How Does FAW Attack Impact an Imperfect PoW Blockchain: A Simulation-based Approach

Haorao Zhu[a], Runkai Yang[a], Jelena Mišić[b], Vojislav B. Mišić[b], Xiaolin Chang[a]
[a]Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, P. R. China
[b]Ryerson University, Toronto, ON, Canada
Email: [a]{21112051, 18112049, xlchang}@bjtu.edu.cn; [b]{ jmisic, vmisic}@ryerson.ca

ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9838837

***Abstract*—Malignant miners with small computing power can achieve unfair revenue and degrade system throughput through launching Fork after withholding (FAW) attack in a Proof-of-Work (PoW) blockchain system. The existing works about FAW attack have some of the following issues: (i) only studying Bitcoin blockchain, (ii) assuming that the blockchain network is perfect and then ignoring forks due to block propagation delay, and (iii) assuming that there is only one pool under attack. This paper attempts to investigate FAW attack in imperfect Bitcoin and Ethereum networks where malicious miners attack multiple victim pools. We develop a simulator to capture the chain dynamics under FAW attack in a PoW system where the longest-chain protocol is used. Two different computing power allocation strategies for malicious miners, PAS and EAS, are investigated in terms of the profitability of FAW adversaries, the loss of victims, and the blockchain throughput. The results reveal that FAW adversaries can get more revenue under PAS when more victim pools are subjected to attack in both Bitcoin and Ethereum. If FAW adversaries adopt EAS and the number of victims vary from 1 to 12, they can get maximal revenue when attack 7 victims in Bitcoin. The blockchain throughput decreases significantly under PAS while it is almost unchanged under EAS with the increasing number of victims in both Bitcoin and Ethereum. Our work helps the design of countermeasures against FAW attack.**

Keywords—Bitcoin, Ethereum, Fork After Withholding, Proof of Work.

I. INTRODUCTION

Blockchain is a system of recording information in a way that makes it difficult or impossible to change, hack, or cheat the system. It is being applied in a range of scenarios such as banks, energy, Internet of Things (IoT), and more [1]. Most blockchain systems adopt Proof-of-Work (PoW) consensus protocol, where miners earn rewards by producing blocks. Bitcoin [2] and Ethereum [3] are the two most popular PoW-based blockchains, which are vulnerable to multiple attacks, such as double-spending, attack, selfish mining, and block withholding attack, which are three classic attacks [4]. Among them, block withholding attack (BWH) is one of the easiest attacks to be performed in terms of the attacker's computing power and fork after withholding attack (FAW) is one of the most profitable BWH variants [5]. FAW attackers can get unfair revenue even with small computing power and they are hard to be detected [5]. However, double-spending adversary needs more than 50% of total computing power and selfish mining needs about 20% [5]. Both of these two classic attacks need adversaries to make significant changes to the blockchain [11], and thus, they are easy to be detected.

FAW attack has gained attention from researchers. The prior works [5][6] about FAW attack only studied perfect Bitcoin network, where there was no propagation delay. However, propagation delay is non-negligible and then delay-caused forks (named unintentional forks in this paper) are a fundamental phenomenon in PoW blockchain networks, affecting blockchain performance [7]. Thus, it is necessary to consider the unintentional fork when analyzing attacks. There have been researchers studying selfish mining and withholding attacks in imperfect Bitcoin and Ethereum, but there is less work on analyzing FAW attack in imperfect blockchains. In addition, FAW adversary is able to attack multiple victim pools, which is not considered in [6]. Although the authors in [5] gave the metric formulas when there were *n* victim pools, they only analyzed the scenario of two victim pools in Bitcoin with a perfect network.

All the above discussions motivate the work of this paper. This paper aims to analyze the relationship between the number of victim pools and the revenue of FAW adversary. Although state-space model-based techniques can quantitatively analyze the relationship, they have the issue of the state space explosion when there is more than one victim pool in an imperfect blockchain network. In this paper, we exploit a simulation-based technique for studying FAW attack. The contributions of this paper are summarized as follows.

