Uncle-Block Attack: Blockchain Mining Threat Beyond Block Withholding for Rational and Uncooperative Miners

Sang-Yoon Chang[1], Younghee Park[2], Simeon Wuthier[1], and Chang-Wu Chen[3]

[1] University of Colorado Colorado Springs, Colorado Springs CO, USA
[2] San Jose State University, San Jose CA, USA
[3] AMIS, Taipei, Taiwan

Abstract. Blockchain-based cryptocurrency replaces centralized institutions with a distributed network of Internet-based miners to generate currency and process financial transactions. Such blockchain systems reach consensus using proof of work (PoW), and the miners participating in PoW join mining pools to reduce the variance for more stable reward income. Prior literature in blockchain security/game theory identified practical attacks in block withholding attack (BWH) and the state of the art fork-after-withholding (FAW), which have the rational and uncooperative attacker compromise a victim pool and pose as a PoW contributor by submitting shares but withholding the blocks. We advance such threat strategy (creating greater reward advantage to the attackers at the expense of the other miners in the victim pool) and introduce the uncle-block attack (UBA) which exploits uncle blocks for block withholding. We analyze UBA's incentive compatibility and identify and model the critical systems- and environmental- parameters which determine the attack's impacts. Our analyses and simulations results show that a rational attacker is always incentivized to launch the UBA attack strategy (over FAW or protocol compliance) and that UBA is effective even in the unfavorable networking environment (in contrast, in such case, FAW is reduced to the suboptimal BWH attack and does not make use of the withheld block).

Keywords: Blockchain · Cryptocurrencies · Distributed consensus protocol · Security · Mining game · Block withholding attack

1 Introduction

Blockchain technology builds a distributed ledger comprised of irrevocable transactions and has emerged as the backbone technology for digital cryptocurrencies, which govern, generate, and process the financial transactions in a distributed manner (as opposed to relying on a centralized bank), e.g., Bitcoin [17] and

Ethereum [5, 21]. As of April 2019, the total market cap of the blockchain-based cryptocurrencies is estimated to be more than \$170B [3, 8].

Underlying blockchain is the consensus protocol so that all nodes agree on the transactions on the ledger without relying on a trusted third party. The most popular consensus algorithm is based on proof of work (PoW). The miners participate in PoW to generate new currency and process the transactions and are financially incentivized to so by the block rewards, the winnings from solving the probabilistic PoW computational puzzles. To lower the variance of such reward, the miners join and operate as mining pools to share the computational power and the corresponding reward winnings.

PoW consensus protocol is driven by networking, as the found blocks need to get propagated via broadcasting for "timestamping" [17] to determine which block got mined first and will become a part of the blockchain. Because it is in a distributed environment and due to the networking/propagation delays, there can be collisions in the block findings, which results in forking. To improve fairness and decentralization across nodes with varying networking environments, blockchain systems incorporate uncle reward, which are partial rewards for the blocks that got forked but did not become part of the main chain. (We explain the relevant background in blockchain in greater details in Section 2.)

According to the consensus protocol, the miner submits a PoW solution *once* it is found. However, previous research identified practical and relevant attacks which have the attacker compromise a victim mining pool and undermine the winnings by posing as a PoW contributor without actually contributing (the attacker also has a different reward channel in its main pool/solo mining, in which it does not need to share the reward with others, unlike the victim pool). In such attacks, the attacker mines the block and controls the timing of the block submission (including possibly permanently withholding the block, which effectively corresponds to discarding the found block) as opposed to following the protocol and immediately submitting the found block. These attacks are in the forms of block-withholding (BWH) attack and fork-after-withholding (FAW) attack. FAW attack, in particular, builds on both selfish mining and BWH and improves the attacker's reward by submitting the block found in the victim mining pool when there is a fork/collision with a third-party miner, making the threat more relevant to the incentive-driven rational miners.

In this paper, we advance the withholding-based attacks and introduce the uncle-block attack (UBA) which extends the block-withholding to uncle blocks. UBA amplifies the state of the art in FAW attack by increasing the attacker reward at the expense of the other miners within the victim pool. UBA not only builds on FAW (inheriting the attacker-desirable properties of FAW) but is also effective independent to the networking environment of the attacker (i.e., effective even when the attacker loses the forking race, in contrast to FAW). While the UBA attack undercuts the rewards of the other miners in the victim pool (and thus corresponds to the strategy which will be taken by a malicious attacker focusing on sabotaging the victims), we extend our threat model further to include the uncooperative and rational attackers and analyze the attacker's incentives in financial rewards for UBA. Our analyses show that the attacker maximizes its reward by launching UBA attack over the state of the art in FAW or the protocol-complying honest mining. Our reward/payout analyses for the attacker uses lower bounds to be conservative in measuring the attacker's performances. Nevertheless, we show significant gains and incentives for launching UBA. While we use formal modeling and analyses to abstract away from the implementation details and deliver fundamental insights about the advanced withholding threat, our model constructions and analyses are driven by the real-world blockchain implementations and our findings are relevant and applicable to the current practice/design of blockchains.

The rest of the paper is organized as follows. Section 2 provides background information about blockchain relevant to our work; Section 3 discusses about the prior work in blockchain mining security; and Section 4 establishes the threat model and the attack requirements (the same as those for BWH and FAW). We construct our mining game model in Section 5 and use it to recap BWH attack and FAW attack. Afterward, we introduce UBA and analyze its reward in Section 6. We analyze the attacker's dynamic behaviors (optimizing its infiltration of the victim pool and the computational power use) and its attack payouts/rewards in simulations in Section 7. We then discuss about the potential countermeasures in Section 8 and conclude the paper in Section 9.

