SoK: Algorithmic Incentive Manipulation Attacks on Permissionless PoW Cryptocurrencies

Aljosha Judmayer[1,2], Nicholas Stifter[1,2], Alexei Zamyatin[3], Itay Tsabary[4], Ittay Eyal[4], Peter Gaži[5], Sarah Meiklejohn[6], and Edgar Weippl[2]

[1] SBA Research {ajudmayer,nstifter}@sba-research.org
[2] Uni Wien edgar.weippl@univie.ac.at
[3] Imperial College London a.zamyatin@imperial.ac.uk
[4] Technion and IC3 Ittay@technion.ac.il,itaytsabary@gmail.com
[5] IOHK peter.gazi@iohk.io
[6] University College London s.meiklejohn@ucl.ac.uk

eprint from 2021-03-01

Abstract. A long standing question in the context of cryptocurrencies based on Nakamoto consensus is whether such constructions are incentive compatible, i.e., the intended properties of the system emerge from the appropriate utility model for participants. Bribing and other related attacks, such as front-running or Goldfinger attacks, aim to directly influence the incentives of actors within (or outside) of the targeted cryptocurrency system. The theoretical feasibility of bribing attacks on cryptocurrencies was first highlighted in 2016 by Bonneau, with various different techniques and approaches having since been proposed. Some of these attacks are designed to gain in-band profits, while others intend to break the mechanism design and render the cryptocurrency worthless. In this paper, we systematically expose the large but scattered body of research in this area which has accumulated over the years. We summarize these bribing attacks and similar techniques that leverage on programmatic execution and verification under the term *algorithmic incentive manipulation* (AIM) attacks, and show that the problem space is not yet fully explored. Based on our analysis we present several research gaps and opportunities that warrant further investigation. In particular, we highlight *no- and near-fork* attacks as a powerful, yet largely underestimated, AIM category that raises serious security concerns not only for smart contract platforms.

Keywords: Algorithmic incentive manipulation · Cryptocurrencies · Bribing · Goldfinger · Front-running.

1 Introduction

Bitcoin, and most of its cryptocurrency descendants, is based on what is termed Nakamoto consensus [47,16,55]. In a nutshell, Nakamoto consensus enables *anyone* to initiate valid state transitions to a replicated state machine if they solve a cryptographic puzzle of sufficient hardness that depends on the prior state. This is usually implemented by appending a block of ordered transactions and an appropriate proof-of-work [21] to a directed rooted and cryptographically linked tree of other blocks. The path to the leaf with the highest depth (resp. difficulty) is called the *longest (heaviest) chain* and thus the current state of the system.

A crucial part of this so-called *permissionless* [60] consensus concept is the utilization of incentives in the protocol design to provide a motivation for miners to participate. A long-standing question in this regard is whether or not this construction is incentive compatible, i.e., that the intended properties of the system emerge from the appropriate utility model for miners [14,15]. As Nakamoto did not provide a formal description of the protocol in [47], several attempts towards formalization [7] have been made to prove certain security properties of the protocol. Thereby, most approaches, such as [26,48,13,28], do assume a sufficient *honest* majority of miners without considering incentives, or like [12] explicitly do not consider *bribing attacks* to manipulate incentives of participants.

Bribing attacks target incentive compatibility and assume that at least some miners accept bribes to maximize their profit. Hereby, bribing not necessarily refers to illegal activity, but merely that a payment is made in exchange for a certain action [15]. If the attacker, together with all bribable miners, can gain a sizable portion of the computational power even for a short period of time, attacks are likely to succeed. To the best of our knowledge, the first discussions of *bribery* attacks on Bitcoin date back to a bitcointalk forum post from 2012 by a user called *cunicula* [18]. Since then attacks on incentives in cryptocurrencies have been sporadically discussed in the cryptocurrency community [40,11], with the first peer reviewed paper on the subject presented in 2016 by Bonneau [14]. Over the years several different techniques and approaches of bribing attacks have been proposed [18,40,11,14,56,42,43,59,44,31,32,61,37,58]. These proposals vary regarding their system models, technical methods and evaluation criteria, which makes comparing them a challenging task. What all this approaches have in common is that they are targeted to manipulate the incentives of actors in the cryptocurrency ecosystem.

All these attacks, as well as meta arguments [25,17,36,15,14] and recent research [19,35] have fueled the debate around incentives in Nakamoto consensus. A key observation hereby noted in Bonneau [14] and Budish [17] is, that the security guarantees of Nakamoto consensus against bribing attacks to facilitate double-spending, are *linear* in the number of blocks and the expenditure on mining power to produce them (in terms of financial resources required). In contrast, the achievable security guarantees of many other investments in IT security, like for example traditional usage of cryptography, are designed to "*yield convex returns*" [17]. Large scale temporary majority attacks, in which an attacker overtakes a cryptocurrency for a short period of time, have gained further practical importance as they have been observed more frequently in recent history [9,6,5,8].

[7] For a summary see [55].

Lately, also other attempts to manipulate incentives targeted to influence the *order* of transactions within a not yet mined block [22,19,32] (front-running), or to *exclude* transactions [32,61] have received increased attention, as some less sophisticated variants of them have been observed in the wild already [19,4]. These attacks show that the security properties of Nakamoto consensus under real-world incentives are still not fully understood, not only by cryptocurrency users and smart contract developers, but also in the research community.

This paper aims to systematize the landscape of research on attacks that target the incentives of actors within – and through the use of – cryptocurrencies. To systematically expose the large body of research on bribing-, front-running-Goldfinger- and other related attacks, we jointly consider them under the general term *algorithmic incentive manipulation attacks* (AIM). Thereby, we want to distinguish programmatic ways to setup and execute incentive attacks on cryptocurrencies using cryptocurrencies, from "classical" bribing attacks, like for example using a suitcase full of cash to bribe miners, as the latter does not require technical means, but at the same time lacks technical enforcement [14]. The classification of AIM attacks in this paper forms a necessary prerequisite and basis for the comparison and discussion of work in this field – being it attacks or meta arguments. In summary, our contributions are:

1. A definition of algorithmic incentive manipulation (AIM) providing a unified view of different approaches targeting the incentives of actors.
2. A generalized attack model for AIM.
3. A classification framework for AIM that is applicable to describe a broad class of attacks.
4. A classification of existing AIM approaches and discussion of main observations and gaps.

1.1 Structure of this work

We start by giving a definition of AIM in Section 2. Next, in Section 3, we provide a generalized attack model that can be readily applied to most presented attacks by adjusting the provided parameters. We then present the main classification criteria for AIM in Section 4. The classification and analysis of existing attacks is provided in Section 5, by comparing them property by property. In Section 6 we highlight the challenges when comparing costs and profits of AIM attacks. We conclude by discussing the relation of AIM to other ways of gaining capacity in Nakamoto consensus and present directions for future work in Section 7.

2 Algorithmic Incentive Manipulation

To meaningfully partake in a Nakamoto consensus protocol, a certain capacity of a scarce resource is required. In case of Proof-of-Work (PoW) these resources are mining hardware and electricity to solve cryptographic puzzles. In [15] the different ways to gain capacity in Nakamoto consensus are grouped into four different strategies: *rent*, *bribe*, *build* and *buy out*. It is well known, that an actor who builds a new datacenter running specialized mining hardware, or rents GPU clusters, or buys existing mining hardware from current miners can increase his influence on the targeted cryptocurrency and thereby (depending on his resources) potentially launch attacks [15,17]. This permissionlessness [60] which allows such kinds of attacks is a desired property of Nakamoto consensus based on PoW[8].

In this paper we want to focus on methods of *algorithmic incentive manipulation* (AIM) to gain capacity in permissionless PoW based cryptocurrencies, as all existing attacks which fall into this category – and are classified in this paper – explicitly target PoW systems. Algorithmic or "virtual" methods of gaining capacity rely on the usage of game theory and cryptocurrencies to perform payments which are cryptographically secured and dependent on certain conditions. This ability of cryptocurrencies to tie payments to the fulfillment of certain conditions, like for example the existence of prior transactions, or the successful execution of smart contract invocations, are a way to promise *credible but conditioned* payments.

Utilizing such techniques, AIM methods do not involve the physical transfer of resources, like buying, or building and maintaining hardware. Instead, these methods assume that at least some fraction of actors within, or outside of the system behaves rationally in the sense that they want to maximize their short-term profits[9]. Some approaches for AIM have been referred to as *bribing*, but AIM goes beyond of what is currently viewed as a bribing attack in the literature, as they should incorporate Goldfinger [38] and front-running [22,19] attacks as well. Therefore our broader definition is as follows:

Definition 1. *Algorithmic Incentive Manipulation (AIM) utilizes either credible threats, or conditioned rewards denominated in cryptocurrency units, to incentivize certain actions, within a targeted cryptocurrency system, to be taken by capable actors.*

Hereby, the definition of *capable actor* depends on the requirements of the concrete attack, as well as the targeted system. For most attacks, the timely creation and submission of valid PoW solutions – complying to the required difficulty – is necessary to qualify as a capable actor. If the target would be a Proof-of-Stake (PoS) cryptocurrency for example, a capable actor would be required to control voting stake.

