgar Weippl

June 29, 2017

1 Problem statement

The cryptocurrency Bitcoin started back in the year 2008 with the release of the Bitcoin white paper [13] and reached as of today a market capitalization of over 20 billion dollars [12]. Internally the Bitcoin cryptocurrency records all transactions in a public ledger called the blockchain. The blockchain is basically an immutable linked list of blocks where a block contains multiple transactions of the cryptocurrency. In Bitcoin, each block needs to contain a so-called proof of work (PoW) which is the solution to a costly and time-consuming cryptographic puzzle. Miners connected in a peer-to-peer network compete with their computation power to find solutions to the puzzle and hence to find the next block for the blockchain. Finding a block allows the miners to add a transaction to the block and gives them a certain amount of bitcoins out of thin air. Additionally, the grouping of the transactions in blocks creates a total order and hence makes it possible to prevent double-spending. After a block is found by a miner all other miners should adopt to this new tip of the chain and try to find a new block on top. This mining process is considered as incentive compatible as long as no single miner has more than 50% of the total computation power.

Eyal and Sirer showed that also miners under 50% have an incentive to not follow the protocol as described depending on their connectivity and share of computation power in the peer-to-peer network. By implementing a so-called selfish mining strategy a miner can obtain relatively more revenue than his actual proportion of computational power in the network. In general, the miner simply does not shares found blocks with the others and secretly mines on his own chain. At some point, the public miners will catch up with their public chain because they have over 50% of the computation power. In this case, the selfish miner publishes his blocks. If his chain is longer then the public chain, he is able to overwrite all blocks found by the honest miners. If the two chains have the same length the private miner also publishes his block and causes a block race. Now the network is split into two parts where one part is mining on the public tip and the other part is mining on the now public-private tip. In general, the selfish miner achieves that the other miners are wasting their computational power on blocks which will not end up in the longest chain.

Further research [14, 15, 10, 11, 1] explored different modifications of the original selfish mining algorithm by Eyal and Sirer and found slightly modifications of the algorithm which perform better under certain circumstances. For example, it could make sense for the selfish miner to even trail behind the public chain.

To prove the existence and attributes of selfish mining different approaches were applied. The researchers used simple probabilistic arguments [8, 1], numeric simulation of paths with state machines [10, 14], advanced Markov Decision Processes (MDP) [15, 11] or gave

results of closed-source simulations [8, 15]. Unfortunately, we cannot discuss the closed source simulations but all other methodologies do not use discrete event simulation (DES) [9] to simulate selfish mining. This type of simulation would allow reducing the abstraction used in the above-mentioned methodologies. One of the main abstraction introduced by the other researchers is that they model the block races with a simple probabilistic distribution. With a DES-simulation, it would be possible to model the whole network topology and simulate selfish mining under more realistic conditions. Hence the simulation could capture network delays and natural forks of the chain.

2 Expected result

The expected outcome of this thesis is a DES-simulation of different selfish mining strategies. The selfish mining strategies include:

- selfish mining [8]
- lead stubborn mining [14]
- trail stubborn mining [14]
- equal-fork stubborn mining [14]

For the simulation these strategies are combined with different distributions of computation power and different network topologies. The result of the simulations should show which strategy is the best strategy for a certain combination of a network topology and distribution of mining power.

Furthermore, the simulations should emphasise the recent work in the area of selfish mining and show that the current implementation of Bitcoin protocol is vulnerable against different selfish mining strategies.

3 Methodology and approach

First, the different strategies selfish, lead stubborn, trail stubborn and equal-fork stubborn mining from Nayak et al. and Eyal and Sirer need to be implemented. This is achieved by implementing a proxy which eclipses a normal Bitcoin client from the other nodes in the network. Now, if a block is found the proxy decides, depending on his selfish mining strategy, if a block should be transmitted from the eclipsed node to the rest of the network or vice versa. The design pattern proxy makes it possible to implement the selfish mining strategies without altering the reference implementation of Bitcoin and is therefore preferred over an implementation directly in the Bitcoin client.