- We develop a simulator to imitate the chain evolution and dynamics in an imperfect PoW blockchain system. The characteristics of the simulator are as follows: a) the number of victim pools is unlimited; b) the network is imperfect and then there are unintentional forks in the system; c) FAW adversary distributes his computing power arbitrarily, and we develop two strategies, and d) malicious miners in victim pools submit blocks in some cases (detailed in Section III.A). To the best of our knowledge, we are the first to analyze FAW attack in an imperfect blockchain with multiple victims.
- We consider miner revenues in Bitcoin and Ethereum systems, respectively. Miner revenue of a type of miners is the sum of their rewards. Both malicious miners' gain and victims' loss are considered in our work. Bitcoin and Ethereum use different reward mechanisms and then their methods of calculating revenues are different. Moreover, we investigate the system throughput under FAW attack. To the best of our knowledge, we are the first to study these metrics in FAW attack.
- By conducting simulations, we investigate the impact of FAW attack in a variety of blockchain systems. Moreover, we make a comparison between the two computing power allocating strategies. To the best of our knowledge, we are the first to propose computing power allocation strategies for FAW adversary when attacking multiple pools.

By our work, honest miners can detect whether there is FAW attack and pool managers can design countermeasures like more secure in-pool reward mechanisms to mitigate FAW attack. We leave these works to future work.

The rest paper is organized as follows. Related work is given in Section II. Section III describes the system to be investigated, the designed simulator, and the metrics to evaluate FAW attack. Section IV gives the experiment settings and the results. Conclusions and future work are given in Section V.

II. Related Work

This section focuses on FAW attack in Bitcoin and Ethereum systems, and on some related attacks. Through FAW attack, attackers can attain about 56% extra reward than BWH attack [5]. The comparison of FAW attack with BWH attack was conducted from the aspect of mining rewards in [6], where attacking and defense strategies were investigated. However, unintentional forks were ignored in these works and only the metric of mining revenue was analyzed.

Conversely, we investigate the reward mechanisms used in Bitcoin and Ethereum, and more system features are studied as follows: (a) We investigate the impact of network delay on miner revenue and system throughput. (b) In our system, in order to get rewards, each pool finds blocks on its own block (if they have) when there exist forks. (c) The authors in [5][6] assumed that infiltration miners (a part of FAW attackers) have no computing power when computing the probabilities of MP and VP in generating blocks. Actually, infiltration miners also contribute computing power, produce blocks and impact the probabilities. In our system to be studied, the efforts of IMs are involved in calculating the revenue of miners. (d) In our system, the numbers of VPs and OPs are unlimited. Attackers are allowed to infiltrate into an arbitrary number of VPs. The detailed definitions of MP, VP, and OP are described in Section III.A.

To resist FAW attack and other withholding attacks, new pool reward mechanisms like [6] have been designed. Our work can advance the design and evaluation of the countermeasures against FAW attack.

III. System Description and Simulator

This section first describes the system in Section III.A. Then the simulator and the metrics used for evaluation are detailed in Section III.B and Section III.C, respectively.

A. System Description

The blockchain system in the paper uses PoW consensus protocol and longest-chain protocol as the chain selection protocol. There are multiple miners in the system, and miners try to generate blocks by solving a hash-based puzzle in order to earn rewards. There exists propagation delay in the blockchain network so that unintentional forks can occur.

It is very difficult for individual miners to generate a block in a PoW blockchain with extremely large total computing power [8]. Thus, miners get together and form multiple mining pools, and miners in the same pool share the pool revenue. Most blocks are generated by mining pools instead of individual miners in Bitcoin and Ethereum [9][10]. There are two major roles in a mining pool, manager and in-pool miner. The manager determines the block on which in-pool miners mine and publishes valid blocks. He also allocates mining rewards to in-pool miners. All pools can be classified into three types: malicious pool (MP), victim pool (VP) and other pool (OP). There are two types of in-pool miners in a VP, honest miner (VH) and infiltration miner (IM). Different from VP, there is only one type of in-pool miner in MP and OP, innocent miner and honest miner, respectively. IMs in all VPs and the in-pool miners in the MP are malicious miners. The rest are honest miners. All malicious miners share the revenue they get. To analyze the maximum revenue of malicious miners, we assume that there is only one MP while there are multiple VPs and OPs in the system.