[8] In comparison, in proof-of-stake (PoS) cryptocurrencies it would not be possible to rent or build *new* capacity, as all stake eligible for voting has to exist in the system already [15].

[9] For a discussion on rationality in this context see, Section 7.

3 Generalized Attack Model for AIM

For all analyzed AIM attacks we describe the following generalized attack model, which can readily be applied in most cases.[10] If an analyzed attack deviates from this model, it is highlighted in detail when the attack is described.

As most bribing and related attacks in this area are designed to target Bitcoin, Ethereum, or other derived cryptocurrencies, we also focus in our model on AIM in *permissionless* [60] proof-of-work (PoW) cryptocurrencies. That is, we assume protocols adhering to the design principles of Bitcoin, or its backbone protocol [47,27,48], which is sometimes referred to as Nakamoto consensus [20,55]. Within the attacked cryptocurrency we differentiate between *miners*, who participate in the consensus protocol and attempt to solve PoW-puzzles, and *clients*, who do not engage in such activities. As in previous work on bribing attacks [42,56,44,14], the set of miners is assumed to be fixed, as well as their respective computational power within the network is assumed to remain constant.

To abstract from currency details, we use the term *value* as a universal denomination for the purchasing power of a certain amount of cryptocurrency units or any other out-of-band funds such as fiat currency. Miners and clients may own cryptocurrency units and are able to transfer them (i.e., their value) by creating and broadcasting valid transactions within the network. Moreover, in most prior work [57,42,44] the simplifying assumption is made that exchange rates remain constant over the duration of the attack.

Actors. Our generalized attack model splits participating miners into three groups and their roles remain static for the attack duration. Categories follow the *BAR (Byzantine, Altruistic, Rational)* [10,41] rational behavior model. Additionally, we define the *victim(s)* as another group or individual without hashrate.

- **Byzantine miners or attacker(s)**: The attacker B wants to execute an AIM attack on a *target cryptocurrency*. B is in control of bribing funds $f_B > 0$ that can be in-band or out-of-band, depending on the attack scenario. He has some or no hashrate $p_{\mathcal{B}} \geq 0$ in the target cryptocurrency. The attacker may deviate arbitrarily from the protocol rules.
- **Altruistic or honest miner(s)**: Altruistic miners A are honest and always follow the protocol rules, hence they will not accept bribes to mine on a different chain-state or deviate from the rules, even if it would offer larger profit. Miners A control some or no hashrate $p_{\mathcal{A}} \geq 0$ in the target cryptocurrency.
- **Rational or bribable miner(s)**: Rational miners R control hashrate $p_{\mathcal{R}} > 0$ in the target cryptocurrency. They aim to maximize their short term profits in terms of *value*. We consider such miners "bribable", i.e., they follow

[10] Only the Proof-of-Stale blocks [43,59] attack, as well as Fomo 3D [4] are fundamentally different: The former is targeted to attack mining pools, while the latter is designed as an exit scam, but can also lead to scenarios resembling an attack.

strategies that deviate from the protocol rules as long as they are expected to yield higher profits than being honest. For our analyses we assume that rational miners do not concurrently engage in other rational strategies such as selfish mining [24].

- **Victim(s)**: The set of victims or a single victim, which loses value if the bribing attack is to be successful. The victims control zero hashrate, and therefore can be viewed as a client.

It holds that $p_{\mathcal{B}} + p_{\mathcal{A}} + p_{\mathcal{R}} = 1$. The assumption that the victim of a AIM attack has no hashrate is plausible, as the majority of transaction in Bitcoin or Ethereum are made by clients which do not have any hashrate in the system they are using. Nonetheless, this assumption is often left implicit (e.g. [42]).

Some bribing attacks (e.g. [56]) implicitly model victims (in this case the betrayed collaborators of the double-spending attack) as honest, i.e., as strictly following the protocol. We emphasise that this is not necessarily the case, especially if economically rational counter-attacks by the victim should be considered. This distinction between rational and honest victims is more important if V is in possession of some hashrate, but even in a setting where V has no hashrate, he can use his funds (f_V) for counter attacks.

Whenever we refer to an attack as *trustless*, we imply that no trusted third party is needed between the briber and the bribee to ensure correct payments are performed for the desired actions. It is clearly desirable from the attacker's perspective to design AIM attacks in a way that the attacker(s) as well as the collaborating miners have no incentive to betray each other if they are economically rational.

Communication and Timing. As previous AIM attacks, we assume that all miners in the target cryptocurrency have *perfect knowledge* about the attack once it has started, if not stated differently. Miners with imperfect information can be modelled by adding their respective hashrate to the hashrate of altruistic miners ($p_{\mathcal{A}}$). All participants communicate through message passing over a peer-to-peer gossip network, which we assume implements a reliable broadcast functionality. This does not mean, that every transmitted transaction will make it into the next block, as the block size is bounded by the underlying blockchain protocol. Analogous to [27], we model the adversary as "rushing", meaning that he gets to see all other players' messages before he decides his on strategy.

If more than one cryptocurrency is involved in the considered scenario, for example when out-of-band payments should be performed in another cryptocurrency, an additional *funding cryptocurrency* is assumed. While the attack is performed on a *target cryptocurrency*, the funding cryptocurrency is used to orchestrate and fund it. In such a case, we also assume that the difficulty and thus the mean block interval of the funding chain is fixed for the duration of the attack. Further, no additional attacks are concurrently being launched against either of the cryptocurrencies.

4 Classification Framework for AIM

We first introduce a general classification along four main dimensions: the *state of the targeted transaction(s)*; the *intended impact* on these transactions; the *required interference with consensus*, i.e., the depth of blockchain reorganizations caused by forks for the attack to be successful; and finally the *used payment methods*. Besides these main distinguishing properties, there are also other characteristics which are introduced when they become relevant during the classification of existing AIM attacks in Section 5. To get a feel for our classification framework and the herein introduced dimensions, see Section A for an example usage.

4.1 State of Targeted Transactions

A core goal for permissionless PoW cryptocurrencies is to achieve an (eventually) consistent and totally ordered log of transactions that define the global state of the shared ledger. Therefore, our classification uses a transaction-centric viewpoint to systematize different attacks and their relation to the underlying consensus. We differentiate between three *states* a transaction can be in from the perspective of a participant (miner or client):

- **unconfirmed**[11], the transaction has been broadcasted in the respective P2P network;
- **confirmed**, the transaction has been confirmed by at least one block, i.e., has been included in a block;
- **settled**, the transaction has been confirmed by at least k blocks, where k is defined by the recipient of the transaction. We denote $k_{participant}$ to refer to the confirmation policy of a participant if it is not clear from the context e.g., k_V denotes the confirmation policy of the victim.

4.2 Intended Impact/Influence on Transactions

We further separate between the following four main types of how AIM can have an influence on transactions and their ordering:

- **transaction revision**, change a previously proposed, possibly confirmed or settled transaction;
- **transaction exclusion/censorship**, exclude a specific transaction, or set of transactions, from the log of transactions for a bounded amount of time i.e., the transaction remains unconfirmed.
- **transaction ordering**, change either the proposed, confirmed or already settled upon order of transactions;
- **transaction triggering**, incentivize the creation of one or multiple transactions by a specific actor or group of actors, e.g., trigger spending transactions for *anyone-can-spend* outputs.

[11] Sometimes also referred to as *proposed*, or *published* in related literature.

The design paradigms of the underlying cryptocurrency have to be considered to assess the impact and effects of the mentioned manipulation methods. For example, the impact of transaction (re)ordering in a smart contract capable cryptocurrency, is greater than for a cryptocurrency platform which does not support smart contracts. Conversely, the censorship of undesired transactions is easier to define programmatically in an UTXO based model, as there can only be one transaction spending a certain unspent output, compared to a smart contract capable cryptocurrency where a transaction to a smart contract can be routed through several layers of contract invocations. Therefore, influence methods such as transaction ordering and exclusion have variable impact depending on the targeted platform. Similarly, the ability to invalidate a transaction can result from successfully performing one or more of the above transaction manipulation types. Thereby, the definition of "invalid" depends on the underlying cryptocurrency and is different for UTXO and account-based models. For example, to invalidate a transaction to a smart contract in Ethereum two approaches exists: Either a transaction is not accepted because a transaction with the same nonce was already included in Ethereum, or the transaction throws an exception during execution because it operates on a (unexpectedly) changed state. The first would be a result of transaction revision, while the later can happen because of a change in the order of invoked transactions or the exclusion of a previous transaction.

Some AIM attacks may allow multiple types of transaction manipulation at the same time, while others are specifically constructed to support only one method (see Table 1). Depending on the state of the targeted transaction(s) (proposed, confirmed, settled) the attack might vary in cost and in the required level of interference with consensus.