2 Background in Blockchain Mining

The PoW consensus comprises of two parts: a probabilistic PoW based on solving a computational puzzle (finding the preimage/input of a one-way hash function) yielding a valid block, and the submission/broadcasting of the block via peer-to-peer (p2p) networking (so that the others update their blockchain ledgers with the block to reach a consensus).

Miners execute the consensus protocol and are incentivized to do so because solving the computational puzzle and consequently registering a block on the blockchain generates financial rewards (new currency and the transaction fees). The consensus protocol is based on a race and only the miner solving the PoW puzzle and submitting/broadcasting the corresponding block the earliest wins the corresponding reward in a round; once that happens, the other miners start a new round of mining by updating the chain with the newly found block. While the PoW consensus is designed to be computationally fair (distributing the reward winning proportionally to the computational power of the miners), the PoW miners experience high variance when finding blocks and winning rewards. Because a miner is competing with a global-scale group of other miners and the mining difficulty gets adjusted accordingly, finding a block is very sporadic and bursty.

To lower the variance and to get a more stable stream of reward income, miners form a pool to combine their computational power to share the power as well as the mining reward winnings. The increased computational power by pooling them together increases the occurrence of winning a block whose reward gets split across the pool miners according to their computational contributions. To estimate each of the miner's contributions, mining pool samples more PoW solutions by introducing *shares*, which correspond to solving the same computational puzzle with the same block header as the block but with easier difficulty; a block solution is a share but a share is not necessarily a block. The reward distribution within a pool using such shares is also designed to be computationally fair. To manage the mining pool, the *pool manager* keeps track of the individual miner members' contributions, registers/broadcasts the block upon its discovery, and distributes the reward to the pool members. Joining the mining pool is popular, e.g., as of July 2018, the computationally-largest mining pool (BTC.com) has 24% of the computational power for the entire Bitcoin mining network, and eight mining pools collectively have greater than 85%.

Forking occurs when two block solution propagations result in a collision (i.e., some nodes receive one block earlier than the other while the other nodes receive the other block first), creating a partition between the miners on which block to use for its impending round of mining. Forking gets resolved by having the miners choose the longest chain, e.g., if one partition finds another new block and propagates that to the other partition, then the miners in the other partition accept that chain which is one block longer than the one that they have been using[4]. The block that causes a fork but does not get registered as the main block is called the *uncle block* (as opposed to the parent block registered on the chain)[5]. Blockchain systems provide partial reward for uncle blocks to increase fairness between the miners, i.e., reduce the effect of the networking discrepancy.

3 Related Work in Mining Security

For uncooperative miners (willing to diverge from the set protocol), sophisticated attacks exist to further increase the mining reward beyond following the protocol

[4] In significantly rarer cases, an intentional *hard fork* retains the partition and introduces a new branch/chain. For example, since the launch of the main chain in 2016 ("Homestead"), Ethereum had five hard forks, including three hard forks in 2016 (including the infamous DAO hack incident which split Ethereum and Ethereum Classic), one in 2017, and one in 2019. Intentional *soft forks* are also used for upgrading the blockchain protocols and rules but, in contrast to hard forks, are backward-compatible to the clients running the older softwares. Such intentional forks are out of scope for this paper and we focus on the *accidental forks* caused by the block propagation discrepancy in networking, because the accidental forks occur significantly more frequently than the intentional forks and because the intentional forks are treated differently than the accidental forks which get automatically resolved as described by the longest-chain rule without software updates.

[5] This terminology for the blocks which are valid solutions but did not become the main blocks can differ across implementations, e.g., in Bitcoin, they are called orphan blocks. We uniformly call them *uncle blocks* for simplicity but introduce relevant variables, including the reward amount, to generalize it across implementations.

of timely block submission. These attacks on the consensus protocol actively control the timing of the block submission, including permanently withholding the submission in certain situations. *Selfish mining* withholds a block so that the miner can get a heads-start on computing the next block and have the rest of the miners discard and switch from the blocks that they were mining [10, 13, 19]. Blockchain's confirmation mechanism (which waits for multiple blocks to get mined before finalizing the transactions) resists selfish mining because the probability of successful selfish mining decreases exponentially with the number of blocks required for confirmation [17].

There are further attacks in cases of the mining pool. *Block-withholding attack* (BWH) withholds the mined block in mining pools in order to increase the attacker's reward at the expense of the rest of the pool member miners [18]. In BWH, to sabotage the victim mining pool, the attacker simply never submits the found block while submitting the shares. As a consequence, the attacker still reaps the benefits from submitting the share solutions to the victim pool (pretending to contribute and getting the credit for it) while never actually contributing to the pool (since it never submits the actual block solution which will benefit the victim pool). The attacker also increases the expected reward in its main pool by launching BWH on the infiltrated victim pool.

A recent fork-after-withholding (FAW) attack [14] combines selfish mining and BWH. FAW builds on selfish mining and BWH but creates intentional forks in cases when there is a block being broadcasted by another pool with which the attacker has no association. In other words, while always submitting the shares to gain greater payout on the victim pool, the attacker withholds the found block and either discards it (if the attacker's main pool finds another block or if another miner from the victim pool finds a block) or submits it only when there is another competing block that is being propagated by another third-party pool (creating an intentional fork). FAW attack is significant because it forgoes the miner's dilemma (which establishes that the Nash equilibrium between multiple pools is to launch block-withholding attack against each another, which results in suboptimal performances for all the pools, motivating the pools to be cooperative with each other [9]); there is a real incentive (unfair reward gain) for rational miners to launch FAW attack.

Other researchers investigate the emerging security issues as the block reward transitions to only variable transaction-fees (as opposed to also including the base fees that generate new currencies) [20, 6]. The variance of the block reward amount incentivizes the attackers to selectively mine the blocks in time (e.g., there are "mining gaps" when the attackers decide not to mine). Although related, our work is orthogonal to these issues since the attacker in our work maximizes its reward advantage *given* the block reward.

