Pooled Mining Makes Selfish Mining Tricky

Suhyeon Lee[1,2] and Seungjoo Kim[1]

[1]CIST*, Korea University, Korea
[2]Agency for Defense Development, Korea
[1]{*orion-alpha, skim71*}*@korea.ac.kr*
[2]*korea@add.re.kr*

December 22, 2018

Abstract

Bitcoin, the first successful cryptocurrency, uses the blockchain structure and PoW mechanism to generate blocks. PoW makes an adversary difficult to control the network until she retains over 50% of the hashrate of the total network. Another cryptocurrency, Ethereum, also uses this mechanism and it did not make problem before. In PoW research, however, several attack strategies are studied. In this paper, we researched selfish mining in the pooled mining environment and found the pooled mining exposes mining information of the block which adversary is mining to the random miners. Using this leaked information, other miners can exploit the selfish miner. At the same time, the adversary loses revenue than when she does honest mining. Because of the existence of our counter method, the adversary with pooled mining cannot do selfish mining easily on Bitcoin or blockchains using PoW.

1 Introduction

Bitcoin suggested by Nakamoto is a decentralized ledger in 2008 [10]. The decentralized ledger for recording the transmission of cryptocurrency, e-cash in [10], is recorded in the form of a block in the ledger by a specific rule. Each block is configured so that it can not be manipulated in the middle by referring to the hash value of the previous block. Proof-of-Work(PoW) mechanism was adopted as a block generation rule for its security. Miners can earn revenue proportional to their hashrate by the law of large numbers (LLN). An adversary needs over 50% of the hashrate of the total Bitcoin network to control block generation. The fact that Bitcoin and Ethereum, which use the PoW mechanism, did not

*Center for Information Security Technologies

experience a big crisis in block generation demonstrates that the PoW works as intended.

For several years, PoW mechanism has met some challenges in security research as well as its drastic resource consumption problem. Research has found several cheating methods against honest participants. Especially, in this paper, we will take note of two severe attacks among these. One is selfish mining, and the other is block withholding attack. Selfish mining makes an adversary who has over 25% - this value depends on the network environment - can get revenue over its hashrate proportion to the whole network. The existence of the selfish mining not only means that it is unfair to solve PoW puzzles but also is a severe flaw in integrity of blockchain. Block Withholding attack between pools makes a malicious pool can earn extra reward than its honest mining by joining to other pools. It is not severe as much as selfish mining, however, it makes PoW pooled mining which is the dominant approach to miners unfair. At the same time, defense methodology has been analyzed against the attacks against PoW. Defense research about selfish mining suggested countermeasures in order for that attackers not to keep secret blocks during long time. Most of them, however, need the drastic changes in Bitcoin protocol itself. Also, Defense research against block withholding attack proposed not only changes in Bitcoin protocol itself but also mining pool policies or algorithms.

In this paper, we propose a counter strategy scheme for miners against selfish mining. In contrast that previous researches need to change the original Bitcoin protocol, our method does not need any change in the protocol.

This paper is organized as follows. An overview of Bitcoin block structure will be given in Section 2. Our model of pooled mining is described in Section 2.2 and we demonstrate that other mining pools can creat a PoW task with the leaked information. Based on the model, we propose a counter mining strategy against selfish mining and show the simulation result in Section 4. Our model and strategies are evaluated and analyzed from various perspectives in Section 5.

1.1 Contributions

- We propose and investigate a countermeasure against selfish mining in the pooled mining environment. This method can be applied without any change in Bitcoin protocol. Our simulation result shows that the selfish miner get much less revenue than honest mining if 15% of other miners use this strategy. And the miners using our method always get much high extra reward.

- We show that, in view of our simulation, it is too tricky to do selfish mining for mining pools. Consequently, selfish mining will not happen in the situation that most of the miners are joining in mining pools.

1.2 Related Works

Network-based Attack Bitcoin has attack vectors on decentralized network structure. Sybil attack is modifying reputation in peer-to-peer networks by forging identities [4]. An attacker can implement it by filling the network with clients she controls. Then most of the nodes in the network inevitably connect with the attacker's nodes. Bitcoin prevents it efficiently with the Proof-of-Work mechanism by demanding over 50% of the total network.