4.3 Required Interference with Consensus

While the previous classification of transaction manipulation attacks describes the intended impact, here we consider the *required interference* with consensus by which they can be achieved. Specifically, we introduce three different fork requirements:

- **Deep-fork required**, where a fork with depth of at least ℓ exceeding a security parameter k_V is necessary (i.e., $\ell > k_V$). The victim defines k_V [26,54] and it refers to its required number of confirmation blocks for accepting transactions[12]. In other words, the victim indirectly defines the required minimum fork length ℓ by his choice of k_V.
- **Near-fork required**, where the required fork depth is *not* dependent on k_V, but forks might be required. In other words, the attacker defines the gap k_{gap} (which can be smaller than k_V) he wants to overcome.[13]

[12] We emphasize that each transaction has a recipient (and thus a potential victim with an individual k_V), in practice there is no global security parameter k which holds for all transactions.

[13] The length of k_{gap} also depends on the attacker's resources and willingness to succeed (e.g., to exclude a certain block).

- **No-fork required**, where no blockchain reorganization is necessary at all (i.e., $\ell = 0$).

The required interference with consensus specifies the chain reorganization needed. A classical double-spending attack scenario [49,54] can be considered as a *transaction revision* attempt in which a single attacker aims at producing a longer chain (possibly in secret [24,52]) than the main chain to revert one (or possibly many) of *his own* transactions. Therefore, this attack requires *deep forks* ($\ell > k$) to reorganize the chain. Since the classic case attacker is assumed to have full control over the required hashrate to perform the attack, he can also arbitrarily order and exclude transactions from the longest chain. Clearly, an attacker with more than 50% of the hashrate is able to eventually produce a longer chain with probability one and thus can revert/undo any transaction and permanently perform all four kinds of transaction manipulation attacks by providing a longer chain. [14]

No-fork attacks distinguish themselves from the other two categories by aiming to manipulate miner's *block proposals* rather than (preliminary) consensus *decisions*, i.e., already mined blocks. In the context of PoW cryptocurrencies, manipulating a miner's block proposal means influencing the input block used for finding and adding a valid PoW. Deep- and near-fork attacks seek to undo state-updates to the ledger that are already confirmed by subsequent PoW.

4.4 Used Payment Method

AIM attacks either pay for compliant behaviour, or they penalize for non-compliant actions. How this mechanism is set up depends on the attack in question, but there are three general methods that differ in which currency is used for the payment.

- **In-band payment**: The payment is performed in the target cryptocurrency. Most early bribing attacks where designed to gain in-band profits, like for example checklocktime bribes [14], whale transactions [42] or history revision contracts in Ethereum [44].
- **Out-of-band payment**: The payment is performed in another currency, the so-called funding cryptocurrency. Some AIM attacks which utilize out-of-band funding where designed as Goldfinger attacks, like for example GoldfingerCon [44] and Pitchforks [31]. Others can be executed as Goldfinger attack, or with the goal to gain in-band profits, like for example [56], or the out-of-band variants of P2W attacks [32]. This highlights that AIM attacks which are intended to destroy a Cryptocurrency, i.e., perform a Goldfinger attack, inherently requite methods of out-of-band funding.
- **Threat**: No direct payment is performed, but a credible threat is constructed that non-compliant behaviour could lead to losses [11,46].

[14] Actually the heaviest chain by PoW, e.g., in Bitcoin measured in difficulty periods.

5 Classification of Existing AIM Approaches

Equipped with our generalized attack model and the classification by state of and intended impact on transactions as well as the resulting required interference with consensus, we now inspect and compare existing AIM attacks within this section. Table 1 presents an overview of our systematization of existing proposals. Each row represents a different attack (in chronological order of their release) and columns outline respective properties.

	Tx rev.	Tx ord.	Tx excl.	Tx trig.	Required interference with consensus	Attacker hashrate $p_{\mathcal{B}}$	Rational hashrate $p_{\mathcal{R}}$	Distracts hashrate	Requires smart contract	Payment	Trustless for attacker	Trustless for collaborator	Subsidy	Compensates if attack fails
Bribery [18]	✓	✗	✗	(✓)	Deep fork	$\approx (0, \frac{1}{2})$	$\approx [\frac{1}{2}, 1)$	✗	✗	in-band	✓	~	✗	✗
Dark side attack [40]	✓	(✓)	(✓)	✗	Deep fork	$\approx (0, \frac{1}{2})$	$\approx [0, 1)$	✗	✗	in-band	✗	✓	✗	✗
Feather-forks [11]	✗	✗	✓	✗	Near-/No forks	$\approx (0, \frac{1}{2})$	$\approx [\frac{1}{2} - 1)$	✗	✗	threat	-	-	-	-
Checklocktime bribes [14]	✓	✗	✗	(✓)	Deep fork	✗	$\approx [\frac{1}{2}, 1]$	✗	✗	in-band	✓	~	✗	✗
Negative fee miningpool [14]	✓	(✓)	✓	✗	Near-/No- /Deep forks	✗	$\approx [\frac{1}{2}, 1]$	✗	✗	out-of-band	✗	✗	✗	✓
Script Puzzle double-spend [56]	✓	(✓)	✓	(✓)	Deep fork	$(0, \frac{1}{2})$	$1 - p_{\mathcal{B}}$	✓	✗	in-band	~	✗	✗	~
Script Puzzle 38.2% attack [56]	✗	(✓)	✓	?†	Near-/No forks	$[0.382, \frac{1}{2})$	$1 - p_{\mathcal{B}}$	✓	?†	out-of-band	?†	?†	✗	✓
Whale Transactions [42]	✓	✗	✗	✗	Deep fork	$(0, \frac{1}{2})$	$1 - p_{\mathcal{B}}$	✗	✗	in-band	✓	~	✗	✗
Proof-of-Stale blocks [43,59]	-⋆	-⋆	-⋆	(✓)	-⋆	✗	-	✓	✓	out-of-band	~	✓	✗	✓
Fomo3D game [4]	-	-	-	✓	No fork	✗	[0,1]	✗	✓	in-band	✓	✓	✗	~
CensorshipCon [44]	✗	(✓)	✓	(✓)	Near-/No forks	$[\frac{1}{3}, \frac{1}{2})$	$[\frac{1}{3}, \frac{2}{3})$	✓	✓	in-band	~	✗	✓	✗
HistoryRevisionCon [44]	✓	✗	✗	(✓)	Deep fork	✗	$\approx [\frac{1}{2}, 1]$	✗	✓	in-band	✓	~	✓	✗
GoldfingerCon [44]	-	-	✓all	(✓)	No fork	✗	$\approx [\frac{1}{2}, 1]$	✗	✓	out-of-band	✓	✓	✗	✓
Race to the door [15]	-	-	-	✓	No fork	✗	[0,1]	✗	✓	o.o.-band/threat	✓	✓	✗	~
Pitchforks [31]	-	-	✓all	✗	No fork	✗	$(\frac{1}{3}, 1]$	~	✗	out-of-band	✓	✓	✓	✗
Front-running [22,19]	✗	✓	✗	(✓)	No fork	✗	$(0, 1]$	✗	✗	in-band	✗	✓	✗	✓
Pay per Miner Censorship [61]	✗	✗	✓	-	No fork	✗	1	✗	✓	in-band	~	~	✗	✗
Pay per Block Censorship [61]	✗	✗	✓	-	No fork	✗	1	✗	✓	in-band	~	~	✗	✓
Pay per Commit Censorship [61]	✗	✗	✓	-	Near-/No fork	✗	1	✗	✓	in-band	~	~	✗	✗
P2W Tx Excl. & Ord [32]	✗	✓	✓	(✓)	Near-/No fork	✗	$[\frac{1}{2}, 1]$	✗	✓	out-of-band	✓	✓	✗	✓
P2W Tx Rev. & Excl. & Ord. [32]	✓	✓	✓	(✓)	Deep fork	✗	$[\frac{1}{2}, 1]$	✗	✓	out-of-band	✓	✓	✗	✓
P2W Tx Ord. (in-band) [32]	✗	✓	✗	(✓)	No fork	✗	$(0, 1]$	✗	✓	in-band	✓	✓	✗	✗
P2W Tx Excl. (in-band) [32]	✗	✗	✓	(✓)	Near-/No fork	✗	$[\frac{1}{2}, 1]$	✗	✓	in-band	✓	✓	✗	✗
BDos [46]	-	-	✓all	✗	Near-/No fork	$\approx [0.21, \frac{1}{2})(for BTC)$	1	✗/✓	✗	threat	-	-	-	-
HTLC bribing [37,58]	✗	✗	✓	✗	Near-/No fork	✗	1	✗	✗	in-band	✓	~	✗	✗

Table 1: Comparison of existing AIM approaches on cryptocurrencies in *chronological order* according to their appearance.
A property is marked with ✓ if it is achieved and with ✗ otherwise, - is used if a property does not apply. If the symbol is within brackets, e.g., (✓), this means that this propery is achieved (or can be augmented), but this was initially not discussed or considered by the authors.
~ means that the property cannot be clearly mapped to any of the previously defined categories without further details or discussion which is given in the textual description.
⋆ means that this attack aims against mining pools and hence is not intended to manipulate the content of the blockchain.
† means that the paper does not explicitly specify the out-of-band payment method but assumes its correctness.