In the next step, a DES-simulation program is implemented. To be able to control when a certain node finds a block all Bitcoin nodes should be executed in test mode. In test mode the real proof of work algorithm is disabled and every node accepts a command which lets the node create immediately a new block. With this functionality, it is possible to define a block discovery series which basically reflects the computation power of each node. The more blocks are found by a node the more simulated computation power the node has. Additionally to the block generation, the simulation program should also control the network topology and hence the connectivity of each node. For the simulation run, it is important that the connectivity of the nodes stays the same to make the results better comparable. This should be achieved by setting the connections from the nodes by the simulation program itself which is in contrast to normal behaviour. Normally Bitcoin nodes share their connections with other nodes over the Bitcoin protocol and try to improve the connectivity over time.

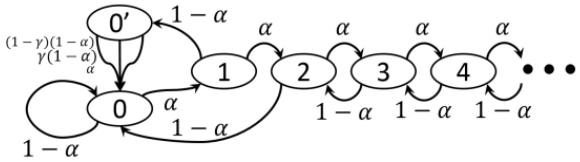

Figure 1: Selfish mining state machine with transition probabilities [8]

After the implementation of the selfish mining strategies and the DES-simulation program, the mining strategies are simulated. Different settings for the connectivity and distribution of computation power are used to compare the relative gain of the selfish mining strategies over the normal, honest mining.

4 State-of-the-art

Already in the year 2010 the user *ByteCoin* described the idea of selfish mining in the Bitcoin forum *bitcointalk* [6]. He provided simulation results of the attack which at that time was called *mining cartel attack*. Nevertheless, the discussions in the thread never caught fire and no further investigations or countermeasures were taken by the community [5, 1].

Later in 2014 Eyal and Sirer released the paper *"Majority is not enough: Bitcoin mining is vulnerable."* and coined the term selfish mining. The paper gives a formal description of selfish mining and proves how a miner can earn more than his fair share by conducting the attack. Figure 1 shows the attack as a state machine where α denotes the mining power share of the selfish miner. The labels of the states are representing the lead of the selfish miner over the public chain. Whenever the public network finds a block and the selfish miner publishes a competing block of the same height a block race occurs denoted with the state *0'*. In the case of such a block race, the variable γ expresses the probability of the selfish miner to win the block race. Hence γ part of the miners are mining on the public-private block and respectively $(1 - \gamma)$ are mining on the public block. The labels on the transitions are representing the transition probabilities between the states. The profitability of the simple strategy of [8] was proved by using probability calculations based on the state machine of figure 1. Furthermore, results of an undisclosed Bitcoin protocol simulator were given. In the simulation, 1000 miners with the same mining power were simulated and a fraction of these miners formed a pool which applied the selfish mining algorithm. In the case of a block race they artificially split the network where one part is mining on the public block and one part is mining on the block of the selfish pool.

Further research showed that more generalised selfish mining strategies lead to even more relative gain for the selfish miner [14, 15, 10, 11, 1]. Figure 2 shows a possible categorization of the different selfish mining strategies where α and γ is used equivalent as in figure 1 and β expresses $(1 - \alpha)$. Furthermore, the prime states are standing for states where a block race happens on certain height and the *0"* represent the state where both chains have the same height but the selfish miner and the rest of the network are mining on different branches. The idea behind the different variations of selfish mining strategies in figure 2 are:

- Lead stubborn mining strategy compromises the idea to cause as many block races as possible and to never overwrite the public chain with a longer chain. This strategy continuously tries to split the network to mine on different blocks and is therefore especially promising when the probability to win the block race is very high.

