Towards Detection of Selfish Mining Using Machine Learning

Matthew Peterson, Todd Andel and Ryan Benton
The University of South Alabama, Mobile, USA
mp1524@jagmail.southalabama.edu
tandel@southalabama.edu
rbenton@southalabama.edu

Abstract: Selfish mining is an attack against a blockchain where miners hide newly discovered blocks instead of publishing them to the rest of the network. The selfish miners continue to mine on their private chain while the honest miners waste resources mining on a shorter chain. According to the blockchain protocol, a longer chain takes precedent and shorter chains are discarded which allows the selfish miners to gain an advantage by keeping their chain secret. This attack can be used by malicious miners to earn a disproportionate share of the mining rewards or in conjunction with other attacks to steal money from cryptocurrency exchanges. Several of these attacks were launched in 2018 and 2019 with the attackers stealing as much as $18 Million. Developers made several different attempts to fix this issue, but the effectiveness of the fixes is currently unknown. Although this attack is possible against both Proof-of-Work and Proof-of-Stake blockchains, this research concentrates on detection in Proof-of-Work blockchains. As is difficult to evaluate security advances in the real-time blockchain, it is imperative to focus on simulation to evaluate blockchain security properties. To this end, we extend a blockchain simulator and add the ability to simulate selfish mining attacks. Several existing simulators are examined before choosing SimBlock for this research. Our goal is to identify the factors that identify selfish mining. Using existing research, we choose several factors that could identify an attack in an unlaunched state, an active state, or historically. We plan to use simulated data to train a machine learning model to detect selfish mining. Using the modified simulator, we generate training and test data for unlaunched and active attacks. For historical attacks, we will use historical data from known selfish mining attacks. While some existing research has examined the detection of selfish mining, it only examines active attacks. In this paper, we seek to lay the groundwork for future research into detecting attacks that are unlaunched, active, or historical.

Keywords: blockchain, selfish mining, proof-of-work, detection, simulation, machine learning

1. Introduction

Blockchains, at their core, are data structures that contain cryptographically-linked transactions. This data structure enables a peer-to-peer network to reach consensus about the state of transactions without requiring a central authority. Blockchain's first and most compelling application is that of digital currency. Although the beginnings of a blockchain-based digital currency already existed in the form of a white paper, the idea gained momentum when Bitcoin first appeared (Dai, no date). In order to make cryptocurrencies work, a mechanism is needed that enables network participants to reach consensus about the state of the blockchain. Each network participant keeps a copy of the ledger of transactions and without a consensus mechanism, one could compromise its integrity. Blockchains use many different consensus mechanisms, but the two most common ones are Proof-of-Work (PoW) and Proof-of-Stake (PoS).

In 2013 authors Eyal and Sirer discovered a vulnerability in the Proof-of-Work consensus algorithm that allowed malicious users to launch an attack against Bitcoin (Eyal and Sirer, 2018). Dubbed the selfish mining attack, this attack allows a miner, or a pool of miners, to earn more than their fair share of the mining rewards by withholding discovered blocks from the rest of the network. The attacker will publish these hidden blocks later to undo all the work of the other miners. Although other consensus algorithms are also vulnerable to selfish mining, this research focuses on the PoW algorithm. Selfish mining broke onto the national scene when it was used with double-spend attacks to steal cryptocurrency from cryptocurrency exchanges. In a double-spend attack, the attackers seek to spend their cryptocurrency twice by manipulating the protocol. The attackers will create two transactions where one transaction sells the cryptocurrency to an exchange while the second sends the same currency to an attacker-owned wallet.

These attacks started happening in 2018 and continued into 2019. In April 2018, the Verge cryptocurrency was attacked twice with the attacker getting away with as much as $2.8 Million (*A History of 51 Percent Attacks on Blockchains and Cryptocurrencies*, 2019). That same month, someone attacked Monacoin and stole another $90,000. The very next month the biggest attack to date happened to Bitcoin Gold when attackers used selfish mining and double-spending to steal $18 Million from cryptocurrency exchanges (Hertig, 2018). Following the Bitcoin Gold attack, Litecoin Cash was attacked in the same month, Zencash was attacked in June, and lastly,

Ethereum Classic was attacked in January of 2019 (Nesbitt, 2019; *A History of 51 Percent Attacks on Blockchains and Cryptocurrencies*, 2019).