5.1 Impact on Transactions

The different ways of how AIM attacks can have an impact on transactions are outlined in Section 4.2.

Tx revision: In the first bribing attack, proposed by Bonneau [14], the use of *lock time transactions* is suggested, which are only valid on the attacker's chain, but there they can be claimed by anyone (anyone-can-spend outputs). Miners are hence expected to be incentivized to mine blocks on the attacker's chain to collect these bribes as inputs in new transactions included in their new blocks. As a by-product one transaction per new block is triggered to claim the anyone-can-spend output. Therefore, transaction triggering is technically achieved, but set into parenthesis as it is not the main intent of the attack. A variation of the checklocktime bribes which does not trigger additional transactions was proposed by Liao and Katz [42] and uses high fee transactions (*whale transactions*) to provide incentives for miners to join the attack. In [44] they proposed a smart contract (`HistoryRevisionCon`) which pays additional in-band rewards to miners of the attacker's desired Ethereum chain branch, iff the effects of the double-spending transaction have occurred on this branch. Strictly speaking, this attack also triggers transactions as the promised rewards have to be claimed by the bribees from the smart contract. The mentioned attacks ([14,42,44]) rely on in-band payments and are designed to replace or *revise* a specific transaction, i.e., perform a single double-spend. As a consequence, they do not consider defining the order or exclusion of arbitrary transactions. Except for the double-spending transaction itself, the block content of subsequent blocks can freely be defined by the bribed miners. Thus – if not explicitly considered – also the blocks produced by the bribed miners will not be fully under control of the adversary. Therefore, it would be possible for such miners to also perform a double-spend of one of their transactions for free, by piggybacking on the attack financed by the original attacker.

Tx exclusion: There is one notable exception which was specifically designed to exclude transactions: `CensorshipCon` [44] rewards mining uncle blocks to distract the hashrate of bribable miners, which in turn enables the attacker to overtake the Ethereum blockchain s.t., blocks exclusively come from the attacker. Since this attack is in-band, it only works in Ethereum and relies on the uncle block reward scheme of Ethereum to subsidise the attack, i.e., reduce the value of the required bribes. To succeed, it requires that the hashrate of the attacker is larger than $\frac{1}{3}$ and the hashrate of the bribable miners to be between $[\frac{1}{3}, \frac{2}{3})$. If the attack is successful it allows for arbitrary transaction ordering as well and thus also for arbitrary transaction exclusion, as all blocks appended to the main chain during the attack come from the attacker.

`GoldfingerCon` [44] can be seen as a special case of the transaction exclusion attack which rewards Bitcoin miners for mining empty blocks with the help of an Ethereum smart contract. In this case, all transactions are excluded to reduce the utility of the respective cryptocurrency for all its users. So called *Goldfinger attacks* have been first described by Kroll et al. [38], but Goldfinger-Con was the first practical instantiation. The name is derived from the James

Bond movie villain Goldfinger, who seeks to destroy the gold reserves stored in Fort Knox to increase the value of his own holdings. An important aspect of Goldfinger attacks is that the payments have to be performed out-of-band since, if successful, the value of the targeted cryptocurrency is intended to drop. Similarly, *Pitchforks* [31] leverage merged mining [33] to subsidize the creation of empty (or specially crafted) blocks in the attacked parent chain [31]. As with all Goldfinger-style attacks, the attacker is required to achieve utility outside of the cryptocurrency economy he wants to attack [38]. In case of the Pitchfork attack, the external utility comes from a hard-fork, which creates a new cryptocurrency. In this new cryptocurrency, the merge-mined PoW consists of blocks which attack the forked parent cryptocurrency, e.g., are empty. As the hashrate is repurposed in this case, it is technically not directed anywhere else i.e., not distracted.

Distracted hashrate is redirected from the valid tip(s) of the attacked blockchain to some other form of puzzle, or alternative branch, that does not contribute to state transitions of the targeted cryptocurrency. The *Script puzzle* 38.2% [56] and `CensorshipCon` attack [44] distract hashrate of bribable miners to gain an advantage over the remaining honest miners. The former redirects the hashrate from the main chain towards puzzles which promise more rewards than honest mining, the later rewards uncle block mining in Ethereum. The goal of both attacks is that the attacker gains the majority of the hashrate in the respective main chain, and he can hence arbitrarily exclude, or order transactions. Although, the attack does not explicitly aim to allow the specific ordering of certain transactions, this capability is achieved as a by-product. Neither attack is reverting blocks to change history, which is a different scenario and requires further analysis in this context, as reverting blocks would change the incentives of miners which have produced them.

Also in *Script Puzzle double-spend* [56] PoW like puzzles, offering in-band rewards, are published within the respective cryptocurrency with the intent to distract the hashrate of rational miners. Using the gained advantage to overtake the main chain requires attacker hashrate. Again, transaction ordering comes as a by-product and was not an explicit design goal, but theoretically this is the only existing attack utilizing *in-band* payments, which can achieve the three properties: revision, ordering and exclusion. Although, upon successful execution rational miners are deprived of their bribes as the previously hidden attack chain becomes the longest chain and does not pay the promised puzzle rewards. This renders the attack non-repeatable against rational miners.

Tx rev./ord./excl.: There are only two proposed attack methods which achieve these three properties in an out-of-band payment scenario: *negative-fee mining pools* [14] and P2W Tx Rev. & Excl. & Ord. [32]. A negative-fee mining pool is like a classic mining pool, except that it pays out an above-market return. "*Because such a pool would lose money on expectation, no honest pool should be able to match this reward*" [14]. As with most classic mining pools[15] the pool

[15] In P2Pool for example, there is no single operator which can define the content of a block proposal.

operator can define the content of a block proposal and hence forge arbitrary attack blocks. Even if miners are rational and hence willing to actively participate in such operations, this approach has at least two major limitations: First, miners would still have to trust the pool owner to pay out the promised rewards. Second, miners could report only solutions which are below the current difficulty target (shares) to prove that they are working for the pool, but withhold blocks which actually match the difficulty target. Thereby, they would potentially gain profits by pretending to participate in the attack/pool without actually doing so. This *miner's dilemma* is a general problem for mining pools [23].

The smart contract design presented in [32] resolves the limitations of negative-fee mining pools by automating the payment of bribes to complacent miners without requiring any further interaction of the attacker. Thereby, the attacker publishes block templates to the smart contract and offers a bribe for the first miner who can provide a valid PoW solution for such a template. As only payments for valid PoW solutions are provided by the smart contract, it is ensured that the actions of bribees are specifically targeted to aid the attacker. If the attacker deems that the ongoing attack is not likely to succeed, he can stop the investment of further funds by not publishing any further block templates.

Tx triggering: The are only two existing AIM techniques, which are intended to trigger transactions: The *Fomo3D game* [4] and the *race to the door* Goldfinger attack sketched by Bonneau [15]. In an race to the door, the attacker *"credibly commits"* to buy out half of all funds present in the targeted cryptocurrency, to utilize them for destroying the system. Therefore, the price the attacker has to pay for those funds is likely to drop the more users decide to sell, increasing the likelihood of the attack to succeed. This creates a vicious cycle, resulting in a race to the door. The idea was not presented in great detail and mainly discussed in context of overtaking PoS/PoW cryptocurrencies, but of course such an attack would also trigger sell transactions. Moreover, there are plenty of ways to attack the value of a cryptocurrency while holding substantial amounts of it that are left unexplored.

There are multiple variants of Fomo3D, but roughly the rules are as follows. In this game, which is open for everybody, the last account which has purchased a ticket wins when a timer goes to zero and every purchase again increases the timer by 30 seconds. This leads to the situation that transactions are triggered by rational players as soon as the timer gets close to zero. It was conjectured that the game would never end, but in august 2018 the first round of the game ended and the winner collected $10,469$ Ether ($\approx \$2.1M$ USD at that time)[16]. It can be argued that a single instance of this game does not qualify as an "attack", but the same concept of presumably "free money" available to grab from a smart contract can also be used as an attack method (see our discussion in Section 7). The interesting aspect about these tx triggering attacks is, that they have effects

[16] The winner flooded the network with unrelated high gas transactions to custom smart contracts which congested the network blocking other "last" payments to the game.

for any hashrate of rational miners as long as there are rational clients. Even if $p_{\mathcal{R}} = 0$, rational clients in the network will issue transactions.
Tx ordering: Dedicated ordering attacks, like front-running [22,19], P2W Tx Excl.& Ord., or P2W Tx Ord. (in-band) [32], target unconfirmed or confirmed transactions and therefore are cheaper as their interference with consensus is less severe.

