Pitchforks in Cryptocurrencies:

Enforcing rule changes through offensive forking- and consensus techniques

Aljosha Judmayer[1,2], Nicholas Stifter[1,2], Philipp Schindler[1], and Edgar Weippl[2]

[1] SBA Research, Vienna, Austria
`(firstletterfirstname)(lastname)@sba-research.org`
[2] University of Vienna, Vienna, Austria {`firstname.lastname`}`@univie.ac.at`

Abstract. The increasing number of cryptocurrencies, as well as the rising number of actors within each single cryptocurrency, inevitably leads to tensions between the respective communities. As with open source projects, (protocol) forks are often the result of broad disagreement. Usually, after a permanent fork both communities "mine" their own business and the conflict is resolved. But what if this is not the case? In this paper, we outline the possibility of malicious forking and consensus techniques that aim at destroying the other branch of a protocol fork. Thereby, we illustrate how merged mining can be used as an attack method against a permissionless PoW cryptocurrency, which itself involuntarily serves as the parent chain for an attacking merge mined branch of a hard fork.

Keywords: Merged Mining · Cryptocurrencies · Game Theory

1 Introduction

The concept of forking is a core design property of Nakamoto consensus (NC). Even without the interference of malicious actors, the proposal of a block by two different miners at approximately the same point in time can trigger a fork. Such a fork eventually resolves as soon as new blocks on top of either branch are found and propagated. Apart from these temporary forks that are resolved when one branch takes the lead, there also exists the concept of a permanent fork, also referred to as a *chain split*. In such a case, the end result are two different systems. This can happen intentionally or unintentionally, for example, when the underlying protocol has to be updated.

In this context, the loosely defined terms *hard fork* and *soft fork* have established themselves as descriptors of different classes of upgrade mechanisms for the underlying rules.The term *hard fork* has established itself [3, 7] as a descriptor for protocol changes that can incur a permanent split of the blockchain, as they permit or even enforce the creation of blocks that are considered invalid under previous protocol rules. In contrast to this, *soft forks* intend to retain some level of compatibility with older protocol versions, specifically towards clients adhering to previous protocol rules. Apart from these two more frequently used types of forks, also an approach termed *velvet fork*, which expands upon the concept of a soft fork, was outlined by Kiayias et al. [9]. Specifically, velvet forks intend to avoid the possibility of disagreement by a change of rules through rendering modifications to the protocol backward compatible *and* inclusive to legacy blocks. So, in contrast to temporary forks, which can happen during normal operation, hard-, soft- and velvet forks refer to a protocol upgrade that might or might not lead to a permanent fork, i.e., a chain split.

We expand upon the concept of different protocol changes by discussing the potential security implications that can arise from permanent hard forks (protocol changes) that are interleaved with the original protocol/system in a malicious way. We describe the *pitchfork* as an example to enforce rule changes in a targeted cryptocurrency, denoted as Π. In a pitchfork, the attacker creates a different protocol version Π' and uses techniques from merged mining to interleave his new hard forked cryptocurrency with the targeted parent chain Π s.t., mining "bad blocks" in the parent chain is a prerequisite for being accepted as a valid PoW in the new cryptocurrency which consists of the pitchfork child chain Π'.

Our construction highlights some interesting questions. In particular, with regards to the underlying (game-theoretic) incentive model, such attacks can lead to negative side effects in permissionless cryptocurrencies based on NC.

2 System Model and Goals of a Pitchfork

The increasing number of cryptocurrencies, as well as the rising number of actors within every single cryptocurrency, inevitably leads to tensions between the respective communities. As with open-source projects, (protocol) forks are often the result of broad disagreement. Usually, after a permanent fork (chain split), both communities "mine" their own business and the conflict is resolved. But what if this is not the case? In this paper, we outline the possibility of malicious forking and consensus techniques that aim at destroying the other branch of a protocol hard fork (protocol update). Thereby, we illustrate how merged mining can be used as an attack method against a permissionless PoW cryptocurrency, which itself involuntarily serves as the parent chain for an attacking merge mined branch of a hard fork.

Merged mining is already known for posing a potential issue to the child cryptocurrencies, for example demonstrated in the case of CoiledCoin[3]. However, so far, no concrete example has been given that merged mining can also pose a risk to the parent chain. Since (parent) cryptocurrencies can not easily prevent being merge mined[4], an attack strategy using this approach would be applicable against a variety of permissionless PoW cryptocurrencies. In this paper, we describe a scenario where merged mining is used as a form of attack against a parent chain in the context of a hostile protocol fork.

[3] Cf. https://bitcointalk.org/index.php?topic=56675.msg678006#msg678006

[4] The inclusion of a hash value within a block to provably attributed it to the creator of the proof-of-work (PoW) is enough to support merged mining [8]

2.1 Actors

For our attack scenario, we assume a permissionless PoW-based cryptocurrency Π, whose miners cannot agree on whether or not to change the consensus rules. Some of the miners want to adapt the consensus rules in a way such that newly mined blocks may not be valid under the old rules, i.e., perform a hard fork, or protocol update. Thereby, we differentiate between the following actors:

- **A**ntiquated or backward-compatible miners ($\mathcal{A}$): The fraction of miners (with hashrate $p_{\mathcal{A}}$) in a currently active cryptocurrency Π that does not want to change the consensus rules of Π.
- **B**yzantine, or change enforcing miners ($\mathcal{B}$): The fraction of miners (with hashrate $p_{\mathcal{B}}$) in a currently active cryptocurrency Π, that wants to change the consensus rules, i.e., perform a hard fork/protocol update. Moreover, they want that only their branch of the fork survives.
- **I**ndifferent, independent, or neutral miners ($\mathcal{I}$): The set of miners (with hashrate $p_{\mathcal{I}}$) that has no hard opinion on whether or not to change the consensus rules. They want to maximize their profits and act economically rational to achieve this goal, with the limitation that they want to avoid changes as far as possible. If there is no imminent need that justifies the implementation costs for adapting to changes, they will not react[5].