Despite the growth of selfish mining attacks, there is limited research into their detection. Several authors have suggested ways to detect this attack but stopped short of doing any research into it (Göbel *et al.*, 2016; Eyal and Sirer, 2018). A few papers researched detection but only focused on a single factor or only worked on detection in a single stage (Chicarino *et al.*, 2020; Kędziora *et al.*, 2020; Liu *et al.*, 2020). Blockchain developers have released patches to prevent selfish mining, but without reliable detection, it is impossible to tell if these patches were successful. Detection of selfish mining is needed in the historical blockchain data as well as in the live network. Our goal is to examine all the factors that indicate a selfish mining attack and attempt to identify the significant ones by using machine learning. Selfish mining attacks exists in three distinct stages: unlaunched, active, and historical. Detection is the first and most important step in preventing an attack and we attempt to see if it is possible to detect an attack in all three of these stages. This paper lays the groundwork for further research into this topic.
In summary, our goals are as follows:

1. Determine the significant factors that identify an unlaunched selfish mining attack
2. Determine the significant factors that identify an active selfish mining attack
3. Determine the significant factors that identify a historical selfish mining attack from existing blockchain data

2. Literature Review

2.1 Proof-of-Work (PoW) Consensus Algorithm

To understand how selfish mining works, it is important to understand how mining works. Miners use a process called Proof-of-Work (PoW) to mine new blocks. Originally designed by Adam Black to counter denial-of-service attacks, PoW allows a peer-to-peer network to agree on the state of a blockchain (Back, 2002). At its core, a blockchain is a data structure of cryptographically-linked transactions. When someone wants to make a purchase using Bitcoin, he broadcasts the transaction to the network. Transactions are picked up by miners who bundle transactions together into blocks. Miners create blocks by taking the transactions and running them through a hashing algorithm until they find a hash that matches the network-defined target value.

This process of finding a valid hash is very computationally expensive and time-consuming. Miners are incentivized to create new blocks by receiving newly minted cryptocurrency when their block is accepted by the network.

The blockchain can fork in two when two miners discover a block at the same time. For example, if Miner A and Miner B both find a block and broadcast them to the network, Miner C will accept the block he hears about first. Blockchains resolve this issue with the principle of the longest chain.

The longest chain principle causes the network participants to adopt the longest available blockchain (the one with the most Proof-of-Work). If a miner is mining on a chain with a height of n and is notified about a valid chain with a height of $n + 1$, he will discard his current chain and switch to the new chain. This means the fork is resolved by whichever version of the blockchain finds the next block first (Nakamoto, 2008).

2.2 Selfish Mining

Selfish mining manipulates the way consensus algorithms work to gain a disproportionate share of the mining rewards. Under normal operation, miners work to discover a new block to add to the blockchain. Once found, this new block is broadcast to the rest of the network and each node adds it to their local blockchain. Selfish mining occurs when a miner keeps a discovered block hidden instead of letting the rest of the network know. On the surface, this seems counterintuitive. If the miner refuses to broadcast the block, they run the risk of another miner finding a block and rendering their work useless. If, however, a selfish miner discovers two or more blocks before the rest of the network, they cause the rest of the network to waste their time mining on a shorter chain. The minute the network gets close to the selfish miner's lead, they will release the longer chain on the network and reverse all the blocks mined by the honest miners using the longest chain principle discussed in the previous section.

The selfish mining attack was first discovered by Eyal and Sirer who showed that miners could attack a blockchain by keeping their mined blocks hidden. As shown in Table 1, Eyal and Sirer defined the parameter α as the percentage of the total hash rate controller by the selfish miners and parameter γ as the percentage of the honest nodes who adopt the selfish miner's block in the case of a network fork. Their papers showed that miners with $\alpha \leq 25\%$ and $\gamma = 50\%$ have enough monetary incentive to launch a selfish mining attack (Eyal and Sirer, 2018). Since this discovery, there have been multiple papers that examine different ways to fix selfish mining, extend the original selfish mining model, or dispute the proposal that selfish mining is incentive compatible.