5.2 Required Interference with Consensus

The concept of required interference with consensus is outlined in Section 4.3 and classifies if an attack can be realized without, with a near- or with a deep-fork. Depending on the scenario and the desired attack outcome, e.g., if only ordering is relevant, deep forks are not necessarily required. For example if, the victim accepts unconfirmed transactions, transaction revision can happen without any fork by simply updating the transaction. Bitcoin [2] as well as Ethereum allow something like *replace-by-fee* i.e., if there is a transaction signed by the same sender with the same nonce but a significantly higher gas value [3], the transaction with the higher gas value replaces the original one in certain clients. This circumstance is also used in the context of *front-running* [22,19]. But front-running is only a subset of possible (re-)ordering attacks, as it might be desirable to place a transactions more accurately in between two other transaction, e.g., as required for exploiting the `BlockKing` contract [53].

Prior to 2018, ordering attacks on smart contract cryptocurrencies have not been intensively studied [53,34]. This has recently changed as order fairness has been exposed as a fundamental issue in leader based consensus protocols [19,35,39]. In context of Nakamoto consensus, every miner that is capable of producing blocks can define the order of the transactions in his blocks. This circumstance alone can be used to gain an advantage in certain scenarios e.g., where transactions race against each other to collect something that is claimable by everybody like an *anyone-can-spend* transaction, the reward of a puzzle, or arbitrage[17]. But when rational actors are assumed, there are also scenarios where the ordering of transactions can be manipulated by attackers which are not necessarily miners themself, but have funds at their disposal to launch incentive attacks. In classical front-running miners are incentivized to prioritize transactions because they carry a larger fee. This however is not a consensus rule and thus lacks enforcement, as transaction with the highest fee can still be included at the end of a block, resulting in an all-pay auction [19]. In [32] an in-band as well as an out-of-band AIM attack is proposed, which allow arbitrary transaction ordering while only paying if the desired ordering is observed. Both attacks can be executed without any hashrate assuming rational miners.

[17] Interestingly the problem of racing transaction was known very early on in the cryptocurrency community, which lead to the first fork of Bitcoin, i.e., Namecoin [1,33], which introduced a commit reveal scheme to prevent races while registering domain names on the blockchain.

5.3 Required Hashrate

Required attacker hashrate $p_{\mathcal{B}}$ specifies how much hashrate is required to be under direct control of the attack (without considering the effects of AIM) for the attack to be successful. As observable in Table 1 there are three attacks which require $p_{\mathcal{B}} > 0$. The Script Puzzle 38.2% attack is designed to overtake the blockchain entirely by offering alternative script puzzles with higher rewards to distract the hashrate of rational miners. This allows an adversary with appropriate hashrate to establish a computational majority and gain a net profit without considering double-spending attacks. In Script Puzzle double-spend the adversary has no explicit minimum hashrate requirement, however low hashrate has to be compensated with more puzzle funds. Moreover, it is designed as a single-shot double-spending attack that, if successful, deprives rational miners of their bribes. `CensorshipCon` uses a smart contract to offer in-band bribes for mining uncle blocks to distract hashrate. Thus, it requires attacker hashrate to include uncle blocks from rational miners in the main chain. Since it has to include all mined uncle blocks, it requires the hashrate of the attacker to be larger than $\frac{1}{3}$ and the hashrate of the bribable miners to be between $[\frac{1}{3}, \frac{2}{3})$.

It makes sense to bound the attacker hashrate below $\frac{1}{2}$ since otherwise the attacker has no need to perform bribing attacks as he could overtake the chain single handedly.

Required rational hashrate $p_{\mathcal{R}}$ specifies how much hashrate is requited to be under control of rational miners for the attack to have a chance to succeed as described and evaluated in the respective paper. Generally, all bribing attacks have to assume that at least some of the miners are rational and hence bribable. Generally, it makes sense to assume that more than half of the miners are rational s.t. attacks have at realistic change to win longer block races. Both Script Puzzle attacks require all miners to be rational, i.e., $p_{\mathcal{B}} + p_{\mathcal{R}} = 1$, as well as the `Pay per ...` attacks ($p_{\mathcal{R}} = 1$).

However, the attacks observed in practise provide no guarantees for the attacker that the desired ordering is achieved even if the highest transaction fee has been paid as the resulting game is an all pay auction [19].

5.4 Payment Method

This specifies where the payments to the bribees are performed (see 4.4). It can be argued that miners will try not to harm the value of their own cryptocurrency holdings by accepting in-band bribes, hence out-of-band AIM are of particular interest. **Subsidy** means that the attack leverages some characteristic of the cryptocurrency, or the environment to become cheaper. In case of CensorshipCon the rewards from uncle blocks are used to subsidize the attack, whereas in Pitchforks the additional income from merged mining is used as an incentive.

Compensates if attack fails refers to the property that at least a portion of the bribe is paid irrespective of the outcome. To successfully engage rational miners, attacks such as Checklocktime bribes [14], Whale Transactions [42] and HistoryRevisionCon [44], must pay high rewards in case of success to compensate the financial risk faced by bribees if the attack fails despite of their participation. So far the only attack which facilitates transaction revision that achieves this property is [32]. Script Puzzle double-spend defrauds the bribed miners if successful and hence actually only pays out rewards if it fails. In front-running attacks, high transaction fees are usually incurred even if the desired ordering effect is not achieved. Thus, in this case it is also an undesirable property for the attacker. The same holds true for negative-fee mining pools as rewards have to be paid for performed work even if no attack block fulfilling the difficulty target has been submitted by a miner.

5.5 Trustlessness

Trustless for attacker specifies if the attack itself can be exploited by allowing collaborating/bribed miners to profit without adhering to the attack. For example, Script Puzzle attacks require some form of freshness guarantee to prevent bribees from intentionally waiting until the attack fails before computing puzzle solutions to obtain rewards. It is also possible to claim rewards for stale honest blocks that are later on submitted as uncles to the CensorshipCon. Also in naive front-running attacks the attacker has no guarantee that the desired ordering will be achieved by paying a high fee. The Pay per ... attacks are only modelled theoretically without providing concrete instantiation. Therefore, it cannot be evaluated in this regard.
Trustless for collaborator specifies if bribees have to trust the attacker that they will receive their payments, if they adhere to the attack. In Checklock-time bribes a lock time on individual transaction outputs intends to ensure that they cannot be spent before a particular block height, even by the creator. This ensures that at each height a locked output is released and split into a anyone-can-spend and another locked output. However, the holder of the associated private key can cheat, by creating a conflicting/racing transaction, which also becomes valid after the intended lock time has passed. This conflicting transaction, transfers the whole output back to the owner without an additional anyone-can-spend output. However, this attempt is only possible if the attacker is under control of some hashrate $p_{\mathcal{B}} > 0$, as a miner would never prefer this transaction before the other. The same holds true for Whale Transactions, or HTLC bribes since the attacker has to provide new high fee transactions for each block on the attack chain at each step of the attack. While HistoryRevisionCon does not explicitly consider trustlessness for collaborating miners, an augmentation is possible[18], CensorshipCon requires that the attacker includes blocks produced by collaborating miners as uncle blocks and thus is not trustless. The Script Puzzle double-spend attack is designed as a one-shot attack that defrauds collaborators. The Script Puzzle 38.2% attack does not specify how payments are performed and assumes a working trusteless out-of-band payment method.

[18] The issue stems from the fact that the bribing contract checks the balance of the Ethereum account which should receive the bribing funds before issuing any bribes, but without any additional locking constraints these funds can be moved by the attacker once received.

6 Costs, Profits and Extractable Value

In this section we want to highlight the challenges of comparing existing AIM attacks with respect to their costs and potential profits.

First of all, the presented attacks differ significantly with respect to their system- and attack models, which have diverse goals regarding their intended influence on transactions (revision, ordering, exclusion, triggering), as well as varying assumptions regarding the capabilities of the attacker, e.g., hashrate and funds.

Second, not all existing proposals have analyzed the involved costs and gains in a comparable way. Attacks such as the Script Puzzle double-spend or CensorshipCon express the required funds in terms of the hashrate which is also required to successfully execute it [56,44]. For transaction revision using Whale Transaction or P2W attacks [42,32] concrete values are provided while at the same time no hashrate is required. In GoldfingerCon [44] only the costs of invoking the smart contract are provided.

Costs: What stands out in the comparison of costs is that: i) Attacks which compensate collaborating rational miners even if the attack fails are cheaper. The reason for this is that such attacks do not have to provide high bribes to account for the risks faced by bribees if the attack is unsuccessful [61,32]. ii) Attacks which exclusively focus on transaction exclusion or (re)ordering of unconfirmed transactions are substantially cheaper as they only compete with the fee, i.e., extractable value, of the transaction(s) in conflict [61,32,19,37,58].