For our example, we assume that $\mathcal{B}$ wants to increase the block size, while $\mathcal{A}$ does not want to implement any rule change. The goal of the attackers in $\mathcal{B}$ is twofold: 1) *Enforce* a change of the consensus rules in the target cryptocurrency. 2) *Disrupt* the operation of the old branch of the hard forked target cryptocurrency (Π), which does not follow the new consensus rules Π'.

2.2 Characteristics of a Pitchfork

For this paper, we are only interested in forking scenarios that are *not bilateral*. In a bilateral fork, conflicting changes are intentionally introduced to ensure that two separate cryptocurrencies emerge [16]. An example of such a scenario would be the changed chain ID between Ethereum and Ethereum Classic. It is commonly believed that in a non-bilateral forking event, the only reliable possibility to enforce a change requires that the majority of the mining power supports the change. Thereby, two main cases can be distinguished according to [16]:

If the introduced change *reduces* the number of blocks that are considered valid under the new consensus rules, all newly mined blocks are still considered valid under the old rules, but some blocks which previously would have been considered valid are no longer considered valid under the new rules. An example of such a scenario would be a *block size decrease*. In this case, the first goal (*enforce*) of our attack is easy to achieve if $p_{\mathcal{B}} > p_{\mathcal{A}} + p_{\mathcal{I}}$ holds since any fork

[5] This should capture the observation that not all miners immediately perform merged mining if it is possible, even though it would be rational to do so [8].

introduced by $p_{\mathcal{B}}$ will eventually become the longest chain and be adopted by $p_{\mathcal{A}}$ and $p_{\mathcal{I}}$ because of the longest heaviest chain rule. Therefore, in this case, the rules have been successfully changed from Π to a subset Π' since $p_{\mathcal{A}}$ is the minority in this case. If $\mathcal{A}$ decides to continue a cryptocurrency under the original rules Π, such that larger blocks are again possible, they have to declare themselves as a new currency since the original one has been overtaken and the rules changed to a subset. Therefore, the goal to *enforce* is clearly reached in such a case. However, the *disruption* goal cannot be reached directly if all miners in $\mathcal{A}$ create a new cryptocurrency that follows the original rules Π.

If the introduced change *expands* the set of blocks that are considered valid under the new consensus rules, then some blocks following the new rules will not be considered valid under the old rules. Therefore, any mined block that is only valid under the new rules will cause a fork. An example of such a scenario would be a *block size increase*[6].In this case, a permanent hard fork will only occur if the chain containing blocks following the new rules grows faster, i.e., $p_{\mathcal{B}} > p_{\mathcal{A}} + p_{\mathcal{I}}$ holds. The result would be that the forking event creates two different currencies: cryptocurrency Π', which includes big blocks, and cryptocurrency Π, which forked from the main chain after the first big block. Therefore, again the *disruption* goal cannot be reached directly. To reach this goal, some miners in $\mathcal{B}$ could be required to switch to the original cryptocurrency Π and disrupt its regular operation, e.g., by mining empty blocks. This, of course, has the drawback that the respective attacking miners that switched from Π' to Π do not gain any profits in Π', and their rewards in Π will be worthless if they succeed in rendering Π unusable.

The *pitchfork* attack method proposed in this paper aims to achieve both attack goals simultaneously, even in cases where $p_{\mathcal{B}} < p_{\mathcal{A}} + p_{\mathcal{I}}$ holds.

3 Pitchfork Attack Description

The basic idea of a pitchfork attack is to use merged mining as a form of attack against the other branch of a fork in a permissionless PoW cryptocurrency resulting from a disputed consensus rule change. The pitchfork should disrupt the normal operation of the attacked branch to such an extent that the miners abandon the attacked branch and switch to the branch of the hard fork, which performs merged mining and follows the new consensus rules. We call the cryptocurrency up to the point of the fork *ancestor* cryptocurrency $\bar{\Pi}$. After the forking event, the cryptocurrency which still follows the same rules is denoted as Π, whereas the *change enforcing* cryptocurrency branch that uses merged mining and the new consensus rules is denoted as Π'.

To execute the attack, the new merge mined branch Π' accepts valid *empty blocks* of Π as a PoW for Π'. In the nomenclature of merged mining, the chain Π, which should be attacked, is called the *parent chain*, and chain Π' is called the *child chain.* For a valid parent block b of Π, the following additional requirements

[6] Our example, in which $\mathcal{B}$ wants to increase the block size and $\mathcal{A}$ does not want to implement any rule change, would resemble such an *expanding* protocol change.