2.3 Detecting Selfish Mining

Little research has been done in detecting selfish mining. In their original paper, Eyal and Sirer noted the difficulties facing accurate detection. They claimed that the two biggest indicators of selfish mining are the orphan block rate and the timings between successive blocks (Eyal and Sirer, 2018).

The orphan block rate is difficult to measure as the blockchain protocol discards orphaned blocks, so unless the orphaned blocks are captured in real-time, the data is lost forever. Another factor that makes detection difficult is the naturally occurring orphan block rate. Forks naturally occur in the blockchain and the protocol is designed to converge back to a single chain by orphaning one of the two chains. The difficulty is knowing when the orphaned blocks are part of an attack vs when they occurred naturally.

The natural orphan block rate is relatively low. Göbel, Krzesinski, and Taylor examined the orphan block rate in the absence of network delay and found that it hovers around 0.53 blocks per day in simulations and 2-3 blocks on the real Bitcoin network (Göbel *et al.*, 2016). They also examined the orphan block rate as it relates to selfish mining by simulating 70 days of blockchain activity for 1,000 nodes. They found that an increase in the number of blockchain forks suggested someone was engaged in selfish mining.

While this research showed several good selfish mining indicators, some factors make it difficult to implement with live network traffic. The only nodes who know of a blockchain fork are the nodes on the edge of the fork. If a node receives a blockchain from a peer that is the same length as his current blockchain, he will discard it. Similarly, if sends his blockchain to the same peer, that peer will discard the blockchain. This creates a natural dividing line through the peer-to-peer network where only the nodes on the line will know about the fork. For a node to discover a spike in the fork rate, it would have to be connected to every other node on the network or deploy multiple listener nodes.

Decker and Wattenhofer also examined the rate at which the Bitcoin network forked and thereby produced orphan blocks. After collecting 10,000 blocks, they found that the blockchain forked 169 times for a rate of 1.69% (Decker and Wattenhofer, 2013). This averages out to 2.4 orphan blocks per day which correlate with the findings by Göbel, Krzesinski, and Taylor (Göbel *et al.*, 2016). This number assumes that only one block was orphaned during the fork but multiple blocks can get orphaned at once. Additionally, they found that larger block sizes made the network more susceptible to forking due to the time spent validating the larger blocks.

The second indicator of selfish mining is the timings between successive blocks. If two or more blocks are discovered in quick succession, someone is likely to be engaged in selfish mining. The blocks in a PoW blockchain are exponentially distributed so the sudden deviation of finding two blocks close in time to each other is significant. While unlikely, it is worth noting that finding two blocks in quick succession is possible without a selfish mining attack.

One of the only papers to look at the block distribution was an unpublished paper by Fangyang Cui. This researcher graphed the distribution of Bitcoin blocks and found that the Bitcoin network follows the expected exponential distribution (Cui, 2015). This research is confirmed by another article that used the blockchain.info API. This author did not find any evidence of selfish mining but found that the blocks matched the expected exponential distribution.

The third indicator of selfish mining is the miner's average revenue-per-hour. Göbel, Krzesinski, and Taylor suggested that a drop in the revenue-per-hour of miners could indicate selfish mining but stopped short of proving their assumptions (Göbel *et al.*, 2016).

There are a few papers that examine how to detect selfish mining. The first, written by Chicarino, Albuquerque, Jesus, and Rocha, looked at detecting selfish mining using the block height of forks (Chicarino *et al.*, 2020). Using a simulator, they modeled a selfish mining attack and captured the output data. By examining the block forks, they determined that it was possible to detect an active blockchain attack based on the height of the forked chain. If the forked chain's block height was greater than or equal to two, they classified it as an attack. The authors conclude by stating that research is needed into reducing false positives from honest chains with a height greater than two.

Another paper examined different combinations of parameters and how they influenced the vulnerability of the blockchain network (Kędziora *et al.*, 2020). The parameters under study were the difference between the selfish miner's chain and the honest chain, the number of connections made by the selfish miners, and the total hashing power of the selfish miners. They examined how the choice of different parameters affected the distribution of block extraction between selfish and honest miners.