4 Threat Model: Uncooperative and Rational Attacker

We distinguish between *honest miners* and *attackers* by whether or not they are cooperative and comply with the consensus protocol. More specifically, while honest miners submit/broadcast the blocks once they are found, attackers control the timing of the consensus-puzzle submissions/blocks by introducing a delay between the time when the block was found and when they are submitted.

We investigate the attacks launched by the miners, as opposed to the pool managers. Compromising the pool as a miner is easier than misbehaving as a mining pool manager (which is trusted by and interact with all the miners within the mining pool and is therefore in constant scrutiny). Compromising the pool as a miner is especially easy in a public blockchain (as opposed to permissioned or private blockchain) and against an open pool, in which registration and Sybil attack (generating/using multiple identities) are easy and even encouraged for anonymity.

The attacker is also rational and driven by the financial incentives of block rewards. Therefore, we study whether the attackers strategies are compatible to such incentives; the attack is irrelevant if it does not increase the attacker's reward because the attacker will choose not to use such the suboptimal strategy. While the attacker is primarily driven by its own self interest, its unfair advantage negatively affects the other miners of the blockchain because the finite reward is shared among the miners (i.e., if a miner wins more, then the other miners win less). Our threat model therefore encompasses the malicious threat model (where the attacker's goal is to sabotage or degrade the performances of the other miners) but can also be extended to the uncooperative and rational miners (who merely want to increase their own rewards without necessarily harming the others).

We assume the threat model of BWH attack and FAW attack, in which the attacker compromises multiple pools and separates the main pool vs. the victim pool. The attacker behaves honestly in the *main pool* while it can launch an attack by diverging from the protocol in the *victim pool*. As a realistic setup, we assume that the main pool is comprised of the attacker only (which is effectively solo mining without joining a pool); the attacker shares the reward winnings in the victim pool while there is no sharing and the attacker takes all the rewards in the main pool. Our model also generalizes to the case of multiple pools, e.g., the main pool or the victim pool can be a collection of mining pools, since the PoW consensus is power-fair, as opposed to identity-fair, as is captured in our mining-game model in Section 5.

The attacker setting up main pool vs. victim pool is realistic since setting up a miner account or joining a mining pool are generally easy. This is because public blockchain, such as those used in cryptocurrencies, has loose control of identities and is designed for anonymity (which is in contrast to the permissioned blockchains in other emerging applications), e.g., Nakamoto/Bitcoin suggests using new accounts for new transactions [17]. For example, in 2014, Eligius became a victim pool of the BWH attack and lost 300 BTC, which got detected primarily because the attacker only used two accounts for the attack (which resulted in a detection of abnormal behavior where the accounts submitted only shares but no blocks for too long of a time).

5 Mining Game

5.1 Mining and Computational Power Model

To investigate the incentive compatibility of the attacks, we model the mining game between the miners and quantify the expected reward. We build on the approaches of the prior literature analyzing the withholding-based attacks, e.g., [14, 15], but we construct the model to enable the analyses of the novel UBA attack we introduce. The reward distributed depends on the miner's computational power, and we normalize the following variables with respect to the entire mining network's computation power. i.e., the total mining power has a power of 1. The attacker's computation power is α while the victim pool's mining power without the attacker's power is β, so $0 \leq \alpha + \beta \leq 1$. The attacker splits its power between its main pool (honest mining) and the victim pool (willing to launch mining attack strategies to increase the attacker's reward at the expense of the fellow miners in the victim pool), and the fraction of the attacker's power for infiltration of the victim pool is τ (where $0 \leq \tau \leq 1$). Therefore, the attacker's power on the victim pool is $\tau\alpha$, and the total mining power on the victim pool is $\tau\alpha + \beta$ even though the attacker's power may not contribute to the pool earning reward depending on the attack strategies (honest mining is within the attacker's options, and the attacker will do so legitimately contributing to the pool earning if honest mining is the reward-optimal strategy for the attacker). For example, in the simpler BWH attack, the attacker does not submit block at all in the victim pool so the actual power contributing to block earnings of the pool is only β, while the attacker still earns the mining credit and the corresponding reward through share submissions and the reward earning gets split by $\tau\alpha + \beta$ within the victim pool.

In FAW attack, the attacker submits the withheld block on the victim pool only when another block by a third-party pool is submitted and getting propagated. The attacker is motivated to do so and cause a fork because the attacker does not get any reward if a third-party pool wins the block. c denotes the probability that the attacker's block will get rewarded as opposed to the third-party pool's, and c depends on the networking state/topology between the two block submissions.

A miner's expected reward normalized by the reward amount is denoted with R. For example, if an attacker behaves honestly, its expected reward (R_{honest}) is proportional to its computational power by the design of the PoW consensus and the mining pools,

$$R_{\text{honest}} = \alpha \tag{1}$$

The following summarizes the variables used for the reward analyses of the block-withholding threats (more complexity and variables are added as we use them to describe the mining pool game and analyze the uncle-block attack).

α:	Attacker's computational power
β:	Computational power of the victim pool

τ: Fraction of attacker's power for infiltration of victim

c: Probability that the attacker wins the reward given that there is a fork (collision with another block propagation)

5.2 BWH Attack and FAW Attack Analyses

To provide baselines and examples of the use of our model in Section 5.1, we analyze the expected reward of BWH and FAW. This section adapts the prior work in FAW [14], and we only highlight the parts which are the most relevant to our work in this section.