To generate a valid block, a miner has to solve a hash-based cryptopuzzle, where the miner needs to find a nonce value that makes the hash value of the block header less than a target value. The nonce value is named a Full Proof-of-Work solution (FPoW). In a pool, each miner tries to find Partial Proof-of-Work solutions (PPoWs), which are much easier than finding an FPoW and is recorded as the work of in-pool miners. Note that most PPoWs cannot produce a new valid block. When a PPoW is also an FPoW, a new valid block is born. This PPoW will be submitted to the manager and the manager publishes the block and gets rewards according to the reward mechanism. PPoWs can also be used by the manager to distribute rewards for in-pool miners. For example, if an in-pool miner submits 4 PPoWs in a time interval and the pool manager receives 13 PPoWs in this interval, the miner gets 4/13 of the revenue of the pool during this time interval.

Different from the existing works, this paper has no limitation on the number of VPs and OPs, and the numbers can be different. Fig. 1 illustrates the pools and miners in the system, including one MP, n VPs, and m OPs. The malicious miners conduct FAW attack [5] but only IMs mine maliciously. All VHs and the miners in MP and OPs employ the honest mining strategy, namely, submitting all PPoWs as long as finding them out. Thus, from the perspective of pools, all pools employ the honest mining strategy, detailed as follows.

- Each pool always explores blocks in the longest chain from its own opinion.
- Each pool can bring about unintentional forks in imperfect networks due to block propagation delay and/or bad network connection.
- If there are forks and there are multiple chains having the same highest block (the block with largest block height), a pool will mine on the chain which is created by itself. Otherwise, a pool randomly selects one chain to work.
- Pools can reference blocks in Ethereum, and only up to two blocks being referenced by a block is allowed.

IMs always submit PPoWs that are not FPoWs. They submit FPoWs strategically and we describe them as follows.

- Each IM discards its FPoW when MP or VHs generate valid blocks.
- Each IM submits its FPoW immediately when it finds that OPs generate blocks. Note that if MP or VHs generate valid blocks at the same time, each IM still discards its FPoW. If IMs in

different VPs find FPoWs, they also withhold their blocks since no OPs generate blocks.

- If an IM finds another FPoW before other miners in the blockchain generate a valid block, it discards the old FPoW and withholds the new one.

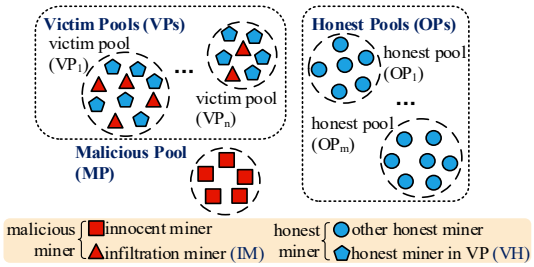

Fig. 1. Illustration of pools and miners in the system.

A miner outside a VP cannot distinguish whether an in-pool miner of VP is IM or VH. Thus, if IMs in VP1 withhold an FPoW and then IMs in VP2 and OPs generate two blocks in the next round, the IMs in VP1 discard their FPoW. The VP's manager also does not know the existence of attackers since they submit PPoWs normally. Thus, the manager allocates rewards to IMs normally although IMs do not submit their FPoWs in many cases. Note that the rewards of attackers contain MP's and IMs'. As a result, MP gets legitimate rewards and IMs get unfair rewards. Thus, malicious miners can conduct FAW attack to get unfair revenue.

The computing power of IMs is a key parameter because this affects the revenue of IMs and then affects the attackers' revenue. Thus, it is important for the attackers to determine the total computing power of IMs and allocate it appropriately. In this paper, we investigate two different computing power allocation strategies as follows:

- Equal allocation strategy (EAS). Attackers distribute the computing power to each VP equally. That is, all IMs have the same computing power.
- Proportional allocation strategy (PAS). Attackers allocate computing power in proportion to VHs' computing power. This means IMs in VPs with high computing power (denoted as large VPs) have more computing power than in VPs with low computing power (denoted as small VPs).

We explain EAS and PAS as follows. Assume that the attackers infiltrate into two VPs and the total computing power of IMs is τ. v_1 and v_2 denote the computing power of honest miners in VP1 and VP2, respectively. If attackers adopt EAS, the computing power of IMs in both VPs is $\tau/2$. However, if attackers adopt PAS, the computing power of IMs in VP1 and VP2 is $v_1 \cdot \tau/(v_1+v_2)$ and $v_2 \cdot \tau/(v_1+v_2)$, respectively.

There are several assumptions in the system we consider.

- At most two blocks can be created simultaneously due to the small probability of the occurrence of a three-chain fork [12]. Thus, the chain number of an unintentional fork is two.
- There is one and only one group of attackers in the system and they only conduct FAW attack.
- The total computing power in the system is constant, and all miners keep mining blocks.