Lastly, Liu et al. examined the theoretical state space of a selfish mining attack by extending a Markov Chain simulation (Liu *et al.*, 2020). Like others, their research looked at an active attack and found that the number of attack states stayed mostly consistent regardless of the attacker's mining power.

3. Method

Our goal is to examine all the factors that identify a selfish mining attack and find significant ones. As we are unsure of what factors will prove significant, we plan on using a Random Forest machine learning algorithm and training it using simulated blockchain data. We view the detection of selfish mining as a classification problem (benign or selfish) and a Random Forest allows us to classify the data and see the feature importance. Using this simulated data, we plan on seeing if the significant factors can detect an attack in all three stages.

3.1 Simulating blockchain data

To train the model, we plan on simulating 70 days' of blockchain data by using SimBlock. SimBlock is a peer-reviewed blockchain simulator written in Java (Aoki *et al.*, 2019). To generate the correct data, SimBlock needed modification to simulate selfish mining. We made several modifications to the program to add selfish mining nodes to the simulated network a copy is available on our GitHub upon request (*themattman18/simblock*, no date). The modifications to SimBlock followed the guidelines from the original selfish mining paper by Eyal and Sirer for the selfish miner's hash rate, and block propagation.

Table 1: Table of Notations

Symbol	Description
s	The selfish miners who are attacking the network
h	The honest miners who are following the Bitcoin mining protocol
α	The percentage of the total network hash rate owned by the selfish miners
γ	The percentage of honest nodes who adopt the selfish miner's block in the event of a network fork

The newly added selfish mining node follows Eyal and Sirer's original selfish mine algorithm. The algorithm defines the following states for selfish miners (*s*) and honest miners (*h*):

- *s* gains a one-block lead → *s* keeps block secret
- *s* has a one-block lead but *h* finds block → *s* releases secret block, causing a fork
- *s* has no lead and *h* finds block → *s* adopts *h* block
- *s* has a two-block lead and *h* finds block → *s* releases their longer chain onto the network
- *s* has a lead > two blocks and *h* finds a new block → *s* does nothing

If *s* has a one-block lead and *h* discovers a new block, *s* will release their block onto the network to intentionally create a fork. The miners who hear about *s* block first will adopt it while those who hear about *h* block first will adopt that one. Selfish mining becomes profitable when the attackers can get ½ of the network to adopt their block in this situation (Eyal and Sirer, 2018). The number of nodes who adopt *s* block is represented by the symbol γ and the nodes who adopt *h* block is (1- γ).

To launch a successful selfish mining attack, the selfish miners need to get ½ of the network to adopt their block in the case of a blockchain fork. The SimBlock program connects nodes based on the observed distribution of connections in Bitcoin in the year 2015 taken from previous research (Gervais *et al.*, 2016; Miller *et al.*, no date).

If a simulation node is randomly selected into a region where the nodes were well connected, the simulation will connect that node to a larger number of other nodes. The better connected a selfish mining node is, the higher γ will end up being.

The total hash rate of the selfish miners is indicated by α with honest nodes controlling (1 – α) of the hash rate. As discussed previously, if the blockchain forks due to competing blocks between the selfish and honest miners, the selfish miners become more profitable as a higher percentage of honest nodes accept and propagate their block. If γ = 0.5 (½ of the honest nodes accept the selfish miner's block) then the selfish miners only need 25% of the network hash rate to successfully launch a selfish mining attack (α = 0.25). We modified SimBlock and gave the newly created selfish mining node 25% of the total network mining power. Although a selfish mining attack can succeed between the ranges 0.5 > α > 0 we went with the threshold of α = 0.25 as defined by Eyal and Sirer (Eyal and Sirer, 2018).

3.2 Generation of Training and Test Data

To generate the training data, we simulated 70 days' worth of blockchain data for 600 network nodes. Seventy days of data is the generally accepted amount of time for gathering data.