For BWH, the attacker has two possible events for earning a positive reward (in other events, the attacker earns zero reward). The first event is when the attacker finds a block in its honest-mining main pool (the event A) while the second event corresponds to when another miner from the victim pool, not the attacker, finds a block (B). Because the probability of winning a block is proportional to the computational power spent on mining the block and because $1-\tau\alpha$ amount of power from all miners actually contributes to finding the block (the attacker uses the other $\tau\alpha$ to still compute the PoW but only submit shares while withholding the blocks), the probability of A is $\frac{(1-\tau)\alpha}{1-\tau\alpha}$ and the probability of B is $\frac{\beta}{1-\tau\alpha}$. Assuming negligible probability for natural forking, the expected reward for block-withholding attack normalized by the reward amount (R_{BWH}) is:

$$\begin{aligned} R_{\text{BWH}} &= \mathrm{E}[R|A]\cdot\Pr(A)+\mathrm{E}[R|B]\cdot\Pr(B) \\ &= 1\cdot\frac{(1-\tau)\alpha}{1-\tau\alpha}+\frac{\tau\alpha}{\beta+\tau\alpha}\cdot\frac{\beta}{1-\tau\alpha} \\ &= \frac{(1-\tau)\alpha}{1-\tau\alpha}+\frac{\tau\alpha}{\beta+\tau\alpha}\cdot\frac{\beta}{1-\tau\alpha} \end{aligned} \tag{2}$$

The FAW attack builds on BWH but provides an extra channel for attacker reward. In addition to the events A and B, the attacker can earn reward by broadcasting the withheld block when a third-party miner outside of the attacker-involved main pool and victim pool finds a block, causing a fork and hence the name fork-after-withholding (FAW). This event of the attacker finding a block *and* a third-party miner finding a block is C. The expected reward for FAW attack normalized by the reward amount (R_{FAW}) is:

$$\begin{aligned} R_{\text{FAW}} &= \mathrm{E}[R|A]\cdot\Pr(A)+\mathrm{E}[R|B]\cdot\Pr(B)+\mathrm{E}[R|C]\cdot\Pr(C) \\ &= 1\cdot\frac{(1-\tau)\alpha}{1-\tau\alpha}+\frac{\tau\alpha}{\beta+\tau\alpha}\cdot\frac{\beta}{1-\tau\alpha}+1\cdot c\cdot\frac{\tau\alpha}{\beta+\tau\alpha}\cdot\tau\alpha\frac{1-\alpha-\beta}{1-\tau\alpha} \\ &= \frac{(1-\tau)\alpha}{1-\tau\alpha}+\frac{\tau\alpha}{\beta+\tau\alpha}\left(\frac{\beta}{1-\tau\alpha}+c\tau\alpha\frac{1-\alpha-\beta}{1-\tau\alpha}\right) \end{aligned} \tag{3}$$

We summarize and list the three events which yield the attacker positive rewards, as we also use them for our analyses of the rewards for UBA:

A:	Attacker's main pool finds a block
B:	Another miner from the victim pool finds a block
C:	Third-party miner finds a block

6 Uncle-Block Attack

Uncle-block attack (UBA) builds on the FAW attack but exploits the uncle blocks, which are blocks which caused fork but did not eventually get selected as the main block on the blockchain. In UBA, the attacker submits all its withheld blocks at the end of the round (when any block gets propagated). This strategy is different from FAW in that the attacker still submits the withheld block in the events A and B, as opposed to not submitting it as in BWH/FAW attack. This results in the attacker receiving two blocks worth (one main block and one uncle block) of rewards. In addition, uncle-block attack increases the reward in the event C because the block which does not become the main block (corresponding to the $1-c$ probability in Section 5.1) also gets rewarded by the uncle reward.

Uncle block rewards (rewarding multiple blocks in a round) results in greater complexity in the mining game and yields the vulnerability for UBA. For example, one can envision an attacker continue to mine using the same block header until it exhausts the reward in that round (although this turns out to be suboptimal if uncle block has less rewards than the main blocks according to our analyses). In the rest of the section, we introduce the UBA strategy, establish that a non-attacker who does not control/vary the timing of the block submissions is incentivized to move on to the next round (as opposed to staying within the round for uncle rewards), and analyze the optimal miner strategy for the attacker controlling the timing of the submissions. Using these insights, we also quantify the expected reward of UBA and provide a lower bound that is independent of networking conditions (c), which is in contrast to FAW.

6.1 Uncle-Block Model: κ and λ

Suppose the blockchain provides non-zero rewards up to λ uncle blocks and the reward per uncle block is κ. These are blockchain system parameters, for example, $\lambda = 2$ and $\kappa = \frac{7}{8}$ in Ethereum. In practice, these parameters have the following constraints: $\lambda < \infty$ limits the number of rewardable uncle blocks, and $\kappa < 1$ bounds the uncle reward so that the main block is more valuable than the uncle blocks. In each round, the blockchain will provide rewards which can range from 1 (no fork and thus no uncle blocks) to $1 + \lambda\kappa$ (1 main block and λ uncle blocks get rewarded), normalized with respect to the main-block reward.

As a trivial result, UBA reduces to FAW if there is no uncle reward, i.e., $\lambda = 0$ or $\kappa = 0$. Therefore, we focus on the case when there is a positive uncle reward, i.e., $\lambda > 0$ and $\kappa > 0$.

6.2 In Main Pool: Advance with New Block

In this section, we study the miner strategy in its main pool (no infiltration/gaming against other pool members, for example, solo member in the pool). In its main pool, a miner who found a block can still mine from the same block header instead of advancing to the next round with the updated block. Despite the option, the following lemma shows that the miner will be incentivized not to do so and follow the consensus protocol to update the PoW header/execution and move on to the next round if the main block yields greater reward than the uncle blocks. In other words, a miner will choose to advance to the next round rather than mining for the uncle block within the same round if $\kappa < 1$.

Lemma 1. *The optimal miner strategy is the honest mining (following the protocol and advancing to the next round) in its main pool if* $\kappa < 1$.