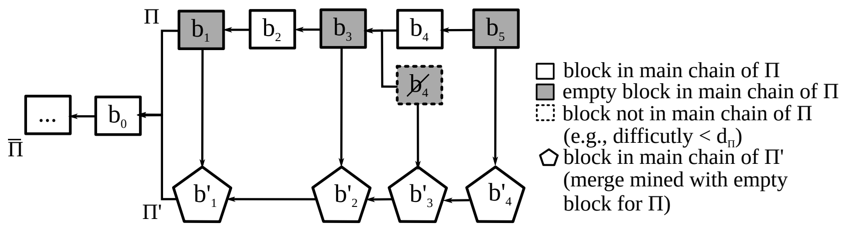
Π
b1
b2
b3
b4
b5
b4
...
b0
Π̄
Π'
b'1
b'2
b'3
b'4
block in main chain of Π
empty block in main chain of Π
block not in main chain of Π (e.g., difficutly < dΠ)
block in main chain of Π' (merge mined with empty block for Π)

Fig. 1. Example of blocks mined after the hard fork into two cryptocurrencies Π and $\Pi' u$.

need to be satisfied: i) The block b has to be empty. Therefore, the contained Merkle tree root in the header of the respective block must only include the hash of the (mandatory) coinbase transaction. Given the corresponding coinbase transaction, it can then be verified that b is indeed empty. ii) The coinbase transaction of b must include the hash of a valid block b' for Π'. The header of block b' contains a Merkle tree root with the actual transactions performed in Π'.

Figure 1 shows the two cryptocurrencies after the fork. The last block in the ancestor cryptocurrency $\bar{\Pi}$ before the forking event is b_0. The first empty block that is merge mined is b_1 in this example. This block (b_1) is valid under the old rules and fulfills the difficulty target in Π, except that it is empty. Moreover, the block b_1 was mined by a miner in $\mathcal{B}$, which happens with probability $p_{\mathcal{B}}$, and contains the hash of block b'_1 in its coinbase. Therefore b_1 serves as a valid PoW for Π' as well. Block b_2 was not mined by a miner in $\mathcal{B}$, which happens with probability $1 - (p_{\mathcal{B}} + p_{\mathcal{I}})$, and therefore it is not empty and does not contain a hash for a valid block for Π' in its coinbase. This shows that the two chains are not necessarily synchronized regarding their number of blocks. The block interval in Π' depends on the difficulty target of Π'. Since we assume that the attacker does not control the majority of the hashrate ($p_{\mathcal{B}} < p_{\mathcal{A}} + p_{\mathcal{I}}$), the difficulty D in Π' should be lower than in Π at the beginning of the attack, i.e., $D_{\Pi'} < D_{\Pi}$ holds. If the difficulty has been adjusted in Π', then the overall number of blocks should be approximately the same for both chains. In such a case, there might be empty blocks such as $\cancel{b_4}$, which do fulfill the difficulty target for Π' but not for Π. Still, if $D_{\Pi'} < D_{\Pi}$ holds, then over time, a fraction of all blocks in Π corresponding to $p_{\mathcal{B}}$ will be mined by a miner in $\mathcal{B}$. If we assume that $p_{\mathcal{B}} \approx 0.33$, then approximately every third block in Π should be empty.

PoW Difficulty: Theoretically, it would be possible that Π' requires the same or even a higher difficulty than Π. If $D_{\Pi'} \geq D_{\Pi}$, then chain Π' would contain fewer blocks than chain Π. This, of course, would have a negative effect on the latency in chain Π', i.e., the time it takes till a transaction is confirmed. However, any merge mined blocks that meet the difficulty requirement of $D_{\Pi'}$ will be considered valid in Π. For example, when $D_{\Pi'} = D_{\Pi}$, the number of blocks in Π' relative to Π would only correspond to the fraction of the hashrate ($p_{\mathcal{B}}$)

that performs merged mining. Nevertheless, since chain Π' increased the block size, the throughput could theoretically remain the same or even be higher than in chain Π (depending on the actual implementation). Some examples regarding an increased block size are discussed in [6, 4]. Alternatively, Bitcoin-NG [5] could also be applicable. The latter approach would have the added benefit that the negative impact on latency and confirmation times is mitigated. To illustrate our attack, it is not of particular relevance which adaptation is used to increase the throughput in Π'.

3.1 Effects of the Attack

In the simplest case, if no countermeasures are taken by the chain under attack, a pitchfork reduces the throughput of the target chain Π by the number of empty blocks corresponding to the hashrate of the attackers ($p_{\mathcal{B}}$). Considering the limited block size in Π and events in Bitcoin, or other cryptocurrencies, where the number of unconfirmed transactions in the mempool peaked, a hashrate of $p_{\mathcal{B}} \approx 0.33$ mining solely empty blocks, would likely have an impact on the duration of such periods of congestion, and hence also transaction fees and confirmation times. This could sway both users and miners in $\mathcal{I}$ to switch to the attacking chain Π', which further reinforces the attack. Two other advantages of the attack are that it is *pseudonymous* and that the risk in terms of currency units in Π, as well as its severeness, is *parameterizable.*

Pseudonymous: Since the pitchfork attack is executed by miners through producing new blocks that are, in addition, merged mined with the attacking chain, it is in theory possible to hide the identities of the attackers because no unspent transaction outputs need to be involved in the attack that could have a traceable history. However, additional care needs to be taken by these miners to ensure that their identity is not inadvertently revealed through their behavior [8].