B. Simulator Design

We implement a simulator by C language. The framework of the simulator is illustrated in Fig. 2. The notations we use in the rest of this paper are as follows.

- n : number of VPs;
- Δ : network delay;
- *MAX_HEIGHT* : maximum block height;
- bgt_i : block generation time for miner/pool i ;
- cp_i : computing power of miner/pool i ;
- τ : the total computing power of IMs;
- *chain_num* : the number of chain at current block height;
- *hold* : IMs withholding flag. If IMs withhold blocks, *hold =1*, otherwise *hold = 0*.

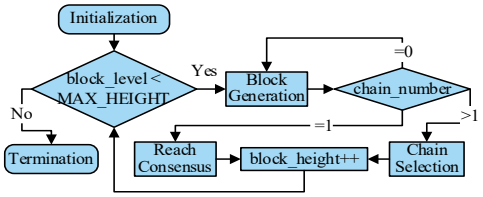

Fig. 2. Main routine of the simulator.

The details of the simulator framework are as follows. Firstly, the parameters about the system (see **Algorithm 1**) are initialized and their settings are given in Section IV.A. In each round of simulation, the simulator runs until the current block height reaches the *MAX_HEIGHT*. Pools first generate valid blocks, shown in **Algorithm 2**. When more than one block is generated and published, which means more than one chain is generated, a pool has to select one chain and create blocks on it. If only one block is published, all pools get to a consensus. However, the number of chains can be zero, which means IMs generate blocks and withhold them. In this case, no miner/pool publishes block so that the simulator goes to block generation function until at least one block is published.

C. Metrics for Evaluation

Two types of metrics are used for evaluation: relative revenue and transactions per second.

1) Relative Revenue

Commonly, reward mechanism is an essential component in PoW blockchains. A pool's revenue is the sum of all types of rewards it obtains. Since the total revenue of all pools can vary,

Algorithm 1 Initialization

Input: $n, \Delta, \tau, MAX\_HEIGHT$

1. set $n, \Delta, \tau, MAX\_HEIGHT$
2. $chain\_num, hold \leftarrow 1, 0$
   / set computing power of IMs */*
3. **if** EAS **then** *//adopt EAS*
4. **for** i **from** 1 **to** n
5. $cp_{IM_i} \leftarrow \tau / n$
6. **end for**
7. **else** *//adopt PAS*
8. **for** i **from** 1 **to** n
9. $cp_{IM_i} \leftarrow \tau \cdot cp_{VH_i} / SUM(cp_{VH})$
10. **end for**
11. **end if**
12. generate genesis block
13. $block\_height \leftarrow 1$

Algorithm 2 Block Generation

```
chain_num ← 0
/* find the first generated block */
for m in {MP, IM_i, VH_i, OP}, i = 1,2,...,n
    m work out a valid block b_m
    bgt_m ← time for m generating a valid block
end for
t ← min(bgt_MP, bgt_IMi, bgt_VHi, bgt_OP), i = 1,2,...,n
for m in {MP, IM_i, VH_i, OP}, i = 1,2,...,n
    if bgt_m − t ≤ Δ and chain_num ≤ 2 then
        if m != IM then
            publish b_m
            chain_num++
        else                          //if IMs generate block
            withhold b_m
            hold ← 1
        end if
    else
        b_m ← NULL
    end if
end for
if only OPs publish blocks and hold==1
    for i from 1 to n                 //IMs submit their blocks
        If b_IMi != NULL then
            publish b_IMi
            chain_num++
        end if
    end for
else if no block published
    IMs withhold their blocks b_IMi if they have
else
    IMs discard their blocks
    hold ← 0
end if
```

we use relative revenue to evaluate a pool's profit. A pool's relative revenue is defined as the proportion of the pool's revenue in all revenues. This paper uses the relative revenue of the attackers to evaluate the profitability of FAW attack. Bitcoin and Ethereum use different reward mechanisms. In Bitcoin, a pool gets one unit of reward after the block generated by him is validated. If two blocks with the same block height are created at the same time (namely, a fork occurs), one of the blocks is selected as the regular block (by the longest-chain protocol), and the other is the stale block. Thus, only a miner/pool is rewarded.