Eclipse attack happens if all nodes connecting to the victim are under the control of the attacker [8]. Since the attacker isolates the victim, the attacker can filter all of the input and output of the network to the victim. For example, the attacker can make the victim waste his computing power by hiding blocks published in the network.

The double-spend attack is not an attack by itself, but it is rather a goal to achieve by other attacks. This attack is on the network or more higher application level. When an attacker gains control over the network, it confuses the network to accept a new chain that is not issued before. It causes the existing transaction to revert and causes a new transaction [13]. To prevent this attack, it is recommended to accept the contents of the block in which there are enough descendent blocks already issued.

Mining-based Attack Selfish mining proposed by Eyal and Sirer [6] is the infamous attack strategy in PoW mining. The attacker with big hashrate over 25% has enough to keep own private chain in secret. To be concrete, if others find a block faster than the attacker, she accepts the block. Only if the attacker finds a block faster than others, she does not publish it and keep mining as her private chain. When she cannot keep her chain longer than the public chain, she publishes her chain, then it makes blocks on public chain invalid as they belong to the shorter chain. This attack has a relative advantage by making the efforts of other diggers in vain. In result, the attacker gains much higher Bitcoin reward than her proportion of total network hashrate. It means revenue is incompatible with their work in Bitcoin. Also, this attack occurs fork frequently so that integrity of Bitcoin decreases. In Eyal [6], when other miners mine any chains fairly in a competition situation of two chains, the attacker needs over 25% of the total network hashrate to make this attack profitable. Other researches make this attack optimize. Stubborn attack [11] combined the selfish mining with network conditions to form more efficient attack scenarios. Optimal selfish mining strategies [16] construct dynamic selfish mining strategy depends on attacker's hashrate proportion to Bitcoin network.

The basic idea of the block withholding attack is that it does not submit a PoW block to the participating mining pool. Rosenfeld [15] proposed the notion about it at first. Eyal advanced this attack a viable strategy between mining pools in his paper [5]. Its strategy is to transfer a portion of the attacker's mining power to the other mining pool, and then it withholds the submission of valid blocks to take some relative advantage. Since mining pools share Bitcoin reward to miners by each miner's partial PoW submission, withholding

valid blocks does not devalue the block withholding miner's contribution. That is, the block withholding miners get free share without actual contribution to mining pools. Since block withholding reduces the expected rewards of the participating mining pools, the actual reward increment of this attack is due to the difficulty of the bitcoin network by not submitting blocks. Unlike selfish mining, this attack does not need big enough computing power. He showed that there is a Nash equilibrium in which two mining pools mutually interfere causes loss to each other. Kwon [9] proposed a more efficient variation, and there is no Nash equilibrium when using this technique.

Defense Research Against selfish mining, research has proposed various methods in order to limit private chains which are kept intentionally unpublished during long time. All of these methods need big changes in Bitcoin protocol. In order to decide if someone keeps blocks before publishing, Solat [17] and Zhang [19] proposed methods to compare block generation time and block publishing time. In their methods, if the selfish miner publishes her blocks kept over limited time, other nodes in the network does not accept the blocks. Fruitchain proposed by Pass [12] does not use the time parameter directly. It uses two-in-one PoW protocol which generate a special solution, fruit, additionally. The priority is not only depends on the height of chains but also depends on fruit references. The published chain has more big possibility to get fruits than private chain of the attacker. Thus, it makes the selfish miner get priority harder. Pass proved its close-to-optimal fairness in incentive. Another approach is treating orphan blocks which are not accepted finally. For security of high rate PoW, GHOST protocol which gives priority to the heavier chain including orphan blocks is proposed by sompolinsky and Zohar [18]. One of the biggest cryptocurrencies, Ethereum employs GHOST protocol [2]. There was, however, no theoritical basis that honest miners generate heavier chains in the paper. The simulation in Ritz [14] shows GHOST protocol is still vulnerable to the selfish mining.

2 Preliminary

2.1 Bitcoin

Bitcoin is the cryptocurrency which uses blockchain structure suggested by Nakamoto [10]. The Bitcoin is for transmitting e-cash in decentralized networks. For this purpose, its blockchain is a set of blocks that contains records of transactions of Bitcoin. Each block can be published when a miner solves the mathematical puzzle. This puzzle is Proof-of-Work(PoW) which proves a specific workload of the miner. An amount of the workload, called as **difficulty**, is adjusted to make Bitcoin network generate a block every 15 minutes. This adjustment refers to the average block generation rate of the previous 1000 blocks. It takes about two weeks to generate 1000 block every minute.