Parameterizable: The attack is not an all-in-move, and its costs, in terms of currency units in Π, can be parameterized. The goal of the attack is to disrupt the original chain Π, but if this fails, the attackers may not lose much. Due to merged mining, the main costs of a failed attack result from the foregone profits from transaction fees that are not collected in chain Π. Additional costs created by merged mining, i.e., running an additional full node for chain Π can be negligible compared to the overall costs related to mining [?]. Moreover, even a failed attack on Π can still be profitable for the attacking miners since the attackers in $\mathcal{B}$ are early adopters of Π'. If the value of the newly created cryptocurrency Π' increases enough, the additional income may not only compensate for the reduced income from mining empty blocks in Π but could even create a surplus for the miners in Π'. In addition, the attack can be made compatible with other available cryptocurrencies that can be merged mined with Π. Therefore, additional revenue channels from existing merge mined cryptocurrencies are not affected by the pitchfork and can even help to subsidize the attack.

As a further parameterization for the attack, it is also possible to execute it in stages. To test whether there is enough support for chain Π', it is possible to first start with a relatively low risk to the attackers by not requiring them to mine empty blocks and instead only demand the creation of smaller blocks that can still include high fee transactions. From there, the attackers can reduce the number of permissible transactions step by step. At a final stage, all coins earned through mining empty blocks in Π can also be used to fund additional attacks, such as triggering additional spam transactions in Π as soon as the cooldown period of 100 blocks has passed. For instance, splitting the coinbase rewards into many individual outputs of a high enough value with different lock times and rendering the output scripts as *anyone can spend* can lead to a large influx of additional transactions as users (and miners) compete to scoop up these free currency units. This is easy to verify as an additional rule in Π'. However, more complex attack scenarios such as those outlined in [1, 14, 11] may also be included as additional consensus rules.

4 Countermeasures

In this section, we outline some countermeasures that can be taken by players in $\mathcal{A}$, as well as their effectiveness.

4.1 Exclude Empty Blocks in Π

The miners in $\mathcal{A}$ can decide to fork off empty blocks and just build on top of blocks containing transactions. This requires the coordinated action of all miners in $\mathcal{A}$. If $p_{\mathcal{A}} > p_{\mathcal{B}} + p_{\mathcal{I}}$, this approach will work in general. A possible counter-reaction by the attackers in Π' would be to introduce dummy transactions to themselves in their blocks in Π. Therefore, it has to be ensured that those transactions are indeed dummies. For example: All used output addresses of every transaction belong to the same entity, but this must not be possible to correlate given just the block b_n in Π. One way to achieve this is to require that all output addresses in a block have been derived from the miner's public key of the respective block, like in a Hierarchically Deterministic (HD) Wallet[7] construction. The *master public key property* of such a construction allows that future ECDSA public keys can be derived from current ones. This is done by adding a multiplication of the base point with a scalar value to the current public key. The corresponding secret key is derived in the same manner but can only be computed by its owner. If it is not possible to perform a transaction to an address for which the miner does not have the corresponding private key, the utility of a block only containing transactions of the respective miner is very limited for users of Π. To check this condition on an arbitrary block b_n, the public key of the miner, as well as the scalar value for the multiplication, is required. These values can be added to the coinbase transaction of the corresponding block b'_n in Π'.

[7] Cf. BIP32 https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki

If such dummy transactions are used, the miners of $\mathcal{A}$ would be required to monitor the chain of Π' to deduce which block in Π has been merged mined with Π' and includes only dummy transactions. If the miners of $\mathcal{A}$ find such a block, they can still cause a fork in Π to ignore it. Besides being more complex, this also poses a potential risk for all transactions in Π. Since the block b_n could be released before b'_n, there is no way to tell whether or not b_n was indeed merged mined. And hence includes a hash to b'_n before b'_n has been published in Π'. With this knowledge, miners in $\mathcal{B}$ can intentionally create forks in Π by holding back new blocks in Π' for a while. By slightly relaxing the rules for dummy transactions and allowing, for example, one transaction output address that is not required to be derivable by the HD construction, double-spends can be executed more easily in Π. In this particular case, miners of merged mined blocks can include a regular transaction that they want to double-spend in their block, being assured that this block will get excluded in retrospect by all miners $p_{\mathcal{A}}$ in Π if b'_n is released in Π'. Therefore, more fine-grained exclusion rules on the transaction level would be necessary.

These examples illustrate that it is non-trivial to change the consensus rules in Π such that the effects of a pitchfork attack are mitigated. Every change of the defenders in $\mathcal{A}$ leads to an arms race with the attackers in $\mathcal{B}$. Moreover, excluding all merge mined blocks in Π requires active monitoring of Π' to detect them. Therefore, at least the miners in $\mathcal{A}$ have to change their individual consensus rules for Π – which they wanted to avoid in the first place.

4.2 Launch a Counter-attack on Π'

Miners in $\mathcal{A}$ can use their mining power to counter-attack the attacking chain Π'. However, this has several limitations: Since every block in Π' requires an empty parent block in Π as part of its PoW, miners cannot create empty merge mined blocks in Π' while at the same time creating full blocks in Π. To stall the pitchfork Π', at least a fraction of $p_{\mathcal{A}}$ ($p'_{\mathcal{A}} \leq p_{\mathcal{A}}$) has to mine empty blocks in Π to also create empty merge mined blocks for Π'. Though, thereby the counter-attackers could actually help the pitchfork attack.

To clearly overtake the pitchfork chain Π', the counter-attacking miners need to have more than 50% of the hashrate in Π'. If not, the lost throughput caused by empty blocks in Π' might be compensated by the increased block size. This introduces the first constraint for the counter-attackers, i.e., that the hashrate $p'_{\mathcal{A}}$ they dedicate to the counter-attack must be larger than the attack hashrate $p'_{\mathcal{A}} > p_{\mathcal{B}}$.