Ethereum allows rewards to be given to some creators of stale blocks. If a stale block's parent block (namely, its previous block) is a regular block and a regular block references it, the creator of this stale block is rewarded. This type of stale block is named uncle block, and a regular block that references uncle blocks is called nephew block. We use distance to denote the difference in block height. Miners can get uncle and nephew rewards only when the distance between its block and referenced is from 1 to 6 in Ethereum. The creators of uncle and nephew blocks can obtain uncle and nephew rewards, respectively.

We now explain why using relative revenue of the largest VP to evaluate the loss of VPs. FAW attack is difficult to detect for VPs [5] since IMs submit PPoWs normally. Relative revenue is a metric for a pool manager to obtain. If the relative revenue gets lower, probably some in-pool miners behave dishonestly. Then the manager can check the honesty of all in-pool miners by some measure. In addition, attackers tend to infiltrate into large pools (pool with large computing power) to get more revenue [5]. Thus, the largest VP is the most vulnerable no matter how many VPs the malicious miners attack.

2) Transactions Per Second (TPS)

We calculate Transactions per second (TPS) of Bitcoin and Ethereum under FAW attack to find the impact of this attack on the whole system. Only the transactions in the regular blocks can be accounted in Bitcoin and Ethereum. The number of transactions in a block is almost constant, and thus TPS can be evaluated by the regular block generation rate. As one regular reward means that a regular block is generated, we can calculate the regular block generation rate by the regular reward.

IV. Experiment Evaluation

This section first evaluates the impact of FAW attack under two computing power allocation strategies given in Section III.A in terms of the profitability of the attackers and the loss of the victims. Then, the effect of FAW attack on the system is studied in terms of TPS.

A. Parameters Settings

Computing power of pools. This is one of the key parameters of mining pools which determines the rate of generating valid blocks. The websites of Bitcoin and Ethereum [13][14] show the pool names and their corresponding computing power. Our experiments consider the mining pools with more than 1% of total computing power, illustrated in TABLE I. Since the computing power fluctuates continuously, we use the approximate value in our simulation (shown in TABLE I). Without loss of generality, the pool with the largest computing power is assumed as attacker, namely, *AntPool* in Bitcoin and *Ethermine* in Ethereum. The attackers infiltrate into pools in order of computing power. That is, if the malicious miners in Bitcoin attack 2 pools ($n = 2$), *F2Pool* and *Foundry USA* are the VPs and the rest of pools are OPs. Moreover, attackers can get more relative revenue when infiltrating into large VP than small VP [5]. Thus, attackers tend to infiltrate into top n largest mining pools when the number of VPs is n .

Computing power of IMs. Attackers can distribute their computing power according to their pre-defined strategy. When EAS is adopted, IMs in each VP have the same computing power. However, for PAS, IMs in large VP have more computing power than in small VP. The attackers need to determine the total computing power of IMs (namely, the value of τ). In each group of parameters, we perform multiple rounds of simulations to find the approximate optimal τ , which maximizes the relative revenue of the attackers.

Network delay. In an imperfect network, it takes a period of time for valid blocks to reach most nodes, which is around 400ms in Bitcoin [15] and 300~500ms in Ethereum [16][17]. Note that the average block generation interval is 10 minutes in Bitcoin and 13.7s in Ethereum [18]. To get general conclusions, we normalize the block generate interval to 1. Thus, the network delay Δ is 0.0005 and 0.025 in Bitcoin and Ethereum, respectively, in our experiments.

Other settings. In each round of the simulation, a blockchain is created with 10 million regular blocks and we do 50 rounds for each group of parameters.

TABLE I MINING POOLS WITH MORE THAN 1% COMPUTING POWER IN BITCOIN AND ETHEREUM

Mining pools in Bitcoin	Computing power / Value in simulator	Mining pools in Ethereum	Computing power / Value in simulator
AntPool	19.87% / 20%	Ethermine	26.10% / 25%
F2Pool	16.74% / 15%	F2Pool_2	24.33% / 25%
Foundry USA	12.97% / 13%	HiveonPool	9.44% / 10%
unknown	11.09% / 10%	NanoPool	4.73% / 5%
ViaBTC	9.83% / 10%	Flexpool.io	3.84% / 3.5%
Binance Pool	7.95% / 7%	0xb7e3…f707	3.28% / 3%
BTC.com	7.11% / 7%	2Miners_PPLNS	3.25% / 3%
SlushPool	6.07% / 6%	MiningPoolHub_7	2.43% / 2.5%
EMCDPool	1.88% / 2%	AntPool	2.37% / 2.5%
SBI Crypto	1.46% / 1.5%	BinancePool_2	2.32% / 2%
MARA Pool	1.46% / 1.5%	0x2a20…0050	2.04% / 2%
Luxor	1.26% / 1%	BTC.com	1.45% / 1.5%
Huobi.pool	1.26% / 1%	HuobiPool	1.30% / 1.5%

B. *Simulation Results*

Fig. 3-Fig. 10 show the results of the profitability of miners and TPS under FAW attack, detailed in the following.