The block consists of two parts, header and transaction as shown in Figure 1 [1].

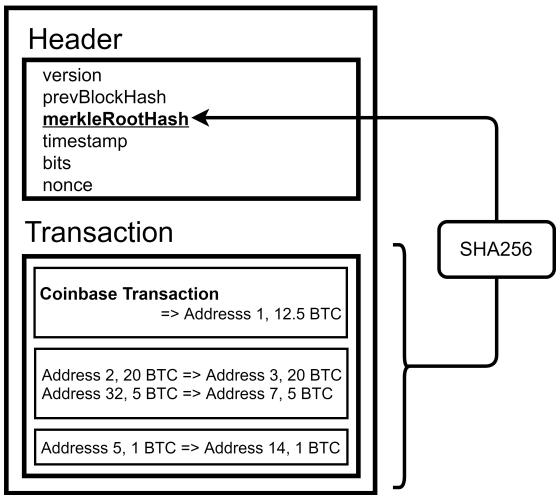

Figure 1: Bitcoin block structure

The transaction part contains records bitcoin transactions with the sender, receiver and the amount of Bitcoin. Among them, the first transaction is a special transaction which is called **Coinbase transaction**. This transaction is to give Bitcoin revenue to the miner who publishes a valid block in Bitcoin network. Because **Coinbase transaction** is a process of creating Bitcoin for miners, it does not contain a sender of the transaction.

The header part contains six elements. **version** is the current version of the Bitcoin protocol. **prevBlockHash** is the hash value of the previous Bitcoin block in order. By containing the hash value of the previous block header, the next block keeps the immutability of the context. **merkleRootHash** is the hash value of the transaction part. So this value keeps the immutability of the whole transactions in the block. **timestamp** is the time when the block created. **nonce** is the value to solve the PoW puzzle of Bitcoin.

To publish Bitcoin block, miners should find **nonce** which satisfies difficulty condition. $H(\cdot)$ is a cryptographic hash function SHA256. It can be semiformally described by applying the description in Garay [7].

$$(H(v, r, H(m), st, T, ctr) < T) \wedge (ctr \leq q) \tag{2.1}$$

where

v is the version of Bitcoin

r is prevBlockHash

m is a list of transactions

$H(m)$ is merkleRootHash

st is timestamp

T is the current difficulty target of Bitcoin network

ctr is nonce

q is the size of the nonce value

2.2 Pooled Mining

Anyone can join the bitcoin network and do mining blocks by solving the PoW puzzle. However, since the average amount of computation required for bitcoin mining is generally so high that individuals can not afford it, constructing mining pools is a dominant approach as shown in Figure 3.

We choose a model of mining pool in Stratum mining protocol [3] which is a text based communication protocol for mining pools. Figure 2 illustrates miners in mining pools. As we mentioned above, miners rarely find a valid block during a long time, though they work hard in reality. Thus it needs a new indicator in order to measure the contribution of each miner. Let D be this indicator. This value is bigger than **difficulty** of the network. In other words, it is easier to satisfy this value. Thus, miners solve the easier PoW puzzles and measure their contribution according to this. Some PoWs meet **difficulty** is distinguished by the pool manager and issued to the network as a valid block. Namely, a block meets difficulty is referred to full Proof-of-Work(fPoW) and a block meets the indicator of the pool is referred to partial Proof-of-Work(pPoW).

So, for miners, their PoW task w consists with v, r, m, st, q except ctr which is a nonce to find. Their mining pools distribute PoW task as string 2.2. Miners should find nonces which meet the inequality 2.3.

$$w = (v, r, m, st, q) \tag{2.2}$$

$$(H(v, r, H(m), st, T, ctr) < D) \wedge (ctr \leq q) \tag{2.3}$$

where

D is the difficulty value for share in a mining pool.