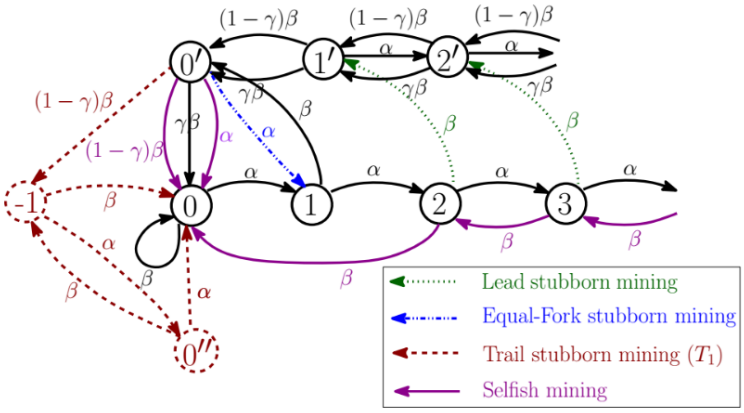

Figure 2: Categorization of different mining strategies [14]

- Equal-fork stubborn mining strategy changes the selfish mining strategy just by one transition. In case the selfish miner finds a block during a block race, he does not publish his block to win the race but he also keeps this block undisclosed to secretly mine on this new tip of the chain.

- Trail stubborn mining strategies reflects the idea to even trail behind the public chain and to eventually catch up. The figure 2 depicts the trail stubborn mining strategy *T1*. The number one denotes that the selfish miner is allowed to trail one block behind the public chain.

The strategy space for a selfish miner is practically endless and combinations of the aforementioned strategies are possible and are leading to even more relative gain [14, 15, 10, 11, 1].

To find the best strategy for a given mining power share α and connectivity γ researchers used different methodologies. [10, 14] used numeric simulations of paths in the state machine to find optimal selfish mining strategies. [15, 11] on the other hand used MDPs based on a state machine to find strategies with the most relative gain. The basic structure of the used state machines is for all publications the same. To further validate their results [8, 15] used a closed-source simulation.

Besides using variations of the selfish mining strategies, the attack can also be combined with other attacks to achieve better results [11, 15, 14, 10]. If the eclipse attack is used in combination with selfish mining the victim contributes its mining power to the private chain and hence, strengthens the position of the selfish miner [14, 11]. [14] additionally shows that the eclipsed victim under certain circumstances can benefit from the attack and therefore has no incentive to stop the attack. Another attack which can be used in combination with selfish mining is double-spending [15, 11]. Every time the selfish miner starts his selfish mining attack he can publish a transaction and include a conflicting transaction in his first secret block. During the execution of the selfish mining attack, the payment receiver may accept the payment depending on his block confirmation time. Now in the case of a successful selfish mining attempt, the adversary can overwrite the public chain, which additionally results in a successful double spending. The operational costs of unsuccessful double-spending can be seen as low because the adversary still would get goods or a service in exchange for the transaction [15, 11].

Last but not least also the prevention of selfish mining is part of the current work in selfish mining research [8, 2, 16, 19]. A backwards-compatible patch to mitigate selfish mining is uniform tie-breaking [8]. This means whenever a node receives two blocks of the same height he randomly select on of the blocks to mine on. [8] showed that this would

raise the profit threshold to 25% of the computational power and hence mitigating selfish mining. The drawback of this proposed change is that it would increase the connectivity of badly connected attackers to almost 50% with no actual effort for them. Uniform tie breaking is currently not implemented in Bitcoin []Another countermeasure foresees unforgeable timestamps to secure Bitcoin against selfish mining [2]. This countermeasure would make all pre-mined blocks of the selfish miner invalid after a certain amount of time. The implementation of this patch would require random beacons and hence introduce complexity and a new attack vector [2]. [19] proposes backward-compatible countermeasure by neglecting blocks that are not published in time and allows incorporation of competing blocks in the chain similar to Ethereum's uncle blocks [18]. This enables a new fork-resolving policy where a block always contributes to neither or both branches of the fork [19]. All of this mentioned countermeasures are not planned to be implemented or implemented in Bitcoin [3, 4]. Ethereum, the currently second largest cryptocurrency by market capitalization [12], has implemented uniform tie breaking as countermeasure against selfish mining [11, 17].