However, the counter-attackers must also take care not to push the total hashrate $p'_{\mathcal{A}} + p_{\mathcal{B}}$, which is dedicated towards attacking Π over 50%. Otherwise, more destructive attack rules than mining empty blocks may be rendered effective. For example, requiring non-empty blocks to be ignored or anyone-can-spend transactions. If the defenders retain the majority in Π, and if they are able to reliably identify all merge-mined attack blocks, they can exclude them in Π. Thus, the second constraint requires that for a counter-attack, the bound $p_{\mathcal{B}} + p'_{\mathcal{A}} < 0.5$ for the share of blocks in the heaviest chain of Π holds.

Depending on the exact implementation of merged mining in Π', the counter-attackers have some options to avoid that their empty blocks for Π, which they are required to provide as PoW for Π', cause further harm to Π. Therefore, we differentiate between *counter-attacks without direct negative consequences* on Π and *counter-attacks with direct negative consequences* on Π.

An example for a counter-attack without direct negative consequences on Π would be to only submit PoW solutions to Π' that fulfill the difficulty target for Π' but not for Π. This counter-attack approach has the marked disadvantage that any block meeting the difficulty target of Π also cannot be submitted as solutions in Π', effectively reducing the counter-attackers' hashrate $p'_{\mathcal{A}}$ in Π' by a factor dependent on the particular difference in difficulty between Π' and Π. A better counter-attack without direct negative consequences on Π can be achieved if the defenders intentionally construct blocks for the parent chain Π that are unlikely to end up in the main chain yet are still accepted as a valid proof-of-work in Π'. For instance, stale branches in Π could be created and extended. However, this is only effective if the freshness requirements for parent blocks in Π' are not too tight. In both cases, since $p'_{\mathcal{A}}$ is no longer contributing toward the effective hashrate of Π, its remaining honest miners, $p_{\mathcal{I}} + p_{\mathcal{A}} - p'_{\mathcal{A}}$ must still retain a hashrate that exceeds that of the adversary to ensure that honest blocks constitute a majority of the heaviest chain. Therefore, the original attacker gains an advantage from merged mining since he can use his full hashrate in both chains at the same time. Moreover, the counter-attacking fraction of the miners would forgo their rewards in Π for the duration of the counter-attack.

We now compare the two cases of counter-attacks with and without direct negative consequences on the parent chain Π and calculate the maximum tolerable hashrate of $p_{\mathcal{B}}$, such that the counter-attack succeeds in dominating at least one of the two systems, i.e., $\mathcal{A}$ has more than half of the hashrate on at least one system. For this analysis, we make the simplifying assumption that the total hashrate of indifferent miners $p_{\mathcal{I}}$ is zero. Hence the total hashrate of $\mathcal{A}$ is $p_{\mathcal{A}} = 1 - p_{\mathcal{B}}$. This hashrate can be split between the two chains Π and Π' arbitrarily to launch the counter-attack.

Counter-attack without Direct Negative Consequences on Π In this case, we assume that direct negative consequences from the pitchfork attack on the parent chain Π can be avoided while creating merge mined child chain blocks for Π. For example, merged mined empty blocks of the parent chain are accepted as a valid PoW for the child chain, even if they do not have a valid predecessor in the parent chain. Indirect negative consequences of the counter-attack, like, for example, an overall hashrate reduction that works on the longest heaviest chain in Π, is ignored for this analysis.

Figure 2 shows the hashrates achievable by $\mathcal{A}$ on the respective chains Π and Π' for defending and counter-attacking. The figure is parameterized by different values for the hashrate of the pitchfork attacker ($p_{\mathcal{B}}$). It can be observed that in this case, an attacker with $p_{\mathcal{B}} > \frac{1}{3}$ total hashrate cannot be countered on both

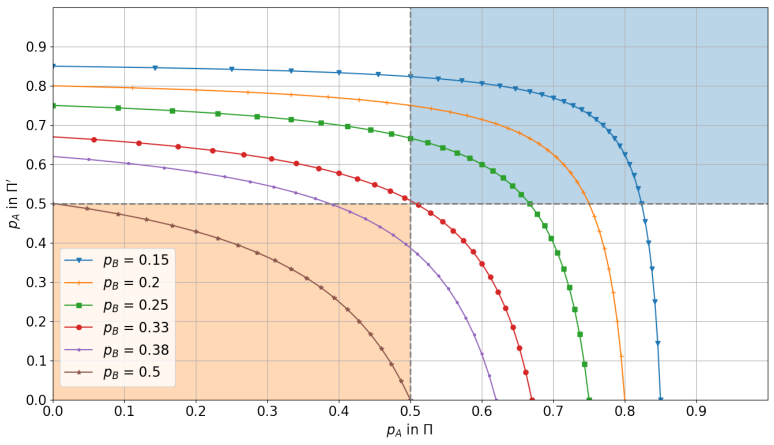

Fig. 2. Hashrates of the defender/counter-attacker in the respective chains Π and Π' for different values of pitchfork attacker hashrate $p_{\mathcal{B}}$. The colored orange square marks the area in which the counter-attackers $\mathcal{A}$ loses in both chains, whereas the colored blue square marks the area in which the counter-attackers $\mathcal{A}$ can retain the majority in both systems.

chains simultaneously without losing the majority $p_{\mathcal{A}} < 0.5$ on one of the two chains.