1) Relative Revenue of the Attackers

We first evaluate the relative revenue of the attackers by varying the number of VPs. Fig. 3 and Fig. 4 show the results, where "EAS" and "PAS" denote the results of adopting equal allocation and proportional allocation strategies, respectively.

We observe that both EAS and PAS enable the attackers to get more relative revenue when the number of VPs (n) is under 10. In Bitcoin, the relative revenue of the attackers is roughly the same by adopting two strategies when $n \le 4$. The relative revenue of the attackers adopting PAS increases slowly when they infiltrate into more than 7 VPs. This is because large mining pools get more revenue than small pools. Attackers infiltrate into more small pools so the computing power of IMs in large pools decrease. Thus, the relative revenue of the attackers increases slightly. However, the relative revenue decreases dramatically when they adopt EAS and infiltrate into more than 7 VPs. This is because the computing power of IMs is the same under EAS so that the computing power of IMs in large VPs decrease significantly when attackers infiltrate into more pools. On the one hand, IMs in large VPs get a lower fraction of revenue and the attackers' relative revenue decreases. On the other hand, the computing power of pools after the 8th (after *EMCDPool* in TABLE I) is around 1% in Bitcoin, which is much lower than that of the top ranked pools. As the result, the attackers' relative revenue drops significantly when they adopt EAS in Bitcoin.

The results are different in Ethereum since miners can get rewards from uncle blocks and nephew blocks, which is impossible in Bitcoin. The relative revenue of attackers increases steadily with the increasing number of VPs when they adopt PAS. However, the relative revenue hardly increases when EAS is deployed. Nevertheless, the revenue still increases with the increase in the number of VPs. In addition, PAS dominates EAS in terms of the relative revenue of attackers, especially when attackers infiltrate into more than 7 VPs. This is because the computing power of IMs in large VPs under PAS is more than under EAS so that attackers get more revenue. Thus, the relative revenue is larger under PAS than EAS. For example, *F2Pool* gets 10 times revenue than *SBI Crypto* in Bitcoin. Though attackers get more fraction of revenue in *SBI Crypto* under EAS, the absolute revenue is still a small number. However, attackers get more revenue from *F2Pool* under PAS than EAS, which is a large number than that in *SBI Crypto*.

2) Relative Revenue of the Largest VP

We evaluate the relative revenue of the largest VP (denoted as VP1) under FAW attack in both Bitcoin and Ethereum, which are *F2Pool* and *F2Pool_2*, respectively. Fig. 5 and Fig. 6 show the results. We observe that the relative revenue of VP1 under EAS is always higher than under PAS in both Bitcoin and Ethereum. This is because PAS leads to a more appropriate allocation of computing power than EAS. Namely, more computing power in large VPs and less computing power in small VPs. Note that the computing power of VP1 is around 15% and 25% in Bitcoin and Ethereum, respectively. We find that VP1 gets additional relative revenue under EAS when attackers infiltrate into more than 5 VPs in both Bitcoin and Ethereum. However, attackers adopting PAS always make VP1 get loss. As a competitor to the attackers, VP1 gets extra relative revenue under EAS, which the attackers do not expect. On the contrary, PAS is a strategy for self-interest. That is, PAS dominates EAS in terms of the competitor's relative revenue.

We also observe that VP1's relative revenue increases rapidly when $n \le 6$ while it increases slowly after $n \ge 7$ in Bitcoin. VP1's relative revenue always grows slowly over n in Ethereum. This is because the computing power of the top 7 (except the attacker) mining pools is large in Bitcoin. The computing power of two adjacent mining pools is close. Thus, the computing power of IMs in VP1 (denoted as IM1) decreases rapidly when attackers infiltrate into more VPs. However, pools after 7th place (after *SlushPool* in Bitcoin and *MiningPoolHub_7* in Ethereum) have little computing power.