Our test data comes from two different places. The first set of test data comes from the modified SimBlock network simulator. We generated 30 days of data and categorized it into benign and selfish and plan on using it to verify the accuracy of the model. The second set of data, used to test the ability to detect historical attacks, will come from the Bitcoin Gold blockchain. Since blockchain data is not stored in a human-readable format, a program was needed to format the data into the input needed by the machine learning model. Bitcoin Gold hosts a block explorer that allows anyone to browse the latest released block on the network (*Bitcoin Gold Block Explorer*, no date). This website presents a way to view the latest blockchain data and allows users to jump to a date to view the recorded blocks on that day.

Two separate periods were gathered from the block explorer website. The first benign set of data was gathered from 5/15/2018 approximately three days before the first known selfish mining attack. 155 blocks were gathered from block height 529040 to 529194. The second set of data came from a selfish mining attack on 5/18/2018-5/19/2018. We gathered a total of 25 blocks ranging from block height 529022 to 529047.

3.3 Feature Selection

The first feature we selected is the block height. As shown by Chicarino, Albuquerque, Jesus, and Rocha, the block height of a fork can be used to detect a launched selfish mining attack (Chicarino *et al.*, 2020). If a blockchain fork had a block height greater than or equal to two, they classified it as an attack. Unfortunately, this resulted in some false negatives from forks that only had one block and it cannot detect historical or unlaunched selfish mining attacks.

The second feature is the timing between blocks. The discovery of new blocks is Poisson distributed which means that the time it takes to discover a new block is independent of when the last block was found. Although the act of finding a new block is an independent variable, the average time it takes is kept at 10 minutes per block. This exponential distribution makes it difficult to tell if two blocks released back-to-back were selfishly mined or just part of the normal mining process but is a useful feature.

Multiple papers have discussed the importance of the fork rate on the detection of selfish mining (Göbel *et al.*, 2016; Eyal and Sirer, 2018). Forks create a natural rift in the network where only the nodes on the edge of this rift know about the fork and detecting all forks would require connecting to all the network nodes. To bypass this limitation, we ignore this restriction in our simulation and report all forks.

The transaction count is another feature we selected. Decker and Wattenhofer measured the effects of block size on propagation speed and noticed a strong correlation between the two (Decker and Wattenhofer, 2013). Smaller blocks propagate through the network faster than larger ones and a selfish miner wants to propagate their blocks as fast as they can to ensure that most of the nodes build on top of theirs. Honest nodes, on the other hand, want to include as many transactions as possible to gain more money from transaction fees. This could result in a large disparity between the transaction counts in honestly mined blocks and the selfishly mined blocks.

The next feature we selected is the transaction output wallet address. An active selfish mining attack could double-spend cryptocurrency and the attackers will send the double-spent money back to a wallet address that they control. While it is best practice to create a new wallet address for every new transaction, it is not required. A selfish miner could continue to selfishly mine and send transactions to the same wallet address.

Another feature we selected is the coinbase transaction which sends the reward for mining a new block to the miner-specified wallet address. Regardless of double-spent transactions, attackers want to receive the financial reward for mining a new block. Just like in the other transactions, the miner can supply a new wallet address for every new coinbase transaction. However, in some observed selfish mining attacks, the attacker repeatedly sent the coinbase transaction to the same wallet address (*Bitcoin Blocks solved On 2018-05-19 | Insight*, no date).

Göbel, Krzesinski, and Taylor observed the importance of the revenue-per-hour as an indicator and calculated it as (*block rewards / percentage of the network hashing power*) (Göbel *et al.*, 2016). When selfish mining increases the honest node's revenue-per-hour will drop. As implied by the name, using the revenue per hour is a slow detection process because it works on the hourly average. The total, real network mining power can fluctuate but the difficulty itself is only adjusted once every two weeks which makes it difficult to know how much mining power there is on a network at any given point.

Using these features, we train the machine learning model to see if it is possible to detect selfish mining. Future research will examine the effectiveness of these features.

4. Review

In this paper, we lay the groundwork for building a machine learning model that can be used to detect unreleased, active, and historical selfish mining attacks. We extended the SimBlock blockchain simulator by adding the ability to simulate selfish mining attacks. By using the updated SimBlock, we generated training and test data for selfish mining attacks which we plan on using to train our machine learning model. The test data for the historical attacks come from examples of actual selfish mining attacks that were launched against the Bitcoin Gold blockchain. The next step is to finishing training the machine learning model and measure its effectiveness.

References

A History of 51 Percent Attacks on Blockchains and Cryptocurrencies (2019) *Blocks Decoded*. Available at: https://blocksdecoded.com/history-51-percent-attacks/ (Accessed: 26 June 2019).

Aoki, Y. *et al.* (2019) 'SimBlock: A Blockchain Network Simulator', in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 325–329. doi:10.1109/INFCOMW.2019.8845253.

Back, A. (2002) 'Hashcash - A Denial of Service Counter-Measure', p. 10.

Bitcoin Blocks solved On 2018-05-19 | Insight (no date). Available at: https://explorer.bitcoingold.org/insight/blocks-date/2018-05-19 (Accessed: 19 April 2020).

Bitcoin Gold Block Explorer (no date). Available at: https://explorer.bitcoingold.org/insight/ (Accessed: 19 April 2020).

Chicarino, V. *et al.* (2020) 'On the detection of selfish mining and stalker attacks in blockchain networks', *Annals of Telecommunications*, 75(3–4), pp. 143–152. doi:10.1007/s12243-019-00746-2.

Cui, F. (2015) 'Detecting Selfish Mining in Bitcoin', *unpublished*, p. 4.

Dai, W. (no date) *b-money*. Available at: http://www.weidai.com/bmoney.txt (Accessed: 15 February 2020).

Decker, C. and Wattenhofer, R. (2013) 'Information propagation in the Bitcoin network', in *IEEE P2P 2013 Proceedings*. *2013 IEEE Thirteenth International Conference on Peer-to-Peer Computing (P2P)*, Trento, Italy: IEEE, pp. 1–10. doi:10.1109/P2P.2013.6688704.

Eyal, I. and Sirer, E.G. (2018) 'Bitcoin Mining Is Vulnerable', *Communications of the ACM*, 61(7), p. 8.

Gervais, A. *et al.* (2016) 'On the Security and Performance of Proof of Work Blockchains', in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16*. *the 2016 ACM SIGSAC Conference*, Vienna, Austria: ACM Press, pp. 3–16. doi:10.1145/2976749.2978341.

Göbel, J. *et al.* (2016) 'Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay', *Performance Evaluation*, 104, pp. 23–41. doi:10.1016/j.peva.2016.07.001.

Hertig, A. (2018) *Blockchain's Once-Feared 51% Attack Is Now Becoming Regular*, *CoinDesk*. Available at: https://www.coindesk.com/blockchains-feared-51-attack-now-becoming-regular (Accessed: 24 February 2019).

Kędziora, M. *et al.* (2020) 'Analysis of Blockchain Selfish Mining Attacks', in Borzemski, L., Świątek, J., and Wilimowska, Z. (eds) *Information Systems Architecture and Technology: Proceedings of 40th Anniversary International Conference on Information Systems Architecture and Technology – ISAT 2019*. Cham: Springer International Publishing (Advances in Intelligent Systems and Computing), pp. 231–240. doi:10.1007/978-3-030-30440-9_22.

Liu, Z. *et al.* (2020) 'A Security Detection Model for Selfish Mining Attack', in Zheng, Z. et al. (eds) *Blockchain and Trustworthy Systems*. Singapore: Springer Singapore (Communications in Computer and Information Science), pp. 185–195. doi:10.1007/978-981-15-2777-7_16.

Miller, A. *et al.* (no date) 'Discovering Bitcoin's Public Topology and Influential Nodes', p. 17.

Nakamoto, S. (2008) 'Bitcoin: A Peer-to-Peer Electronic Cash System', p. 9.

Nesbitt, M. (2019) *Ethereum Classic (ETC) is currently being 51% attacked*, *The Coinbase Blog*. Available at: https://blog.coinbase.com/ethereum-classic-etc-is-currently-being-51-attacked-33be13ce32de (Accessed: 24 February 2019).

themattman18/simblock (no date) *GitHub*. Available at: https://github.com/themattman18/simblock (Accessed: 13 April 2020).