Theorem 1. Assuming the invested counter-attack hashrate does *not* directly strengthen the attack on the merge mined parent chain Π. Then, to successfully perform a counter-attack in which $\mathcal{A}$ dominates a merge minded pitchfork child chain Π', the attacker hashrate $p_{\mathcal{B}}$ must be less than $\frac{1}{3}$, s.t. $\mathcal{A}$ can dominate both chains (i.e., have more than half of the hashrate in both chains).

Proof. Since the pitchfork attack utilizes merged mining, the hashrate of the attacker $p_{\mathcal{B}}$ is available in both systems, Π and Π'. To overtake and dominate the merge mined child chain Π', the most a defender with hashrate $p_{\mathcal{A}}$ can invest in terms of hashrate is $\frac{1}{2}$, as otherwise, we would lose in protocol Π. To keep control of the parent, the following inequality has to hold: $\frac{p_{\mathcal{A}}}{2} > p_{\mathcal{B}}$. Since we are in a two-player model, the hashrates are defined as: $p_{\mathcal{A}} = 1 - p_{\mathcal{B}}$. Replacing $p_{\mathcal{A}}$ in our inequality then gives us the maximum hashrate of the attacker $\frac{1-p_{\mathcal{B}}}{2} > p_{\mathcal{B}}$, which simplifies to $p_{\mathcal{B}} < \frac{1}{3}$, and provides the upper bound for the hashrate of the attacker s.t., the defender $\mathcal{A}$ can win in both chains. $\square$

Figure 3 shows a visualization of this scenario in which merged mining the parent is possible without causing the negative side effects intended by the pitchfork attack on the parent chain. In this example, $p_{\mathcal{B}} = \frac{1}{3}$ and thus no party can clearly win either chain.

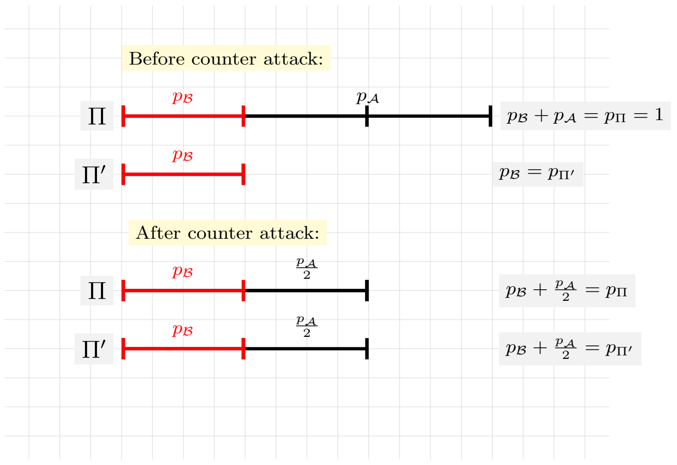
Before counter attack:
$p_\mathcal{B}$
$p_\mathcal{A}$
Π
$p_\mathcal{B} + p_\mathcal{A} = p_\Pi = 1$
$p_\mathcal{B}$
Π'
$p_\mathcal{B} = p_{\Pi'}$
After counter attack:
$p_\mathcal{B}$
$\frac{p_\mathcal{A}}{2}$
Π
$p_\mathcal{B} + \frac{p_\mathcal{A}}{2} = p_\Pi$
$p_\mathcal{B}$
$\frac{p_\mathcal{A}}{2}$
Π'
$p_\mathcal{B} + \frac{p_\mathcal{A}}{2} = p_{\Pi'}$

Fig. 3. Visual comparison of hashrates in the two systems Π and Π', before and after counter attacking the merge minded pitchfork protocol Π', when merged mining of the parent is possible without negative effects on the parent chain. In this example, the hashrate of the attacker is $p_\mathcal{B} = \frac{1}{3}$, which results in a situation where no player can clearly win on any of the two chains in case of a counter-attack.

This result for permissionless PoW cryptocurrencies has an interesting relation to a paper by Lindell et al. [12], in which the authors prove that authenticated Byzantine Agreement protocols only remain secure under parallel or concurrent composition (even for just two executions), when more than 2/3 of the participating parties are honest.

Counter-attack with Direct Negative Consequences on Π In this case, we assume that direct negative consequences of the attack on the parent chain Π cannot be avoided while creating merge mined child chain blocks for Π.

Theorem 2. Assuming the invested counter-attack hashrate directly strengthens the attack on the merge mined parent chain Π. Then, to successfully perform a counter-attack in which $\mathcal{A}$ dominates a merge mined pitchfork child chain Π', with no ability to avoid damage on the target chain, the attacker hashrate $p_\mathcal{B}$ must be less than $\frac{1}{4}$, s.t. $\mathcal{A}$ can dominate both chains (i.e., have more than half of the hashrate on both chains).