Profit: To calculate the profit of the attack it is important to estimate the costs as well as the extractable value. In this context, the term *miner extractable value* [19] has been coined to describe the value which can be extracted by a miner by including a certain transaction in terms of fees or guaranteed profits through token arbitrage. In relation to other AIM attacks surveyed in this paper, this leads to an interesting observation: We argue that the extractable value of a transaction for a certain party can not readily be determined by exclusively looking at the cryptocurrency system in which this transaction is to be performed. The reason is that there might be additional protocols like colored coins [50] or out-of-band payments from AIM attacks at play, which can influence the (miner) extractable value of a given transaction. This is an instantiation of a more general observation that game-theoretic analysis is not composable.

The question whether AIM is profitable can be summarized by comparing the extractable value as well as the costs of the attack and the behaviour intended by the protocol designer. The following simplified equation was adapted from Böhme [7].

$$\texttt{EV}(\text{attack}) - \mathit{costs}_{\text{ attack}} > \texttt{EV}(\text{follow protocol}) - \mathit{costs}_{\text{ follow protocol}}$$

Let's assume two unconfirmed, but conflicting Bitcoin transactions (tx_1, tx_2) are competing for a place in the next block. If the extractable fee of one transaction is greater than for the other $\texttt{Fee}(tx_1) > \texttt{Fee}(tx_2)$, it would be rational from the miner to include tx_1, since $\texttt{EV}(tx_1) = \texttt{Fee}(tx_1)$. But if there is a side payment, due to an AIM attack on a different funding cryptocurrency (e.g., Ethereuem) for including tx_2, which leads to the situation that $\texttt{EV}(tx_2) > \texttt{EV}(tx_1)$, then the situation for the rational miner changes. In this case, the reason for the change is not directly visible in Bitcoin.

The question whether it is possible to upper-bound the extractable value was also touched by Budish [17] in a different setting and from the perspective of double-spending attacks only. Under a simplified model, the extractable value of a double-spend is the transferred value of coins [19]. To calculate the required rewards and fees for making double-spending attacks economically unattractive, the author assumed that in the worst case every transaction in a block is potentially up for double-spending and highlights that the relation between reward and fees, compared to the value transferred in Bitcoin makes such attacks economically feasible in theory. An instantiation of an attack in which every transaction of a block can theoretically become a target for double-spending, has been proposed in [32], where a crowdfunded attack is described, utilizing smart-contracts. The goal is to distribute the costs of multiple double-spend attempts in the same block to the set of transacting entities.

By these examples, we see that it is hard or even impossible to accurately bound the extractable value of transactions (and thus blocks) in a multi cryptocurrency ecosystem by solely looking at data from one cryptocurrency. A related meta argument was presented in [25].

7 Discussion

We finally discuss the relation of AIM to other ways of gaining capacity in Nakamoto consensus, as well as highlight open questions and directions for future work in this area.

Relationship of AIM to other ways of gaining capacity: In the paper [15] an excellent classification of different methods on how to gain capacity in Nakamoto consensus is provided. These methods are separated into: *rent*, *build*, *bribe* and *buy out*. Hereby, rent, buy out as well as build refer to classical methods of renting hardware, buying cryptocurrency units at exchanges, or building new datacenters for mining. We augment this classification and argue that AIM can be used to construct algorithmic ways for all these methods. Table 2 depicts an augmented version from [15] showing the different methods of how to obtain capacity in Nakamoto consensus.

According to the original classification in [15], bribing is a *temporary* attack, which utilizes *existing* resources of miners. If the terms *new* and *existing*, in the context of PoW capacity, are to be interpreted from the perspective of the targeted system, then some existing attacks which rely on out-of-band payments would also classify as *rent*. The reason for this is: They are also able to attract new capacity currently bound in other cryptocurrencies which utilize the same

[19] The dependency between transaction value and confirmation time k_V, is also discussed in [54].

PoW algorithm, like [56,31,32]. Capacity which is present in a different cryptocurrency is also new to the targeted cryptocurrency if miners decide to switch for supporting an attack.

We further argue that *buy out* attacks can theoretically be done algorithmically using cross-chain atomic swaps [30] (or any other blockchain interlinking protocol). A *race to the door* style attack [15] in combination with cross-chain atomic swaps can be imagined to perform Goldfinger style attacks on smart contract capable PoW cryptocurrencies. Hereby, out-of-band payments are used to buy out cryptocurrency units, through a smart contract, which is going to use these previously bought cryptocurrency assets to perform a denial of service attack by dumping the previously bought crypto assets on the market as freely available for anyone to claim after a certain timeout. If there is a limit for what is claimable per transaction, as well as the requirement of a high fee, this on-chain faucet construction will trigger a flood of transactions as soon as the timeout is reached. In this case, existing funds are bought and permanently redistributed with the intent to perform a denial-of-service attack and at the same time collapse the market due to increased supply.

It remains to be shown that it is theoretically possible to build *permanent* AIM attacks. Arguably, any Goldfinger attack, such as GoldfingerCon [44], which creates enough external utility to refuel the attack, can in theory be constructed in a way to run permanently. Although, it is unlikely that a Goldfinger attack has to be continued infinitely long if the intended effects have already occurred. An attack which also discusses its perpetuity is the Script puzzle 38.2% attack. In this case the attack can also theoretically be used to permanently overtake the chain by supplying puzzles that provide out-of-band reward and thereby overtake the original blockchain with 38.2% of the total hashrate. Also, the pitchfork [31], in which the additional revenue stream to sustain the attack comes from a fork of the targeted cryptocurrency and not from a previously determined bribing fund, can in theory be sustained infinitely long. Whether the attack can be sustained depends on the value of the newly generated cryptocurrency. An interesting analogy exists between any permanent AIM attack and a cryptocurrency itself. From the perspective of a miner who exclusively mines on puzzles for any of these three permanent attacks, there is no difference to mining on any other PoW based cryptocurrency other than the format of the associated PoW.

Mitigation and counter attacks: The presented systematization has a very attack centric view on the issue at hand. This is is due to the selection of papers, which almost all have a very attack-focused viewpoint. Therefore, counter measures and counter attacks are often omitted in these papers, or not discussed to a great extent.

Nevertheless, for the victim(s) counter bribing might be a viable strategy against AIM. The difficulty of successfully executing counter bribing highly depends on the respective scenario. In the end, counter bribing can also be countered by counter-counter bribing and so forth. Therefore, as soon as this route is taken, the result becomes a bidding game. Against transaction exclusion attacks, counter bribing can be performed by increasing the fee of the transaction to be excluded such that it surpasses the value promised for not including the transaction. If defenders have imperfect information, they may not be able to immediately respond with counter bribes. In this case some of the attack chain blocks may have already been mined, or even take the lead, before they are recognized by defenders. Counter bribing then necessitates a fork, and thus a more expensive transaction revision attack, leading to asymmetric costs in the bidding game. This illustrates an important aspect of AIM, namely their visibility. On the one hand, sufficiently many rational miners of the targeted cryptocurrency have to recognize that an attack is occurring, otherwise they won't join in and the attack is likely to fail. On the other hand, if the victims of the attack recognize its existence, they can initiate and coordinate a counter bribing attack. So the optimal conditions for AIM arise if all rational miners have been informed directly about the attack, while all victims/merchants do not monitor the chain to check if an attack is going on and are not miners themselves. If the payments are made out-of-band, they are rendered more stealthy to victims who only monitor the targeted cryptocurrency. It can hence be argued that counter attacks by victims are harder to execute as they are not immediately aware of the bribing value that is being bet against them on a different funding cryptocurrency. We also follow the argument in [14] that requiring clients to monitor the chain and actively engage in counter bribing is undesirable, and out-of-band attacks further amplify this problem as clients would have to concurrently monitor a variety of cryptocurrencies.

To prevent repercussions, participating miners can make use of the fact that the PoW mining process itself does not require any strong identity by using different payout addresses. Of course their received rewards can be traced, but available privacy techniques could be used to camouflage the real recipient of the funds, e.g., [51,45,29].

Situation in PoS: Since all considered attacks target PoW cryptocurrencies, the applicability of AIM on PoS cryptocurrencies is not sufficiently understood yet. It remains to be understood which techniques are transferable to PoS cryptocurrencies, and which additional mitigations (e.g., providing collateral, slashing) can increase the induced costs of attacks in this setting.

Rationality & Practicality of attacks: All AIM attacks assume some form of rational behaviour of participants. In practise although, it is hard to define rational behaviour in a general way, as also the individual investments and the long term interests of miners play an important role. Although, there may be scenarios where miners are capable of providing PoW for a targeted cryptocurrency, but at the same time do not have any long-term interest in the well-being of the target. Consider the real-world example of Bitcoin and BitcoinCash which utilize the same form of PoW and can be considered rivals. Thus, the question if the proposed attacks are possible in practice is difficult to answer scientifically. There is already empirical evidence from previous large scale attacks by miners, especially on smaller cryptocurrencies as well as AIM attacks [9,6,5,8,19]. These cases demonstrate that large scale attacks happen and that the topic of incentives in cryptocurrencies is an area deserves further study.