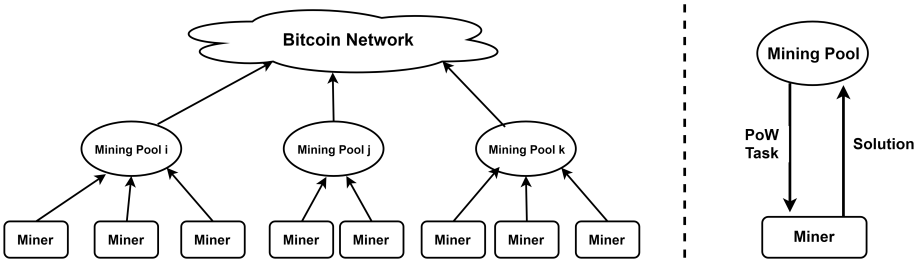

Figure 2: Bitcoin Mining Pool

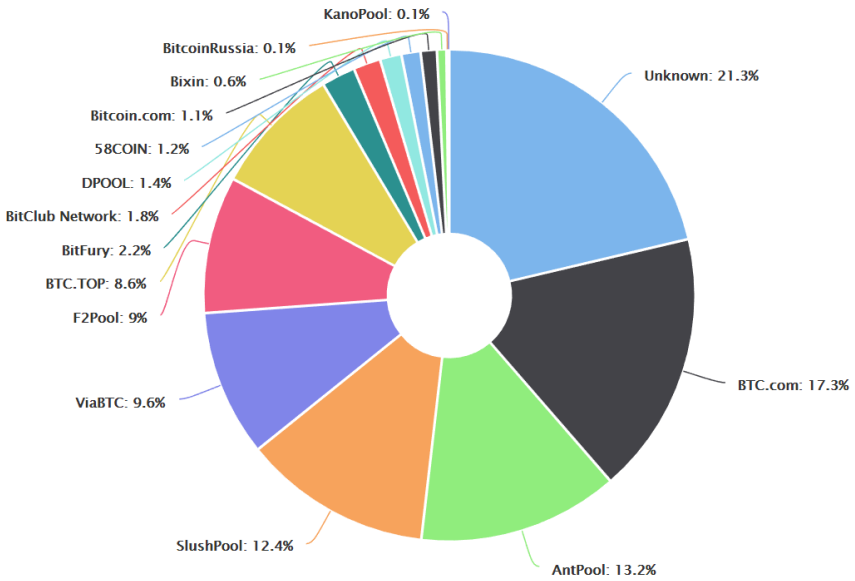

Figure 3: Hashrate distribution by Blockchain.com

3 Model

Before defining the model, we briefly mention the role of miners, miners in mining pools, and mining pools. We assume that every miner can join every mining pool. Besides, we use a modifier called 'honest' in subjects that follow the above in order to distinguish them from attacking or cheating subjects.

- Miners: Miners maintain a recent Bitcoin blockchain. They do mining a new block, fPoW, in the longest chain.
- Miners in mining pools: Miners select a mining pool which they join in. They send blocks, pPoW, to the pool as the task from the pool.
- Mining pools: Mining pools create a task to mine and distribute it to miners who are joining in. Mining pools publish valid blocks, fPoW, in the middle of blocks, pPoW, given by miners and get Bitcoin reward. Mining pools share Bitcoin reward to miners by the proportion of their pPoW submission.

3.1 Model of Miners

Algorithm 1 describes a general Bitcoin miner. It creates a PoW task for Bitcoin network conditions. Subsequently, the miner assigns a random value *ctr* to find the valid block. If a valid block is obtained, the miner issues to the network and Bitcoin compensation is obtained as the *coinbase* transaction.

Algorithm 1 Miner M_i

```
1: w: PoW task
2: T: the difficulty value of the current network
3: share: miner's total share of a mining pool
4:
5: procedure MINING
6:     reward ← 0
7:     w ← CreateTask(v, r, H(m), st, T, q)
8:
9:     while ¬isOtherPublished do
10:        ctr ← random()
11:        if (H(ctr ‖ w) < T) ∧ (ctr ≤ q) then
12:            reward ← reward + publish(ctr ‖ w)
```

3.2 Model of Mining Pools

Algorithm 2 shows the algorithm of an honest miner. At first, the miner selects a pool to join and work. Then he gets PoW task and solves the PoW puzzle. It is calculated by substituting a random ctr repeatedly to find the nonce satisfying the formula 2.1. He gets the share proportional to its valid PoW task. Whenever he wants to change his mining pool to mine, he can stop the mining procedure and start a new mining procedure to work at a new pool.