Proof. Since the pitchfork attack utilizes merged mining, the hashrate of the attacker is available in both systems. The difference now is that the defenders can not switch parts of their hashrate to the pitchfork chain running Π' without harming the target chain running Π. We denote the hashrate that switches and gets “malicious” in Π with $p'_\mathcal{A}$. To determine the maximum $p_\mathcal{B}$, the following conditions have to hold:

$$p_{\mathcal{B}} + p'_{\mathcal{A}} < p_{\mathcal{A}} \quad \text{(Ensures dominance of } \mathcal{A} \text{ on targeted parent chain)} \tag{1}$$

$$p_{\mathcal{B}} < p'_{\mathcal{A}} \quad \text{(Ensures dominance of } \mathcal{A} \text{ on child chain)} \tag{2}$$

$$p_{\mathcal{B}} + p'_{\mathcal{A}} + p_{\mathcal{A}} = 1 \quad \text{(Overall hashrate is bound by 1)} \tag{3}$$

This gives us $p_{\mathcal{A}} > \frac{1}{2}$ and $p_{\mathcal{B}} < \frac{1-p_{\mathcal{A}}}{2}$. Solving this for the point where the hashrate/power on both chains is equal (i.e., $p_{\mathcal{B}} = p'_{\mathcal{A}}$ and $p_{\mathcal{B}} + p'_{\mathcal{A}} = p_{\mathcal{A}}$) gives us the upper bound $p_{\mathcal{B}} < \frac{1}{4}$ for the attacker hashrate s.t., the defender $\mathcal{A}$ can win on both chains.

$$p_{\mathcal{B}} + p'_{\mathcal{A}} - p_{\mathcal{A}} = 0 \tag{4}$$

$$p_{\mathcal{B}} + p'_{\mathcal{A}} + p_{\mathcal{A}} = 1 \tag{5}$$

$$p_{\mathcal{B}} = p'_{\mathcal{A}} \tag{6}$$

$$2p_{\mathcal{B}} + 2p_{\mathcal{B}} = 1 \tag{7}$$

$$p_{\mathcal{B}} = \frac{1}{4} \tag{8}$$

□

Figure 4 shows a visualization of this scenario in which merged mining the parent is not possible without negative side effects on the parent chain. In this example, $p_{\mathcal{B}} = \frac{1}{4}$ and thus no party can clearly win either chain in case of a counter-attack.

4.3 Do nothing

Given the undesired side effects of the other countermeasures discussed so far, one option for the defenders $\mathcal{A}$ would be to do nothing. Then, the users of Π have to live with a reduced throughput in the size of $p_{\mathcal{B}}$. This would work and dry out the funds of the attackers $\mathcal{B}$, if the reduced gains from missed transaction fees in Π cannot be compensated by the new gains of mining an additional cryptocurrency, Π'. If the gains in Π' overcompensate the losses in Π, then the surplus in funds can be reinvested into new mining hardware to increase the overall share in the mining ecosystem until $p_{\mathcal{B}}$ accounts for the majority of mining power. Then, they would be able to overtake Π directly.

In other words, the success of ignoring the pitchfork attack depends on the balance between lost income and newly gained income. If the loss in income from pitchfork mining Π can be avoided, for example, if the consensus rules of Π are designed in a way such that the transaction fees in Π do not count towards the income of the miner but instead are burned, then mining empty blocks would not have any disadvantage. Another option would be to require more complex attacks on Π as a PoW for Π'. Such attacks might even be profitable in the target cryptocurrency Π.

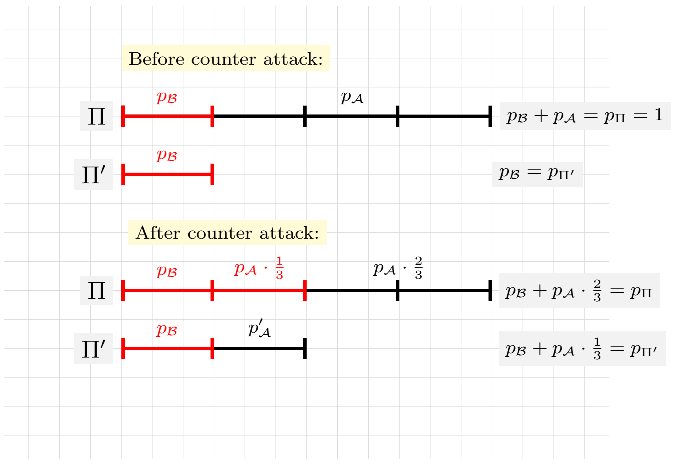
Before counter attack:
Π
$p_{\mathcal{B}}$
$p_{\mathcal{A}}$
$p_{\mathcal{B}} + p_{\mathcal{A}} = p_{\Pi} = 1$
Π'
$p_{\mathcal{B}}$
$p_{\mathcal{B}} = p_{\Pi'}$
After counter attack:
Π
$p_{\mathcal{B}}$
$p_{\mathcal{A}} \cdot \frac{1}{3}$
$p_{\mathcal{A}} \cdot \frac{2}{3}$
$p_{\mathcal{B}} + p_{\mathcal{A}} \cdot \frac{2}{3} = p_{\Pi}$
Π'
$p_{\mathcal{B}}$
$p'_{\mathcal{A}}$
$p_{\mathcal{B}} + p_{\mathcal{A}} \cdot \frac{1}{3} = p_{\Pi'}$

Fig. 4. Visual comparison of hashrates in the two systems Π and Π', before and after counter attacking the merge minded pitchfork protocol Π', when merged mining of the parent is not possibe without negative side effects on the parent chain. In this example, the hashrate of the attacker is $p_{\mathcal{B}} = \frac{1}{4}$, which results in a situation where no player can clearly win on any of the two chains in case of a counter-attack.