5 Relation to "Software Engineering and Internet Computing" curriculum

Bitcoin, the overlying topic, is a relatively new technology. Hence there are no concrete subjects or modules teaching this technology in the current curriculum. But under the hood Bitcoin technically is just a composition of different technologies which can be related to modules of the curriculum. First of all, Bitcoin is a software acting as a distributed system and can, therefore, be linked to the modules *Software Engineering* and *Distributed Systems*. Furthermore, Bitcoin uses cryptography to secure the system, which can be linked to the module *Advanced Security*. Hash functions are the main component of the PoW-algorithm in the mining process which helps to prevent double spending. Furthermore digital signatures based on cryptography are used to secure the bitcoins held by the different users of the system.

In the thesis, the implementation of a proxy enabling selfish mining strategies and a DES-simulation program are carried out. Since both of them are an implementation effort they can be directly linked to the module *Software Engineering*. Furthermore, both software programs are related directly to the module *Distributed Systems*. The proxy needs to be suited between multiple nodes in a Bitcoin network which is a distributed system and the simulation program needs to start-up, manage and tear-down this distributed system.

References

[1] L. Bahack. Theoretical bitcoin attacks with less than half of the computational power (draft). *arXiv preprint arXiv:1312.7013*, 2013.

[2] S. Billah. One weird trick to stop selfish miners: Fresh bitcoins, a solution for the honest miner. 2015.

[3] bitcoin. Bitcoin - reference implementation of the bitcoin protocol. https://github.com/bitcoin/bitcoin. Accessed: 2017-06-21.

[4] bitcoinbip. Bitcoin bips - bitcoin improvment proposals. https://github.com/bitcoin/bips. Accessed: 2017-06-21.

[5] BitcoinTalk. Thread about mining cartel attack on bitcointalk. https://bitcointalk.org/index.php?topic=2227.0. Accessed: 2017-06-21.

[6] ByteCoin. User bytecoin on the mining cartel attack. https://bitcointalk.org/index.php?topic=2227.msg30064#msg30064. Accessed: 2017-06-21.

[7] Coindesk. Bitcoin (btc) market capitalization - Coindesk. http://www.coindesk.com/data/bitcoin-market-capitalization/. Accessed: 2017-06-21.

[8] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *International Conference on Financial Cryptography and Data Security*, pages 436–454. Springer, 2014.

[9] G. S. Fishman. Principles of discrete event simulation.[book review]. 1978.

[10] A. Gervais, H. Ritzdorf, G. O. Karame, and S. Capkun. Tampering with the delivery of blocks and transactions in bitcoin. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 692–705. ACM, 2015.

[11] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 3–16. ACM, 2016.

[12] MarketCap. Bitcoin (btc) market capitalization - CryptoCurrency Market Capitalizations. https://coinmarketcap.com/currencies/bitcoin/. Accessed: 2017-06-21.

[13] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

[14] K. Nayak, S. Kumar, A. Miller, and E. Shi. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 305–320. IEEE, 2016.

[15] A. Sapirshtein, Y. Sompolinsky, and A. Zohar. Optimal selfish mining strategies in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 515–532. Springer, 2016.

[16] S. Solat and M. Potop-Butucaru. *ZeroBlock: Preventing Selfish Mining in Bitcoin*. PhD thesis, Sorbonne Universit es, UPMC University of Paris 6, 2016.

[17] unifromtiebreakingethereum. Release of uniform tie breaking in ethereum. https://github.com/ethereum/go-ethereum/commit/bcf565730b1816304947021080981245d084a930. Accessed: 2017-06-21.

[18] G. Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 151, 2014.

[19] R. Zhang and B. Preneel. Publish or perish: A backward-compatible defense against selfish mining in bitcoin. In *Cryptographers' Track at the RSA Conference*, pages 277–292. Springer, 2017.