8 Acknowledgements

We would like to thank the participants of the Dagstuhl Seminar 18152 (*Blockchains, Smart Contracts and Future Applications*), especially Samuel Christie and Sebastian Faust, as well as the participants of the Dagstuhl Seminar 18461 (*Blockchain Security at Scale*) for all the frutiful discussions.

References

1. Namecoin, https://www.namecoin.org/, accessed: 2020-09-15
2. Replace by fee in bitcoin, https://en.bitcoin.it/wiki/Replace_by_fee, accessed: 2020-12-23
3. Replace by fee in openethereum, https://openethereum.github.io/Transactions-Queue.html, accessed: 2020-12-23
4. How the winner got fomo3d prize - a detailed explanation. medium (2018), https://medium.com/coinmonks/how-the-winner-got-fomo3d-prize-a-detailed-explanation-b30a69b7813f, accessed: 2020-09-15
5. Bitcoin cash miners undo attacker's transactions with 51% attack. coindesk (2019), https://www.coindesk.com/bitcoin-cash-miners-undo-attackers-transactions-with-51-attack, accessed: 2020-09-15
6. Ethereum classic 51% attack the reality of proof-of-work. cointelegraph (2019), https://cointelegraph.com/news/ethereum-classic-51-attack-the-reality-of-proof-of-work, accessed: 2020-09-15
7. Talk: A primer on economics for cryptocurrencies. School of Blocks, Blockchain summer school at TU Wien (2019), https://bdlt.school/files/slides/talk-rainer-b%C3%B6hme-a-primer-on-economics-for-cryptocurrencies.pdf, accessed: 2020-09-15
8. Bitcoin gold (btg) was 51% attacked. github (2020), https://gist.github.com/metalicjames/71321570a105940529e709651d0a9765, accessed: 2020-09-15
9. Ethereum classic suffers second 51% attack in a week. coindesk (2020), https://www.coindesk.com/ethereum-classic-suffers-second-51-attack-in-a-week, accessed: 2020-09-15
10. Aiyer, A.S., Alvisi, L., Clement, A., Dahlin, M., Martin, J.P., Porth, C.: Bar fault tolerance for cooperative services. In: ACM SIGOPS operating systems review. vol. 39, pp. 45–58. ACM (2005), http://www.dcc.fc.up.pt/~Ines/aulas/1314/SDM/papers/BAR%20Fault%20Tolerance%20for%20Cooperative%20Services%20-%20UIUC.pdf
11. socrates1024 (Andrew Miller): Feather-forks: enforcing a blacklist with sub-50hash power, https://bitcointalk.org/index.php?topic=312668, accessed: 2021-1-31
12. Badertscher, C., Garay, J.A., Maurer, U., Tschudi, D., Zikas, V.: But why does it work? A rational protocol design treatment of bitcoin. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II. Lecture Notes in Computer Science, vol. 10821, pp. 34–65. Springer (2018). https://doi.org/10.1007/978-3-319-78375-8_2, https://eprint.iacr.org/2018/138.pdf
13. Badertscher, C., Maurer, U., Tschudi, D., Zikas, V.: Bitcoin as a transaction ledger: A composable treatment. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara,

CA, USA, August 20-24, 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10401, pp. 324–356. Springer (2017). https://doi.org/10.1007/978-3-319-63688-7_11, https://eprint.iacr.org/2017/149.pdf
14. Bonneau, J.: Why buy when you can rent? bribery attacks on bitcoin consensus. In: BITCOIN '16: Proceedings of the 3rd Workshop on Bitcoin and Blockchain Research (February 2016), http://fc16.ifca.ai/bitcoin/papers/Bon16b.pdf
15. Bonneau, J.: Hostile blockchain takeovers (short paper). In: 5th Workshop on Bitcoin and Blockchain Research, Financial Cryptography and Data Security 18 (FC). Springer (2018), http://fc18.ifca.ai/bitcoin/papers/bitcoin18-final17.pdf
16. Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J.A., Felten, E.W.: SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. In: IEEE Symposium on Security and Privacy (2015), http://www.ieee-security.org/TC/SP2015/papers-archived/6949a104.pdf
17. Budish, E.: The economic limits of bitcoin and the blockchain. Tech. rep., National Bureau of Economic Research (2018), https://faculty.chicagobooth.edu/eric.budish/research/Economic-Limits-Bitcoin-Blockchain.pdf
18. Cunicula: Bribery: The double double spend. Bitcoin Forum, https://bitcointalk.org/index.php?topic=122291, accessed: 2021-1-31
19. Daian, P., Goldfeder, S., Kell, T., Li, Y., Zhao, X., Bentov, I., Breidenbach, L., Juels, A.: Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In: 2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020. pp. 910–927. IEEE (2020). https://doi.org/10.1109/SP40000.2020.00040, https://arxiv.org/pdf/1904.05234.pdf
20. Dembo, A., Kannan, S., Tas, E.N., Tse, D., Viswanath, P., Wang, X., Zeitouni, O.: Everything is a race and nakamoto always wins (2020)
21. Dwork, C., Naor, M.: Pricing via processing or combatting junk mail. In: Annual International Cryptology Conference. pp. 139–147. Springer (1992), https://web.cs.dal.ca/~abrodsky/7301/readings/DwNa93.pdf
22. Eskandari, S., Moosavi, S., Clark, J.: Sok: Transparent dishonesty: Front-running attacks on blockchain. In: Bracciali, A., Clark, J., Pintore, F., Rønne, P.B., Sala, M. (eds.) Financial Cryptography and Data Security - FC 2019 International Workshops, VOTING and WTSC, St. Kitts, St. Kitts and Nevis, February 18-22, 2019, Revised Selected Papers. Lecture Notes in Computer Science, vol. 11599, pp. 170–189. Springer (2019). https://doi.org/10.1007/978-3-030-43725-1_13, https://arxiv.org/pdf/1902.05164.pdf
23. Eyal, I.: The miner's dilemma. In: Security and Privacy (SP), 2015 IEEE Symposium on. pp. 89–103. IEEE (2015), http://arxiv.org/pdf/1411.7099
24. Eyal, I., Sirer, E.G.: Majority is not enough: Bitcoin mining is vulnerable. In: Financial Cryptography and Data Security. pp. 436–454. Springer (2014), http://arxiv.org/pdf/1311.0243
25. Ford, B., Böhme, R.: Rationality is Self-Defeating in Permissionless Systems (2019), https://arxiv.org/pdf/1910.08820.pdf, _eprint: arXiv:1910.08820
26. Garay, J.A., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol: Analysis and applications. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II. Lecture Notes in Computer Science, vol. 9057, pp. 281–310. Springer (2015). https://doi.org/10.1007/978-3-662-46803-6_10, https://eprint.iacr.org/2014/765.pdf

27. Garay, J.A., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol with chains of variable difficulty (2016), http://eprint.iacr.org/2016/1048.pdf, accessed: 2017-02-06
28. Gai, P., Kiayias, A., Russell, A.: Tight consistency bounds for bitcoin. Cryptology ePrint Archive, Report 2020/661 (2020), https://eprint.iacr.org/2020/661
29. Heilman, E., Baldimtsi, F., Goldberg, S.: Blindly signed contracts: Anonymous on-blockchain and off-blockchain bitcoin transactions. Cryptology ePrint Archive, Report 2016/056 (2016), https://eprint.iacr.org/2016/056.pdf, accessed: 2017-10-03
30. Herlihy, M.: Atomic cross-chain swaps. In: Newport, C., Keidar, I. (eds.) Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018, Egham, United Kingdom, July 23-27, 2018. pp. 245–254. ACM (2018), https://arxiv.org/pdf/1801.09515.pdf
31. Judmayer, A., Stifter, N., Schindler, P., Weippl, E.: Pitchforks in cryptocurrencies: Enforcing rule changes through offensive forking- and consensus techniques (short paper). In: CBT'18: Proceedings of the International Workshop on Cryptocurrencies and Blockchain Technology (Sep 2018), https://www.sba-research.org/wp-content/uploads/2018/09/judmayer2018pitchfork_2018-09-05.pdf
32. Judmayer, A., Stifter, N., Zamyatin, A., Tsabary, I., Eyal, I., Gai, P., Meiklejohn, S., Weippl, E.: Pay to win: Cheap, crowdfundable, cross-chain algorithmic incentive manipulation attacks on pow cryptocurrencies. Cryptology ePrint Archive, Report 2019/775 (2019), https://eprint.iacr.org/2019/775
33. Judmayer, A., Zamyatin, A., Stifter, N., Voyiatzis, A.G., Weippl, E.: Merged mining: Curse or cure? In: CBT'17: Proceedings of the International Workshop on Cryptocurrencies and Blockchain Technology (Sep 2017), https://eprint.iacr.org/2017/791.pdf
34. Kalra, S., Goel, S., Dhawan, M., Sharma, S.: ZEUS: Analyzing Safety of Smart Contracts. In: 25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018. The Internet Society (2018), http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_09-1_Kalra_paper.pdf
35. Kelkar, M., Zhang, F., Goldfeder, S., Juels, A.: Order-fairness for byzantine consensus. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III. Lecture Notes in Computer Science, vol. 12172, pp. 451–480. Springer (2020). https://doi.org/10.1007/978-3-030-56877-1_16, https://eprint.iacr.org/2020/269
36. Ketsdever, S., Fischer, M.J.: Incentives don't solve blockchain's problems (2019), https://arxiv.org/pdf/1905.04792.pdf
37. Khabbazian, M., Nadahalli, T., Wattenhofer, R.: Timelocked bribes. Cryptology ePrint Archive, Report 2020/774 (2020), https://eprint.iacr.org/2020/774
38. Kroll, J.A., Davey, I.C., Felten, E.W.: The economics of bitcoin mining, or bitcoin in the presence of adversaries. In: Proceedings of WEIS. vol. 2013, p. 11 (2013), https://pdfs.semanticscholar.org/c55a/6c95b869938b817ed3fe3ea482bc65a7206b.pdf
39. Kursawe, K.: Wendy, the good little fairness widget. IACR Cryptol. ePrint Arch. **2020**, 885 (2020), https://eprint.iacr.org/2020/885
40. Lerner, S.D.: The bitcoin eternal choice for the dark side attack (ecdsa), https://bitslog.com/2013/06/26/the-bitcoin-eternal-choice-for-the-dark-side-attack-ecdsa/, accessed: 2021-1-31