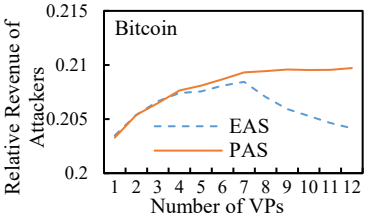

Fig. 3. Relative revenue of attackers in Bitcoin over number of VPs.

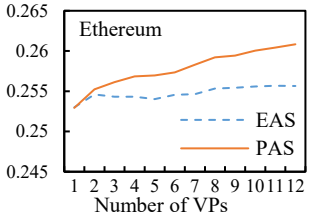

Fig. 4. Relative revenue of attackers in Ethereum over number of VPs.

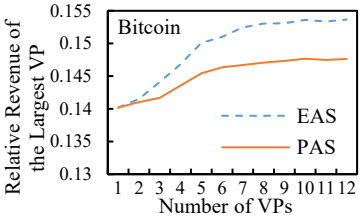

Fig. 5. Relative revenue of largest VP in Bitcoin over number of VPs.

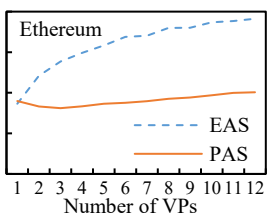

Fig. 6. Relative revenue of largest VP in Ethereum over number of VPs.

When attackers adopt PAS, IMs in these pools have little influence on the computing power of IM1. As the total computing power of the attackers is limited, the total computing power of IMs is also limited. If attackers adopt EAS, each group of IMs' computing power decreases greatly when n is small, while it decreases slightly when n is large. In addition, the approximate optimal τ drops when $n \geq 7$ under EAS, which means the total computing power of IMs decreases. As a result, VP1's relative revenue increases rapidly when n is small and grows slowly when n is large under PAS and EAS in Bitcoin. On the contrary, VP1's relative revenue always grows slowly over n in Ethereum. This is because the top 3 mining pools occupy 60% of the total computing power and the rest pools share the remaining 40%. That is, the computing power of the first two VPs is significantly large than the rest of VPs. Thus, n has little effect on the computing power of IMs in VP1 under PAS. The computing power of IM1 can be almost unchanged by adjusting τ under PAS. τ is stable under EAS in Ethereum so that IM1's computing power is only related to n.

3) Transactions Per Second (TPS)

TPS reflects the impact of FAW attack on the whole system and τ has a significant impact on TPS. Fig. 7-Fig. 10 show the results. We find that TPS is low under large τ but high under small τ. As explained in Section IV.B.1, attackers can obtain more relative revenue under PAS when they infiltrate into more VPs in both Bitcoin and Ethereum. However, there is a maximal relative revenue under EAS in Bitcoin and almost unchanged in Ethereum. Thus, when attackers adopt PAS, they increase τ to get more relative revenue when infiltrating into more VPs so that the TPS keeps decreasing. τ increases first and then decreases under EAS in Bitcoin and TPS is the opposite. In Ethereum, τ is almost stable and TPS has no obvious change under EAS. Both EAS and PAS let the TPS lower than no attack in the system. Furthermore, the loss of TPS under EAS is not as obvious as PAS. That is to say, FAW attack is difficult to detect in Ethereum by observing TPS when attackers adopt EAS.

V. Conclusion and Future Work

This paper exploits a simulation-based technique to study FAW attack. We propose two computing power allocation strategies for attackers and compare them by varying the number of VPs. Simulation results indicate: (i) Attackers adopting EAS can obtain maximal relative revenue when infiltrating into 7 VPs in Bitcoin. Their relative revenue in Ethereum has no obvious change when they infiltrate into different numbers of VPs. (ii) When attackers adopt PAS, the relative revenue increases with the increasing number of VPs in both Bitcoin and Ethereum. (iii) PAS outperforms EAS in terms of relative revenue but EAS is more difficult to detect than PAS. Our future work will put PAS and EAS in real-world systems and consider scenarios of multiple groups of malicious miners.

VI. Acknowledgement

The work of H. Zhu, R. Yang and X. Chang was supported by Beijing Municipal Natural Science Foundation (No. M22037). The work of J. Mišić and V. Mišić was supported by Natural Science and Engineering Research Council (NSERC) of Canada.