Proof. The inter-arrival time between the block findings follows an exponential distribution and therefore the block finding process is memoryless, i.e., given that the block is not found, the expected time to find a block is the same as before. Therefore, the miner gains greater reward by mining for the main block and not for the uncle block if $\kappa < 1$, and the reward-optimal mining strategy is to update the block header whenever available.

6.3 Uncle-Block Attack Reward Analyses

We compute the expected reward of UBA, aggregated across the attacker-involved pools and normalized by the reward amount, to show the effectiveness of the attack. Assuming that the attacker updates the round/block header if it finds a block in its honest-mining main pool (as is established to be reward-optimal in Lemma 1), the following theorems provide insights about the reward of UBA.

Theorem 1. *UBA outperforms FAW attack always (i.e., regardless of* α, β, τ, c, *and* κ*) and its reward is greater than that in Equation 4.*

Proof. In all events which provide positive reward to the attacker, A, B, and C, finding an extra block within the same round (using the same block header) only increases the attacker reward if the uncle reward is positive ($\kappa > 0$). The attacker behavior/strategy changes from FAW in events A and B, and UBA increases the reward from FAW (Equation 3) in all three event cases. In event A (when the attacker finds a block in its honest-mining main pool), the attacker rewards increases from FAW because there is a positive probability of $\tau\alpha \cdot \frac{(1-\tau)\alpha}{1-\tau\alpha}$ that the attacker has been withholding another block in the victim pool (since the main pool mining and the victim pool mining are independent). In such case, the attacker has the control over both blocks and will prioritize the block in the main pool by broadcasting the block in the main pool before the one in the victim pool because, in contrast to the attacker's solo mining in the main

$$
\begin{aligned}
R_{\mathrm{UBA}} &= \mathrm{E}[R|A]\cdot\Pr(A) + \mathrm{E}[R|B]\cdot\Pr(B) + \mathrm{E}[R|C]\cdot\Pr(C) \\
&\geq (1+\kappa\frac{\tau\alpha}{\beta+\tau\alpha}\cdot\tau\alpha)\cdot\frac{(1-\tau)\alpha}{1-\tau\alpha} + (1+\kappa\cdot\tau\alpha)\frac{\tau\alpha}{\beta+\tau\alpha}\cdot\frac{\beta}{1-\tau\alpha} + [1\cdot c+\kappa\cdot(1-c)]\,\frac{\tau\alpha}{\beta+\tau\alpha}\cdot\tau\alpha\frac{1-\alpha-\beta}{1-\tau\alpha} \\
&= \frac{(1-\tau)\alpha}{1-\tau\alpha} + \frac{\tau\alpha}{\beta+\tau\alpha}\left(\kappa\frac{(\tau\alpha)^2}{\beta+\tau\alpha}\cdot\frac{(1-\tau)\alpha}{1-\tau\alpha} + (1+\kappa\tau\alpha)\frac{\beta}{1-\tau\alpha} + [c+(1-c)\kappa]\,\tau\alpha\frac{1-\alpha-\beta}{1-\tau\alpha}\right) \quad (4) \\
&\geq \frac{(1-\tau)\alpha}{1-\tau\alpha} + \frac{\tau\alpha}{\beta+\tau\alpha}\left(\kappa\frac{(\tau\alpha)^2}{\beta+\tau\alpha}\cdot\frac{(1-\tau)\alpha}{1-\tau\alpha} + (1+\kappa\tau\alpha)\frac{\beta}{1-\tau\alpha} + \kappa\tau\alpha\frac{1-\alpha-\beta}{1-\tau\alpha}\right) \quad (5)
\end{aligned}
$$

pool, the reward in the victim pool gets shared by the other pool members (in which case the uncle reward is $\kappa\frac{\tau\alpha}{\beta+\tau\alpha}$). In event B (when another miner in the victim pool finds a block), the attacker could have also been withholding a block in the victim pool, which occurs with a probability of $\tau\alpha\cdot\frac{\beta}{1-\tau\alpha}$, in which case the attacker gets rewarded for both the main block and the uncle block (regardless of which block wins the fork racing) increases to $(1+\kappa)\frac{\tau\alpha}{\beta+\tau\alpha}$ from just discarding/withholding the found block as in FAW/BWH yielding $\frac{\tau\alpha}{\beta+\tau\alpha}$ in Equation 3. In the event C (when the attacker has been withholding a block in the victim pool and a third-party miner finds a block), unlike the FAW attack which disregards uncle blocks, even if the attacker's block does not become the main block with a probability of $(1-c)\cdot\tau\alpha\frac{1-\alpha-\beta}{1-\tau\alpha}$, it still gets rewarded by the uncle reward of κ. Putting it together, Equation 4 presents the expected reward for the UBA attack. The expected reward for launching UBA in Equation 4 is greater than the FAW reward in Equation 3 for any $\alpha,\ \beta,\ \tau,\ c$.

Theorem 1 provides a lower bound of the UBA's reward performance in Equation 4, which is greater than the FAW reward performance, providing greater unfair reward and incentives for the attackers to conduct the attack. Equation 4 is a lower bound because we ignore the cases when the attacker-controlled victim pool finds multiple blocks before the others do within the same round (which provides positive reward increment if $\lambda > 1$). The bound is rather tight and close to the actual expected reward because the ignored reward cases has exponentially decreasing probabilities; more specifically, the probability of the attacker finding x number of blocks in the victim pool within a round grows by $(\tau\alpha)^x$.

Theorem 2. *UBA's reward is greater than Equation 5, which is independent of c, if $\kappa \leq 1$.*

Proof. From Equation 4, by assuming that $\kappa \leq 1$ (i.e., the uncle reward is not greater than the main reward), $c + (1-c)\kappa \geq c\kappa + (1-c)\kappa = \kappa$, which yields Equation 5. The inequality becomes an equality if $\kappa = 1$ (the uncle reward and main reward have the same reward amount) or $c = 0$ (the attacker's fork-causing block always loses to the third-party block submissions).