41. Li, H.C., Clement, A., Wong, E.L., Napper, J., Roy, I., Alvisi, L., Dahlin, M.: Bar gossip. In: Proceedings of the 7th symposium on Operating systems design and implementation. pp. 191–204. USENIX Association (2006), `http://www.cs.utexas.edu/users/dahlin/papers/bar-gossip-apr-2006.pdf`
42. Liao, K., Katz, J.: Incentivizing blockchain forks via whale transactions. In: International Conference on Financial Cryptography and Data Security. pp. 264–279. Springer (2017), `http://www.cs.umd.edu/~jkatz/papers/whale-txs.pdf`
43. Luu, L., Velner, Y., Teutsch, J., Saxena, P.: Smartpool: Practical decentralized pooled mining. In: Kirda, E., Ristenpart, T. (eds.) 26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017. pp. 1409–1426. USENIX Association (2017), `http://eprint.iacr.org/2017/019.pdf`
44. McCorry, P., Hicks, A., Meiklejohn, S.: Smart contracts for bribing miners. In: 5th Workshop on Bitcoin and Blockchain Research, Financial Cryptography and Data Security 18 (FC). Springer (2018), `http://fc18.ifca.ai/bitcoin/papers/bitcoin18-final14.pdf`
45. Meiklejohn, S., Mercer, R.: Möbius: Trustless tumbling for transaction privacy. Proc. Priv. Enhancing Technol. **2018**(2), 105–121 (2018). https://doi.org/10.1515/popets-2018-0015, `http://eprint.iacr.org/2017/881.pdf`
46. Mirkin, M., Ji, Y., Pang, J., Klages-Mundt, A., Eyal, I., Juels, A.: Bdos: Blockchain denial-of-service. In: Proceedings of the 2020 ACM SIGSAC conference on Computer and Communications Security. pp. 601–619 (2020)
47. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (Dec 2008), `https://bitcoin.org/bitcoin.pdf`, accessed: 2015-07-01
48. Pass, R., Seeman, L., Shelat, A.: Analysis of the blockchain protocol in asynchronous networks. In: Coron, J., Nielsen, J.B. (eds.) Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10211, pp. 643–673 (2017). https://doi.org/10.1007/978-3-319-56614-6_22, `https://doi.org/10.1007/978-3-319-56614-6_22`
49. Rosenfeld, M.: Analysis of hashrate-based double spending (2014), `https://arxiv.org/pdf/1402.2009.pdf`, accessed: 2016-03-09
50. Rosenfeld, M.: Overview of colored coins (2012), `https://bitcoil.co.il/BitcoinX.pdf`, accessed: 2016-03-09
51. Ruffing, T., Moreno-Sanchez, P., Kate, A.: Coinshuffle: Practical decentralized coin mixing for bitcoin. In: Computer Security-ESORICS 2014. pp. 345–364. Springer (2014), `http://crypsys.mmci.uni-saarland.de/projects/CoinShuffle/coinshuffle.pdf`
52. Sapirshtein, A., Sompolinsky, Y., Zohar, A.: Optimal selfish mining strategies in bitcoin. In: Grossklags, J., Preneel, B. (eds.) Financial Cryptography and Data Security - 20th International Conference, FC 2016, Christ Church, Barbados, February 22-26, 2016, Revised Selected Papers. Lecture Notes in Computer Science, vol. 9603, pp. 515–532. Springer (2016). https://doi.org/10.1007/978-3-662-54970-4_30, `http://arxiv.org/pdf/1507.06183.pdf`
53. Sergey, I., Kumar, A., Hobor, A.: Temporal properties of smart contracts. In: Leveraging Applications of Formal Methods, Verification and Validation. Industrial Practice - 8th International Symposium, ISoLA 2018, Limassol, Cyprus, November 5-9, 2018, Proceedings, Part IV. pp. 323–338 (2018), `https://ilyasergey.net/papers/temporal-isola18.pdf`

54. Sompolinsky, Y., Zohar, A.: Bitcoin's security model revisited (2016), `http://arxiv.org/pdf/1605.09193.pdf`, accessed: 2016-07-04

55. Stifter, N., Judmayer, A., Schindler, P., Zamyatin, A., Weippl, E.: Agreement with satoshi - on the formalization of nakamoto consensus. Cryptology ePrint Archive, Report 2018/400 (2018), `https://eprint.iacr.org/2018/400.pdf`

56. Teutsch, J., Jain, S., Saxena, P.: When cryptocurrencies mine their own business. In: Financial Cryptography and Data Security (FC 2016) (Feb 2016), `https://www.comp.nus.edu.sg/~prateeks/papers/38Attack.pdf`

57. Tsabary, I., Eyal, I.: The gap game. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 713–728. ACM (2018), `https://arxiv.org/pdf/1805.05288.pdf`

58. Tsabary, I., Yechieli, M., Eyal, I.: MAD-HTLC: because HTLC is crazy-cheap to attack. CoRR **abs/2006.12031** (2020), `https://arxiv.org/abs/2006.12031`

59. Velner, Y., Teutsch, J., Luu, L.: Smart contracts make bitcoin mining pools vulnerable. In: Brenner, M., Rohloff, K., Bonneau, J., Miller, A., Ryan, P.Y.A., Teague, V., Bracciali, A., Sala, M., Pintore, F., Jakobsson, M. (eds.) Financial Cryptography and Data Security - FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10323, pp. 298–316. Springer (2017). https://doi.org/10.1007/978-3-319-70278-0_19, `https://fc18.ifca.ai/bitcoin/papers/bitcoin18-final14.pdf`

60. Vukolić, M.: The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In: International Workshop on Open Problems in Network Security. pp. 112–125. Springer (2015), `http://vukolic.com/iNetSec_2015.pdf`

61. Winzer, F., Herd, B., Faust, S.: Temporary censorship attacks in the presence of rational miners. In: 2019 IEEE European Symposium on Security and Privacy Workshops, EuroS&P Workshops 2019, Stockholm, Sweden, June 17-19, 2019. pp. 357–366. IEEE (2019). https://doi.org/10.1109/EuroSPW.2019.00046, `https://eprint.iacr.org/2019/748`

A Example Use of Our Classification Framework

Whether an attack is executable with or without a fork depends on the intended impact on transactions as well as on the state of the targeted transaction. For example, transaction revision where the victim accepts $k_V = 0$ (zero confirmations) may be executable as no-fork attacks. Other attacks, such as performing a *double spend* where the victim has been carefully chosen k_V [54], may require deep-forks because they need to substantially affect consensus and violate the security assumption that the common prefix of the blockchain remains stable. Transaction exclusion (*censorship*) may require near-forks to exclude the latest blocks which include the respective transaction.

With our classification framework, we can map *front-running* [22,19,32] as an attack which aims to influence transaction ordering, while targeting unconfirmed transactions (state of targeted transactions). Compared to that, the so called *time-bandit attack* [19] also aims to influence transaction ordering, but targets confirmed or even agreed transactions. Note that strictly speaking a time-bandit attack is not AIM, as it does not incentivize other participants to aid the attack, but instead relies on “classic” methods like performing a rental attack to temporarily hold the majority of the hashrate.

B Ways to gain capacity in Nakamoto Consensus

			duration of control	
			temporary	**permanent**
Source	PoW	**new**	rent	build
			AIM	AIM
	PoW & PoS	**existing**	bribe	buy out
			AIM	AIM

Table 2: Strategies to gain capacity in Nakamoto consensus according to [15], augmented with AIM strategies (colored background).