Algorithm 2 Miner M_i in a mining pool

```
1: procedure MINING
2:     selectPool(i)
3:
4:     while keepMiningPool do
5:         w ← getNewTask()
6:         while keepTask do
7:             ctr ← random()
8:             if (H(ctr ‖ w) < D) ∧ (ctr ≤ q) then
9:                 send(ctr ‖ w)
10:        share ← share + recv(i)
```

Algorithm 3 shows the algorithm of an honest mining pool. The mining pool creates a new task by proper element including its coinbase transaction. It distributes the PoW task to every miner who joins it. Then the pool collects pPoW solutions from miners and publishes a block if a solution meets the difficulty condition(Formula 2.1). The manager shares Bitcoin reward to miners proportional to their pPoW contribution.

Algorithm 3 Mining Pool P_i

List $workers$: registered miners
List $jobRate$: amount of submitted pPoW

procedure MANAGE POOLED MINING
    $reward \leftarrow 0$
    $w \leftarrow CreateTask(v, r, m, st, T, D, q)$
    for each $a \in \mathcal{W}orkers$ **do**
        $send(a, w)$
    for each $a \in \mathcal{W}orkers$ **do**
        $(ctr \parallel w) \leftarrow recv(a)$
        if $(H(ctr \parallel w) < T) \wedge (ctr \leq q)$ **then**
            $reward \leftarrow reward + publish(ctr \parallel w)$
        $jobRate[a].add()$
    $totalPoW \leftarrow$ each $jobRate(a)$
    for each $a \in \mathcal{W}orkers$ **do**
        $pay(a, reward \times \frac{jobRate(a)}{totalPoW})$

4 Strategy Against Selfish Mining Pool

4.1 Selfish Mining

The model of selfish mining follows exactly same mechanism suggested in Eyal [6]. To clarify our description, we must define terms and relevant states before. The basic idea of selfish mining is that the adversary does not publish valid blocks to make others waste their mining on the already solved problem. The blocks which the adversary keeps in secret is referred to a private chain. We use the word *lead* as when an adversary who does selfish mining has a longer private chain. *lead* is only used when the attacker's chain is ahead, but a negative lead is also used for an optimal strategy. When the adversary does not lead, there are two situations of the same length, the state 0 and the other state 0'. The former means that the adversary has the same chain of the public chain. So they are working on the same block. The latter state, 0', means they have different chains which have the same length.
where

α is hashrate of the selfish miner

γ is a proportion of the miners who mines on selfish miner's chain during fork.

Below items describe selfish miner's states by situations.

- State 0: The adversary mines the block on the public chain with the highest height. If she finds a block, she gets one lead by not publishing it. If the others find a block, she accepts this block, and the state is still 0.

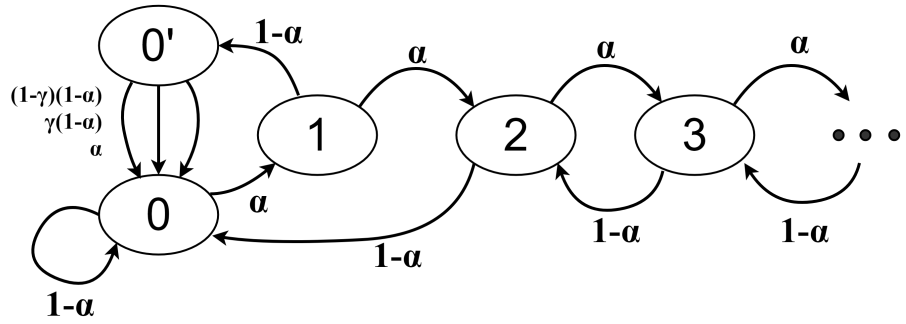

Figure 4: State Machine of Selfish Mining

- State 0': This state happens when the other miners find a block and the adversary had mined her block unpublished on lead 1. Then she publishes her block to compete with the block published by others. All miners except the adversary freely does mining among two chains and one of two chains wins.
- State 1: If she finds a block, she gets two lead by not publishing it. If the others find a block, she published her block and competed on the next block.
- State 2: If adversary finds a block, she gets three lead. If the others find a block, she published her two blocks to get direct rewards from the network.
- State n bigger than 2: If adversary finds a block, she gets n+1 lead. If the others find a block, she gets n-1 lead.