Theorem 2 presents the UBA's reward performance which is independent of c, which is a critical environmental parameter in FAW and depends on the networking topology and conditions of the attacker. (For example, FAW attack converges to BWH if $c \to 0$.) The magnitude of κ determines how close the approximation is to the actual reward (the greater the κ the tighter the approximation).

We use Equation 4 and Equation 5 for our simulation analyses and to quantify the reward performances with actual numerical values. Despite using the lower bound, we show that UBA's effectiveness and impact are significantly superior to the previously known withholding-based attacks, which state of the art is FAW.

To further maximize its reward, an attacker can dynamically control its control parameter τ since if it can know/estimate the rest of the environmental parameters in $\alpha, \beta,$ and c and the system parameter in κ. The attacker can optimize its τ either by solving it mathematically to check for local maximum ($\frac{d\bar{R}_{uba}}{d\tau} = 0$) or by using an optimization algorithm (e.g., gradient-descent algorithm). To present a stronger threat, we assume such capability for the attacker, e.g., the attacker can correctly estimate $\alpha, \beta,$ and c (e.g., α and β are publicly accessible and Equation 5 enables optimization even if c cannot be estimated) and study the impact of the attacker dynamically adapting its τ to improve its reward in Section 7.2; we call the attacker's reward when using the optimal τ *τ-capacity.*

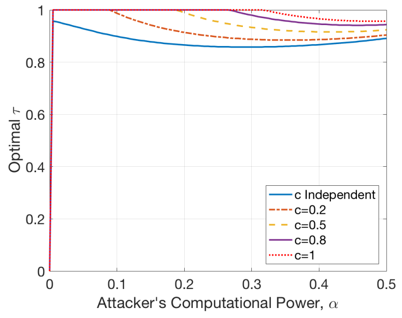
7 Simulation Analyses

7.1 Simulation Setup and Parameters

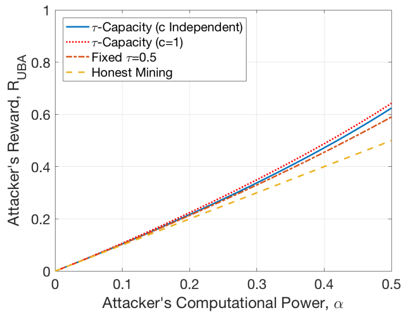
We analyze UBA using Monte-Carlo simulations and numerical analyses. Our model introduces environmental parameters (α, β), attacker's control parameters (τ), and the blockchain system control parameters (κ, λ). We focus our analyses from the attacker's perspective (observing the attacker's reward) and thus vary the attacker-controlled parameters (α, τ). This section explains the simulations settings and the parameter choices to characterize the blockchain system and the victim pool system under attack.

Our blockchain system simulation setup is influenced by modern blockchain implementations, e.g., $\kappa = \frac{7}{8}$ as is in Ethereum (the second largest cryptocurrency behind Bitcoin). We set $\lambda = 1$ which is the worse-case for the attacker's reward; any other reward-distribution algorithm supporting $\lambda > 1$, e.g., Ethereum, provides greater reward to the attackers and therefore provides them with greater incentives/impacts to conduct UBA. For the pool system, $\beta = 0.24$, which value corresponds to the strongest mining pool in real-world mining at the time of this writing [2]. The attacker attacking the stronger pool as its victim pool (as opposed to a weaker pool of $\beta \to 0$) provides greater reward and is aligned with its incentive, which we verify in our simulations and agree with previous literature [14, 1]. These parameters are fixed unless otherwise noted (we vary the variables to analyze the dependency and the impacts).

We also consider the 51% attack where the attacker can fully control the blockchain if the attacker's computational power exceeds the 50% of the net-



(a) Optimal τ for achieving τ-capacity



(b) Dynamic strategy (τ-capacity) vs. fixed strategy (no adaptation of τ), including honest mining

Fig. 1: Uncle-Block Attack (UBA) Analyses

work's. In our context, the 51% attacker can conduct withholding-based selfish mining to reverse the transactions/blocks on the chain and to waste the other miner's computational resources on blocks which the attacker can reverse and make stale. Therefore, we limit our analyses to $0 \leq \alpha \leq 0.5$ (the attacker is capable of 51% attack if $\alpha > 0.5$) in addition to the constraint of $\alpha + \beta \leq 1$ from the definitions of α and β.

7.2 UBA: Dynamic Control and τ-Capacity

In this section, we analyze UBA, which generalizes FAW and BWH. The attacker controls τ and can dynamically adapt it to the environment to maximize its reward. We call such τ the *optimal* τ and the corresponding reward is the τ-*capacity*. Figure 1(a) plots the optimal τ with respect to the attacker's power capability α while varying c. The attacker's optimal τ quickly increases, i.e., the attacker spends more power on infiltration and on the victim pool and less on the honest-behaving main pool, when the attacker has computational power and participates in mining ($\alpha > 0$). With greater c, the attacker's power on the victim pool increases, and there are more α-cases when the attacker's optimal τ is equal to one and the attacker only mines on the victim pool (no power allocated on the main pool). $c = 0$ (i.e., the attacker always loses the forking race against the third-party block submission) represents the worst-case in c and yields the "c independent" reward in Equation 5. Despite this lower bound and the unfavorable networking condition to the attacker, the attacker still focuses larger amount of its power on the infiltration of the victim pool for maximizing its reward. As long as the attacker has non-zero power ($\alpha > 0$), the attacker spends more than 85.8% of its power on infiltration regardless of c.