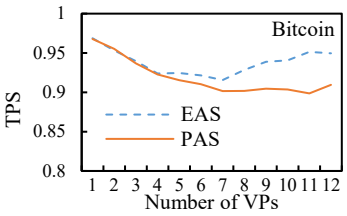

Fig. 7. TPS under FAW attack in Bitcoin over numbers of VPs.

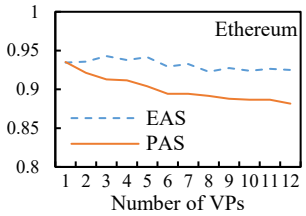

Fig. 8. TPS under FAW attack in Ethereum over numbers of VPs.

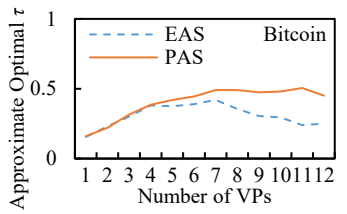

Fig. 9. Approximate optimal τ under FAW attack in Bitcoin over numbers of VPs.

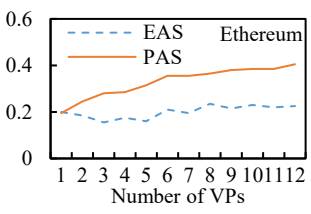

Fig. 10. Approximate optimal τ under FAW attack in Ethereum over numbers of VPs.

References

[1] Q. Chen, G. Srivastava, R. M. Parizi, M. Aloqaily, and I. Al Ridhawi, "An incentive-aware blockchain-based solution for internet of fake media things," INFORM PROCESS MANAG, 57(6), 102370, 2020.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Decentralized Business Review, 21260, 2008.

[3] V. Buterin, "Ethereum white paper," GitHub repository, 1, 22-23, 2013.

[4] H. Kang, X. Chang, R. Yang, J. Mišić, and V. B. Mišić, "Understanding Selfish Mining in Imperfect Bitcoin and Ethereum Networks with Extended Forks," IEEE Trans. Netw. Serv. Manag. 18(3): 3079-3091 (2021).

[5] Y. Kwon, D. Kim, Y. Son, E. Vasserman, and Y. Kim, "Be Selfish and Avoid Dilemmas: Fork After Withholding (FAW) Attacks on Bitcoin," ACM CCS 2017, Dallas, TX, USA, pp. 195-209, 2017.

[6] J. Ke, H. Jiang, X. Song, S. Zhao, H. Wang, and Q. Xu, "Analysis on the Block Reward of Fork After Withholding (FAW)," NSysS 2018, Dhaka, Bangladesh, December 2018. pp. 16-31.

[7] R. Yang, X. Chang, J. Mišić, and V. B. Mišić, "Assessing blockchain selfish mining in an imperfect network: Honest and selfish miner views," Computers & Security, 97, 101956. (2020).

[8] https://tokenview.com/en/minePoolList. Accessed, October 2021.

[9] https://btc.com/stats/pool?pool_mode=year. Accessed, October 2021.

[10] https://etherscan.io/stat/miner/1?range=7&blocktype=blocks. Accessed, October 2021.

[11] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang, and D. Mohaisen, "Exploring the Attack Surface of Blockchain: A Comprehensive Survey," IEEE Commun. Surv. Tutorials 22(3): 1977-2008 (2020).

[12] J. Misic, V. Misic, and X. Chang, "On Ledger Inconsistency Time in Bitcoin's Blockchain Delivery Network," GLOBECOM 2019: 1-6.

[13] https://btc.com/btc/insights-pools. Accessed, November 2021.

[14] https://btc.com/eth/adapter?type=miningstats. Accessed, November 2021.

[15] S. Ben Mariem, P. Casas, and B. Donnet. Vivisecting blockchain P2P networks: Unveiling the Bitcoin IP network. In ACM CoNEXT Student Workshop, 2018.

[16] T. Wang, C. Zhao, Q. Yang, S. Zhang, and S. C. Liew, "Ethna: Analyzing the underlying peer-to-peer network of the ethereum blockchain". arXiv preprint: 2010.01373 (2020).

[17] P. Silva, "Impact of Geo-Distribution and Mining Pools on Blockchains: A Study of Ethereum - Practical Experience Report and Ongoing PhD Work," DSN-S, 2020, pp. 73-74.

[18] https://etherchain.org. Accessed, October 2021.