5 Related Attacks, Potential Enhancement, and Further Analysis of Incentive Manipulation

In [8], it is argued that merged mining could also be used as an attack vector against the parent chain. However, no concrete examples are given. In this paper, we outline that merged mining can be used as an attack method against a PoW cryptocurrency in the context of a hostile protocol hard fork. The pitchfork attack is an example of such a technique, which poses interesting questions regarding the interplay of different cryptocurrencies, as well as their incentive structures. The difficulties regarding counter-attacking a Pitchfork attack on a permissionless PoW cryptocurrency have an interesting relation to a paper by Lindell et al. [12], in which the authors prove that authenticated Byzantine Agreement protocols only remain secure under parallel or concurrent composition (even for just two executions) when more than $2/3$ of the participating parties are honest.

A natural question now is if there are other attack possibilities on permissionless PoW cryptocurrencies leveraging such incentive and concurrency issues. Different bribing methods that can be used in hostile blockchain takeovers are described in [2], placing the focus on attacks where the attacker has an extrinsic motivation to disrupt the consensus process, i.e., Goldfinger attacks [10]. The example given in this paper is a concrete instance of such a situation. The pitchfork attack can also be viewed as a subsidized Goldfinger attack. Therefore, some of the described methods for Goldfinger attacks might also be used in conjunction with our proposed attack. This also holds true for the large body of work on bribing [1, 13] and incentive attacks that distract the hashrate of participants [14, 15].

The general idea of such an *offensive consensus attack*, like the pitchfork, is that the participants of the offensive system are required to provably attack a different system as part of the consensus rules. We show that such attacks are theoretically possible and can lead to an arms race in which defenders are forced to adapt their consensus rules. Still, the consequences, as well as the economic and game-theoretic incentives of such attacks, have yet to be analyzed in greater detail to better understand if they are practicable and, if so, how to protect against them.

References

1. J. Bonneau. Why buy when you can rent? Bribery attacks on Bitcoin consensus. In *BITCOIN '16: Proceedings of the 3rd Workshop on Bitcoin and Blockchain Research*, Feb. 2016.
2. J. Bonneau. Hostile blockchain takeovers (short paper). In *5th Workshop on Bitcoin and Blockchain Research, Financial Cryptography and Data Security 18 (FC). Springer*, 2018.
3. 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 *IEEE Symposium on Security and Privacy*, 2015.
4. K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, and E. Gün. On Scaling Decentralized Blockchains. In *3rd Workshop on Bitcoin and Blockchain Research, Financial Cryptography 16*, 2016.
5. I. Eyal, A. E. Gencer, E. G. Sirer, and R. van Renesse. Bitcoin-NG: A Scalable Blockchain Protocol. In *13th USENIX Security Symposium on Networked Systems Design and Implementation (NSDI'16)*. USENIX Association, Mar. 2016.
6. A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdo rf, and S. Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC*, pages 3–16. ACM, 2016.
7. I. Giechaskiel, C. Cremers, and K. B. Rasmussen. On Bitcoin Security in the Presence of Broken Cryptographic Primitives. In *European Symposium on Research in Computer Security (ESORICS)*, Sept. 2016.
8. A. Judmayer, A. Zamyatin, N. Stifter, A. G. Voyiatzis, and E. Weippl. Merged Mining: Curse or Cure? In *CBT'17: Proceedings of the International Workshop on Cryptocurrencies and Blockchain Technology*, Sept. 2017.
9. A. Kiayias, A. Miller, and D. Zindros. *Non-interactive proofs of proof-of-work*. 2017. Published: Cryptology ePrint Archive, Report 2017/963.
10. J. A. Kroll, I. C. Davey, and E. W. Felten. The economics of Bitcoin mining, or Bitcoin in the presence of adversaries. In *Proceedings of WEIS*, volume 2013, page 11, 2013.
11. K. Liao and J. Katz. Incentivizing blockchain forks via whale transactions. In *International Conference on Financial Cryptography and Data Security*, pages 264–279. Springer, 2017.
12. Y. Lindell, A. Lysyanskaya, and T. Rabin. On the Composition of Authenticated Byzantine Agreement. Technical Report 181, 2004.

13. P. McCorry, A. Hicks, and S. Meiklejohn. Smart Contracts for Bribing Miners. In *5th Workshop on Bitcoin and Blockchain Research, Financial Cryptography and Data Security 18 (FC). Springer*, 2018.

14. J. Teutsch, S. Jain, and P. Saxena. When cryptocurrencies mine their own business. In *Financial Cryptography and Data Security (FC 2016)*, Feb. 2016.

15. Y. Velner, J. Teutsch, and L. Luu. Smart contracts make Bitcoin mining pools vulnerable. In *International Conference on Financial Cryptography and Data Security*, pages 298–316. Springer, 2017.

16. A. Zamyatin, N. Stifter, A. Judmayer, P. Schindler, E. Weippl, and W. J. Knottebelt. A Wild Velvet Fork Appears! Inclusive Blockchain Protocol Changes in Practice. In *5th Workshop on Bitcoin and Blockchain Research, Financial Cryptography and Data Security 18 (FC). Springer*, 2018. (Short Paper).

Acknowledgements

We thank Georg Merzdovnik and Alexei Zamyatin as well as the participants of Dagstuhl Seminar 18152 "Blockchains, Smart Contracts and Future Applications" for valuable discussions and insights. This research was funded by Bridge Early Stage 846573 A2Bit, Bridge 1 858561 SESC, Bridge 1 864738 PR4DLT (all FFG) , CDL-SQI at the Institute of Information Systems Engineering TU Wien, and the competence center SBA-K1 funded by COMET.