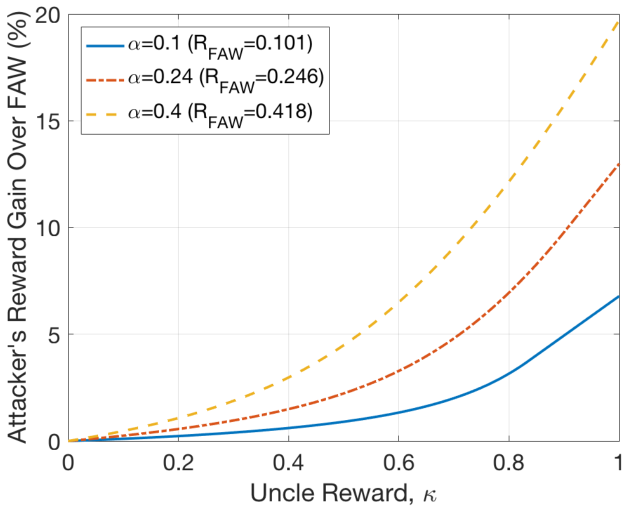


Fig. 2: UBA reward gain over FAW (the legend box includes the FAW reward magnitude corresponding to 0% gain)

Figure 1(b) plots the attacker's reward assuming dynamic τ control (τ-capacity), fixed τ control, and honest mining (no withholding of blocks and shares). The attacker's reward increases with c, and varying c results in reward values between the lowest in $c = 0$ (yielding c-independent reward in Equation 5) and the highest in $c = 1$; for example, if $\alpha = 0.1$, then the reward increases from 0.101 at $c = 0$ to 0.108 at $c = 1$ and, if $\alpha = 0.24$, then the reward increases from 0.250 at $c = 0$ to 0.278 at $c = 1$ (not shown in the plot). The dynamic strategy choosing the optimal τ and reaching the τ-capacity always outperforms the fixed strategy by definition of τ-capacity; to provide a representative fixed-strategy reward analysis distinguishable in plots, Figure 1(b) includes data with the suboptimal fixed strategy of $\tau = 0.5$ (half of the power on the main pool and the other half on the victim pool). Also, the uncle-block attack always outperforms the honest mining, which corresponds to $\tau = 0$ (all computational power in the main pool and no withholding). Moving forward, we assume that the attacker is capable of dynamically adapting τ and achieving the τ-capacity, which requires the correct estimation of α and β (e.g., the hash rate estimations are publicly available, e.g., [2]).

7.3 UBA's Impact Beyond the State of the Art

To compare UBA with FAW, we analyze the *reward gain* of UBA over FAW, which is the difference between the rewards of the two attacks in %. As shown in Figure 2, the reward gain increases with the uncle reward of κ. As the uncle reward κ increases, UBA provides greater reward to the attacker and the reward gain (the difference from the FAW) also increases; in fact, the UBA's reward converges to that of FAW when $\kappa \to 0$. The reward gain also increases with the attacker's power capability of α both in its magnitude and in the baseline of the FAW attack reward (0% reward gain). In other words, not only the FAW reward itself increases with α but also the UBA's reward gain with respect to the FAW increases; Figure 2 plots the results when $\alpha = 0.1$, $\alpha = 0.24$ (which is equal to β), and $\alpha = 0.4$.

8 Discussions for Potential Countermeasures

While our work introduces a new vulnerability/threat and sheds negative lights on the uncle block system component of blockchain, we do not recommend blindly discarding the uncle-block mechanism since it serves a useful purpose in increasing fairness among the miners, especially for those experiencing networking discrepancies. Rather, we intend to inform the blockchain researchers and developers for their parameter decisions and future design constructions. Replacement mechanisms and their incorporation to the overall blockchain system remains an open challenge for blockchain research and development. In this section, we list the potential countermeasure ideas, which we consider to be of relatively lower overheads for deployment except for those in Sections 8.5 and 8.2, to encourage research in preventing and mitigating UBA and other withholding-based threats; the actual development, the system incorporation, and the effective analyses against UBA are outside the scope of this paper.

8.1 Detection Based on Reward Behavior

The withholding-based threats result in abnormal reward behaviors. For example, FAW or UBA increases the forking probability at the blockchain level. Such phenomenon can be sensed and measured for attack detection, which can be then used for mitigation purposes. While we identify behavior-based detection as promising, we do not recommend relying on identity-based detection and mitigation/blacklisting mechanisms, since the identity control is virtually nonexistent and it is cheap for the attacker to generate multiple identities/accounts (Sybil attack) by the design of the permissionless blockchains.

8.2 Block Obfuscation and Oblivious Share

Building on the commit-and-reveal/commitment framework in cryptography, *oblivious share* deprives the miner of the knowledge of whether a share is a block (a share fulfilling a harder requirement) or a share until the miner submits the share [18, 11]. The attacker (malicious or selfish miner) therefore cannot dynamically adopt the withholding-based threats which require distinguishing the share and the block before submission. While effective against the withholding-based attacks, such approach requires a protocol change (an additional exchange between the mining pool manager and the miners) and is not backward compatible (does not work with the existing system unless the protocol change/update is made) [14, 15], causing protocol/communication overheads and making such schemes undesirable for implementation to the blockchain network (which includes closed pools and solo miners, free of withholding vulnerabilities).

8.3 Payout and Reward Function Control

A low-overhead countermeasure is to modify the payout and reward functions, for example, controlling the system parameters of κ and λ in our model. For example, Bag and Sakurai proposes "special reward" for the block submissions [1] to distinguish between block submissions and share submissions (different weights) against block-withholding attack; introducing such parameter can also be useful against UBA.

8.4 Share Timestamping

Timestamping the shares to ensure that the submission order is the same as their finding [7] can help mitigating the withholding-based attacks. Such mechanism can prevent from further share submissions after withholding blocks in UBA and FAW (since the later shares become invalid if they are detected that they have been found after the block) or aid the abnormal-detection mechanisms in Section 8.1 by providing crucial timing/order information. SmartPool [16] also uses timestamping to order/sort the shares but for different purpose (for its data structure and to prevent duplicate share submissions).