4.2 Selfish mining detection

To summarize the selfish mining procedure in the previous subsection, the selfish miner tries to keep its private chain longer than the public chain as much as possible. In this context, we assume the selfish miner is a mining pool which distributes PoW task including various information. We can collect PoW task from mining pools and it is not difficult to collect PoW task of mining pools. For example, a manager of one mining pool can join a suspicious mining pool without mining in order to get information. Here, as shown in Figure 2, we need to pay attention to the facts that the hash value of the previous block and prevBlockHash are open to the public. And, by using these information, we can decide for which block the mining pool(selfish miner) is working. If a mining pool is innocent, it does mining on the last block of the longest chain published. But, if a mining pool is selfish, it does mining with an unknown previous block. Therefore, we detect a mining pool which mine on an unknown previous block as a selfish miner.

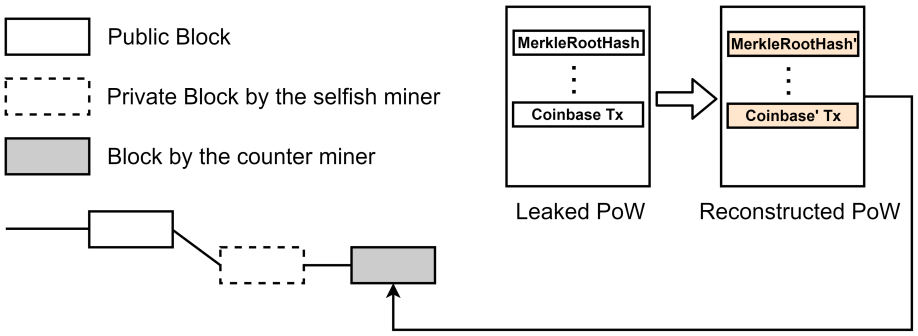

Figure 5: Counter strategy description

4.3 Countering selfish mining

Algorithm 4 Create a PoW task based on leaked information

```
1: w: the leaked PoW task
2: r: the leaked prevBlockHash
3: m′: the new transaction list obtained by the coinbase of the counter miner
4: w′: the counter PoW task
5:
6: procedure CreateTaskFromLeakedInfo
7:     if w is unknown then
8:         (v, r, m, st, T, q) ← w
9:         w′ ← CreateTask(v, r, m′, st, T, q)
10:        return w′
```

We describe our countermeasure after detection of selfish mining in this subsection. We refer miners who use our countermeasure as counter miners. Some miners who detected a selfish miner have the selfish miner's PoW task. The PoW task includes, especially, **prevBlockHash** as shown in Section 2. Miners who have this information can create a new PoW task based on it and it is the first step of our countermeasure. Figure 5 illustrates the situation that the counter miner, who does mining with leaked information, creates a new PoW reconstructed by replacing two values those are coinbase and MerkleRootHash. The counter miner should change the coinbase transaction to make the reward to him and the MerkleRootHash depends on the list of transactions. Algorithm 4 describes the PoW reconstruction process. Consequently, it enables other miners to do mining on the unpublished chain of the selfish miner. If the counter miner publishes the gray box block which is the prior to the selfish miner's block, the selfish miner should publish his private chain because her strategy does not allow the shorter state. Figure 6 shows the total process our strategy.

It means, in any state of the selfish miner, the state has possibility to transit to State 0 if the counter miners find a prior block to the private chain of the

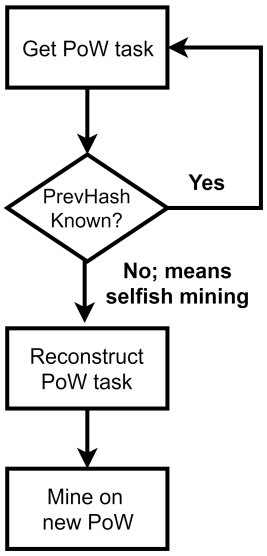

Figure 6: Flowcharf of the Counter Mining

selfish miner. Consequently, the state machine of the selfish mining varies to the following Figure 7. In this figure, it contains transitions to zero state on every positive state. This transition by δ possibility indicates success of the counter mining. δ is a participation proportion of the counter mining. It disturbs the selfish miner to maintain her long private chain. At the same time, the counter miner's revenue increases.

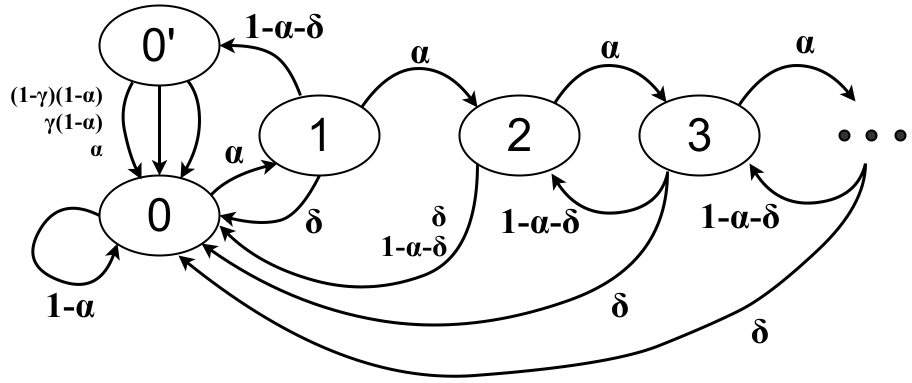

Figure 7: State machine of Selfish Mining with Leaked PoW

where

α is hashrate of the selfish miner

δ is hashrate of the miners who do the counter mining

γ is a proportion of the miners who mines on selfish miner's chain during fork.

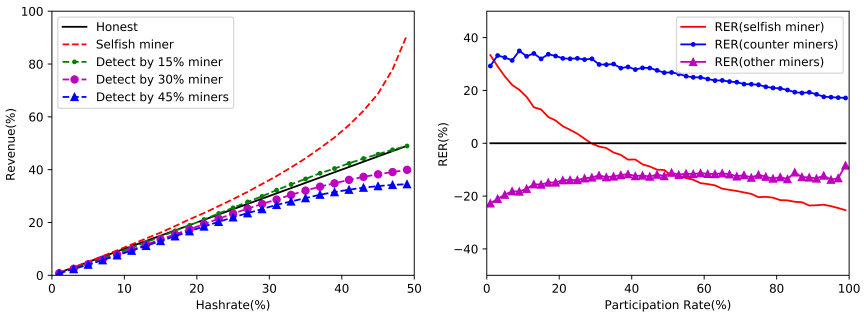

Figure 8: Selfish mining reward variation with counter miners

4.4 Simulation

To validate our counter strategy against the selfish mining, we did two simulations based on the state machine shown in Figure 7. In simulation, we iterated 100,000 times with γ value which is fixed on 0.5. The values on the graph are given by the average value of iteration. At first, we varied a proportion of selfish miner on pooled mining. In the left plot of Figure 8, the dotted red line is revenue of the selfish miner with honest miners. Other lines show revenue of the selfish miner with various proportions of counter miners. When near 15% of the counter miners do counter strategy, the selfish miner get revenue similar to honest mining. Over 15% of the counter miners, the selfish mining gets damage on its revenue as well as no extra revenue. It means, for instance, one mining pool with over 15% hashrate can contain the selfish miner.

In the second simulation, the right plot of Figure 8, we investigated the variation of miners' extra revenue according to the participation rate of miners do detection and the counter strategy. We used Relative Extra Revenue(RER) which shows the proportion of the extra revenue over honest mining. It is given as Formula 4.1. R_h is the revenue on honest mining. R_n is the revenue on the simulation condition.

$$RER = \frac{R_n - R_h}{R_h} \tag{4.1}$$

where

R_n is the revenue of miners with the new strategy

R_h is the revenue of miners with honest mining

This simulation is under the condition that the selfish miner's hashrate is 40% of the Bitcoin network. In total range, counter miners(the circle dotted line) against selfish mining get high RER value over 20%. And the other honest miners(the triangle dotted) who do selfish mining neither counter strategy get negative RER value. Hence, it is always advantageous to use a counter strategy in the presence of a selfish miner. The selfish miner(the red line) gets negative RER value when near 30% miners participate on the counter strategy.

4.5 Variation

Optimal forms of selfish mining are studied in optimal selfish mining [16] and Stubborn attack [11]. In the middle of them, optimal selfish mining [16] uses dynamic selfish mining strategies which depend on miner's hashrate and network environment to make her revenue optimal. Since they share the basic structure of selfish mining strategy, they are based on not publishing blocks. Hence, they are also significantly affected by our counterstrategy using the PoW oracle in the pooled mining environment.

5 Discussion

In this section, we discuss our proposal in several ways. At first, we discuss how our strategy affects the selfish mining. Secondly, we considered security issues which can be happen. At the last part, we consider harmfulness of our method to the network.

Impact to selfish mining Our method makes selfish mining tricky. The result of existing researches in selfish mining says selfish mining gives big revenue to selfish miners. However, with our countermeasure, selfish miners cannot get their goal and even loose their revenue. The counter miners always gain bigger relative revenue than honest mining. If all miners do honest mining, the counter miners do honest mining. This method does not give miners damage in any situations. More counter miners, selfish miners get more damage on their revenue. Consequently, a mining pool over 25% hashrate exists, it cannot try it easily because of risk caused by our research.

Security We can consider an adversary who wants to bypass our method. The adversary needs to hide the core information used in reconstruction of PoW task, PrevBlockHash. This value is necessary for miners to solve PoW puzzles in mining pools. It is, therefore, impossible to distribute PoW task without the hash value of the previous block. To consider other bypass method, the adversary needs to change the coinbase transaction in order to block that counter miners reconstruct a new PoW task. In Stratum protocol, a mining pool gives transaction information to miners [3]. Not like Stratum protocol, mining pools can create a PoW task which does not include a transaction list. Because miners need the hash value of the merkle root not the transaction list in solving

PoW directly. So, the counter miners should create a transaction list and obtain MerkleRootHash from it. If the selfish miner does not leak transaction information and already generated private blocks, a block generated by the counter miner can conflict with transactions already contained in private blocks. In this case, the counter miner can avoid this integrity problem by generating an empty block only with coinbase transaction.

Harmfulness of counter miners The selfish miners exploit honest miners and the counter miners exploit selfish miners. While miners implement the counter strategy, still honest miners get damage on their revenue. In simulation, a little difference of RER of honest miners exists by counter miners. Nevertheless, the honest miners suffers more than 10% in total range as shown in Figure 8. Our method is not malicious but it can be controversial to say it is harmless or not.

6 CONCLUSION

In this paper, we studied selfish mining and its counter strategy in the pooled mining environment. We demonstrated information can easily expose to random miners and other pools can do mining on the attacker's chain with this information. the selfish miner can get damage by counter miners makes selfish mining of mining pools tricky. Moreover, by employing the our method with the selfish miner, the miners can get significant extra revenue. It means motivation to use our method is enough for miners.

References

[1] Bitcoin wiki : Bitcoin protocol documentation.

[2] A next-generation smart contract and decentralized application platform.

[3] Slush pool team, "stratum mining potocol".

[4] John R Douceur. The sybil attack. In *International workshop on peer-to-peer systems*, pages 251–260. Springer, 2002.

[5] Ittay Eyal. The miner's dilemma. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 89–103. IEEE, 2015.

[6] Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. *Communications of the ACM*, 61(7):95–102, 2018.

[7] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol: Analysis and applications. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 281–310. Springer, 2015.

[8] Ethan Heilman, Alison Kendler, and Aviv Zohar. Eclipse attacks on bitcoin's peer-to-peer network.

[9] Yujin Kwon, Dohyun Kim, Yunmok Son, Eugene Vasserman, and Yongdae Kim. Be selfish and avoid dilemmas: Fork after withholding (faw) attacks on bitcoin. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 195–209. ACM, 2017.

[10] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[11] Kartik Nayak, Srijan Kumar, Andrew Miller, and Elaine 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.

[12] Rafael Pass and Elaine Shi. Fruitchains: A fair blockchain. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, pages 315–324. ACM, 2017.

[13] Carlos Pinzón and Camilo Rocha. Double-spend attack models with time advantange for bitcoin. *Electronic Notes in Theoretical Computer Science*, 329:79–103, 2016.

[14] Fabian Ritz and Alf Zugenmaier. The impact of uncle rewards on selfish mining in ethereum. *arXiv preprint arXiv:1805.08832*, 2018.

[15] Meni Rosenfeld. Analysis of bitcoin pooled mining reward systems. *arXiv preprint arXiv:1112.4980*, 2011.

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

[17] Siamak Solat and Maria Potop-Butucaru. Zeroblock: Timestamp-free prevention of block-withholding attack in bitcoin. *arXiv preprint arXiv:1605.02435*, 2016.

[18] Yonatan Sompolinsky and Aviv Zohar. Secure high-rate transaction processing in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 507–527. Springer, 2015.

[19] Ren Zhang and Bart 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.