8.5 Mining Pool Unification

A radical solution to the withholding-based threats is to have one mining pool only so that there is a single mining pool for the miners to join. Having one mining pool for the entire blockchain network eliminates the notion of sabotaging/victimizing a pool. A useful platform for implementing such solution can be distributed mining pools, e.g., SmartPool [16] and P2Pool, which eliminates the centralized mining pool manager and replaces it with a distributed program/computing, motivated to make the blockchain computing more decentralized without the reliance on trusted third party (the mining pool manager in this case) [12, 4]. In fact, SmartPool [16] envisions that their platform can be used to unify the mining pools citing that the elimination of the mining pool fee (charged by the centralized mining pool manager for their services) and the reduced variance (compared to independent mining) will provide the incentives for such vision. However, despite such desirable properties, it can be difficult to enforce the change in behaviors in the miners and have all miners mine at the designated pool, especially for an existing blockchain implementation with the existing miners having already joined a pool; enforcing such pool restriction for the miners can also be viewed as a violation of the freedom of the miners and is not backward-compatible to the existing miners.

9 Conclusion

Blockchain uses PoW-based mining and mining pools to achieve consensus in cryptocurrencies. From real-world blockchain and mining pool system implementations, we discover that uncle blocks can be used for exploits by a rational and uncooperative attacker and introduce the uncle-block attack (UBA) strategy. UBA advances and outperforms the prior withholding-based attack strategies in FAW and BWH and is effective even in the case when the attacker has an unfavorable networking environment (which limited the effectiveness of the previous state of the art attack strategy of FAW). Since UBA has the same attack setup requirements as BWH (which already have reported incidents in the real-world implementations), the more impactful UBA presents greater risk against vulnerable blockchain systems.

While our work introducing the novel UBA threat strategy and focusing on the threat/impact analyses raises issues on the current blockchain system components (e.g., mining pool and uncle block), we do not recommend blindly discarding them since they serve useful purposes. Rather, we intend to inform the blockchain researchers and developers for their parameter decisions and future design constructions. Countering UBA and other withholding threats and depriving the rational attackers of the threat incentives remain open challenge for blockchain research and development.

References

1. S. Bag and K. Sakurai, "Yet another note on block withholding attack on bitcoin mining pools," in *ISC*, ser. Lecture Notes in Computer Science, vol. 9866. Springer, 2016, pp. 167–180.
2. blockchain.com. Hash rate distribution. [Online]. Available: https://www.blockchain.com/en/pools
3. ——. Market capitalization. [Online]. Available: https://www.blockchain.com/charts/market-cap
4. J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "Sok: Research perspectives and challenges for bitcoin and cryptocurrencies," in *2015 IEEE Symposium on Security and Privacy*, May 2015, pp. 104–121.
5. V. Buterin, "Ethereum: A next-generation smart contract and decentralized application platform," https://github.com/ethereum/wiki/wiki/White-Paper, 2014, accessed: 2016-08-22. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper
6. M. Carlsten, H. Kalodner, S. M. Weinberg, and A. Narayanan, "On the instability of bitcoin without the block reward," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: ACM, 2016, pp. 154–167. [Online]. Available: http://doi.acm.org/10.1145/2976749.2978408
7. S.-Y. Chang and Y. Park, "Silent timestamping for blockchain mining pool security," *IEEE International Workshop on Computing, Networking and Communications (CNC)*, 2019.

8. CoinMarketCap. Top 100 cryptocurrencies by market capitalization. [Online]. Available: https://coinmarketcap.com
9. I. Eyal, "The miner's dilemma," in *Proceedings of the 2015 IEEE Symposium on Security and Privacy*, ser. SP '15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 89–103. [Online]. Available: https://doi.org/10.1109/SP.2015.13
10. I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," *CoRR*, vol. abs/1311.0243, 2013.
11. ——, "How to Disincentivize Large Bitcoin Mining Pools," June 2014.
12. A. Gervais, G. O. Karame, V. Capkun, and S. Capkun, "Is bitcoin a decentralized currency?" *IEEE Security Privacy*, vol. 12, no. 3, pp. 54–60, May 2014.
13. A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: ACM, 2016, pp. 3–16. [Online]. Available: http://doi.acm.org/10.1145/2976749.2978341
14. Y. Kwon, D. Kim, Y. Son, E. Vasserman, and Y. 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*, ser. CCS '17. New York, NY, USA: ACM, 2017, pp. 195–209. [Online]. Available: http://doi.acm.org/10.1145/3133956.3134019
15. L. Luu, R. Saha, I. Parameshwaran, P. Saxena, and A. Hobor, "On power splitting games in distributed computation: The case of bitcoin pooled mining," in *2015 IEEE 28th Computer Security Foundations Symposium*, July 2015, pp. 397–411.
16. L. Luu, Y. Velner, J. Teutsch, and P. Saxena, "Smartpool: Practical decentralized pooled mining," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, 2017, pp. 1409–1426. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/luu
17. S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
18. M. Rosenfeld, "Analysis of bitcoin pooled mining reward systems," *CoRR*, vol. abs/1112.4980, 2011. [Online]. Available: http://arxiv.org/abs/1112.4980
19. A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal selfish mining strategies in bitcoin," in *Financial Cryptography and Data Security*, J. Grossklags and B. Preneel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, pp. 515–532.
20. I. Tsabary and I. Eyal, "The gap game," *CoRR*, vol. abs/1805.05288, 2018.
21. G. Wood, "Ethereum: A secure decentralised generalised transaction ledger eip-150 revision (759dccd - 2017-08-07)," 2017, accessed: 2018-05-12. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf