# Incentivizing Blockchain Forks via Whale Transactions

Kevin Liao[1]* and Jonathan Katz[2]

[1] Arizona State University
kevinliao@asu.edu
[2] University of Maryland
jkatz@cs.umd.edu

**Abstract.** Bitcoin's core innovation is its solution to double-spending, called Nakamoto consensus. This mechanism provides a probabilistic guarantee that transactions will not be reversed once they are sufficiently deep in the blockchain, assuming an attacker controls a bounded fraction of mining power in the network.

We show, however, that when miners are rational this guarantee can be undermined by a *whale attack* in which an attacker issues an off-the-blockchain *whale transaction* with an anomalously large transaction fee in an effort to convince miners to fork the current chain. We carry out a game-theoretic analysis and simulation of this attack, and show conditions under which it yields an expected positive payoff for the attacker.

## 1 Introduction

Decentralized cryptocurrencies have generated considerable interest in recent years. Bitcoin [9], the first real-world success of its kind, has laid the foundation for subsequent decentralized cryptocurrencies in part through its innovative solution to double-spending. The solution, known as Nakamoto consensus, provides a high-probability guarantee that an attacker cannot undo a transaction once it is sufficiently deep in the blockchain, assuming the attacker does not control too large a fraction of the computational resources in the network.

The idea behind Nakamoto consensus is as follows. Participants in the Bitcoin network, known as miners, compete to solve a computationally expensive proof-of-work puzzle. A miner who solves this puzzle is permitted to add a block of newly confirmed transactions to the blockchain, a distributed public ledger serializing all transactions ever made. By doing so, the miner is rewarded with a *block reward* (i.e., newly minted bitcoins) and *transaction fees*—determined by the payers at the time of the transaction—for all the transactions that were incorporated in the block by the miner. The new block and its proof-of-work are then broadcast to the network; upon verifying the block, miners will add the block to their corresponding blockchains and continue the mining process on their updated ledgers. Since mining is performed concurrently, it may be the case

* Work done in part while at the University of Maryland.

that conflicting versions of the blockchain, known as branches or forks, may form. In the prescribed protocol, miners resolve this by mining on the longest branch, breaking ties arbitrarily. Note that shorter branches are thus "orphaned," and any transactions in those branches are invalidated.

This opens up the possibility of double-spending if an attacker's transaction ends up on a branch that is orphaned (in which case the attacker can re-spend the coins used in that transaction somewhere else). And, indeed, short branches are orphaned frequently. For this reason, payees will generally not accept a transaction until it is incorporated in a block that is several layers (typically, 6) deep in the blockchain. Recent analysis [4] shows—under the assumption that the attacker controls a bounded fraction of the computational power in the network—that the probability that a branch is orphaned decreases exponentially in its length.

The analysis cited above, however, assumes that all parties other than the attacker continue to follow the protocol as prescribed. It does not take into account that other parties may be *rational*, and thus may deviate from the protocol when it is in their best interests to do so. It also does not consider the possibility that an attacker might actively try to incentivize other parties to deviate from the protocol in an attempt to increase its own profit.

**Our contributions.** We consider an attacker, henceforth referred to as Alice, who controls a minority of the computational power in the network but nevertheless attempts to carry out a double-spending attack against a vendor, henceforth referred to as Bob. To do so, Alice first performs a regular transaction $T$ with Bob, and waits for that transaction to be several layers deep in the blockchain (so that Bob sends Alice whatever goods were purchased by Alice in that transaction). Then, Alice attempts to undo transaction $T$ by inducing a fork in the chain at a point before $T$ was confirmed. The fork initiated by Alice will be shorter than the main fork that includes $T$, but Alice will try to "bribe" other (rational) miners to mine on the fork by issuing a *whale transaction* (i.e., a transaction with an anomalously large transaction fee) that is only valid on the forked branch. (The terminology "whale transaction" is inspired by a possibly erroneous bitcoin transaction[1] in which the payer issued a transaction carrying an exorbitant transaction fee of 291 bitcoins. The current recommended transaction fee for a no-delay transaction is $6.0 \times 10^{-7}$ bitcoins per byte, so at the current median transaction size of 257 bytes this would amount to a transaction fee of only $1.542 \times 10^{-4}$ bitcoins.[2]) Alice will of course also mine on the fork, and if she can incentivize sufficiently many other miners to also mine on the fork via this *whale attack*, then eventually the fork will overtake the main branch and transaction $T$ will be undone.

While the above description conveys the main idea, there are several additional ways Alice can gain further advantage. For example, Alice may begin building her fork before transaction $T$ is sufficiently deep in the main blockchain, keeping the fork private until $T$ is confirmed by Bob. Alice can also make sure

[1] cc455ae816e6cdafdb58d54e35d4f46d860047458eacf1c7405dc634631c570d

[2] Based on `https://bitcoinfees.21.co`, accessed September 1, 2016.

to incorporate a double-spend transaction on the fork itself—conveniently, by spending the coins to another address she owns—so that she immediately recovers her money once the fork overtakes the main chain.

Note that unless Alice can incentivize other miners to mine on her fork, Alice cannot hope for her fork to overtake the main branch. (Recall we assume that Alice does not have a significant fraction of the computational power in the network.) But rational miners will mine on Alice's fork if it is more profitable for them to do so than to mine on the main branch. In making the decision about which branch to mine on, miners must trade off the probability of having their mined block be on the (eventual) main branch—which is higher when mining on the (current) main branch—against the total fee they obtain by mining a new block—which is larger on the fork. The question then becomes: how large of a whale transaction does Alice need to issue in order to convince miners to switch to mining on her fork? Note further that if the whale-transaction fee is set too large, then the attack may not be profitable overall for Alice.

In summary, the contributions of this work are:

1. We introduce and formalize the whale attack, which demonstrates that rationality should be taken into account when evaluating the security of cryptocurrencies.
2. Via theoretical analysis, we establish bounds on the expected cost to Alice of carrying out the attack.
3. We simulate the attack in order to demonstate its feasibility even when the attacker wields a modest amount of mining power and capital.

## 1.1 Related Work

There is a growing body of work examining rational behavior in Bitcoin. Of particular relevance to our analysis is the work on *selfish-mining attacks*, initiated by Eyal and Sirer [3] and explored also in [12, 10], which shows how an attacker can profitably deviate by witholding blocks it has mined in an effort to mine additional blocks on top of those it has already mined and thus obtain more block rewards for itself.

More closely related to this work from a conceptual point of view are a class of attacks we call *bribery attacks* in which an attacker attempts to influence other miners to mine on a fork prefered by the attacker. Bonneau [1] presents various bribery attacks, including paying miners out-of-band to mine on a chosen branch. Teutsch et al. [14] present a different form of bribery attack in which an attacker issues spurious puzzles with their own payoffs (implemented using smart contracts such as Ethereum) in an attempt to draw computational power away from the Bitcoin network and thus increase the attacker's proportion of computational power. Our attack differs from these in that it relies on the Bitcoin protocol itself as a means of payment, and does not rely on any external mechanisms. Bonneau [1] also discusses a bribery attack in which the attacker includes on its desired branch transactions paying certain miners; our attack, however, incentivizes miners collectively to mine on a chosen branch rather than singling out specific miners to mine on the attacker's branch.

There is also a growing interest in understanding the relationship between transaction fees and Bitcoin's long-term health. Möser and Böhme [8] perform a longitudinal analysis of transaction fees and examine the externalities that influence them. Kroll et al. [7], Houy [5], and Kaşkaloğlu [6] consider the economics of Bitcoin mining and discuss potential changes to transaction fees and their policies in the long-term. Carlsten et al. [2] study the effect of diminishing block rewards and the concomitant rise of transactions fees as the dominant form of payoff for mining new blocks, and their implications for the selfish-mining attack.

## 2 Model

We adapt the model used by Rosenfeld [11] and Sompolinsky and Zohar [13]. We assume that the distribution of mining power in the network remains constant, and that an attacker Alice controls a fraction $\alpha < 0.5$ of the mining power. The remaining network consists of $k$ mining entities each controlling a $\beta_i$ fraction of the mining power. We let $\beta = \sum_i^k \beta_i = 1 - \alpha$.

Miners mine blocks according to a Poisson process with rate $\lambda$, which we also assume remains constant. Further, we assume that the propagation of new blocks to the network is instantaneous. Thus, we view time as being marked by block-creation events on either the main branch or Alice's branch. The total payoff for mining a block on the main branch (including both the block reward and average transaction fees) is normalized to 1; the reward for mining a block on Alice's branch is $\delta + 1$, where $\delta$ is the excess transaction fee in the whale transaction initiated by Alice. Throughout the rest of this paper, when we refer to the transaction fee of a whale transaction, we are referring to this parameter $\delta$.

Following each block-creation event, each mining entity (including Alice) chooses a rational strategy that will be followed until the next block-creation event. More specifically, Alice makes a rational decision for whether to continue the attack or reset the attack. Similarly, once whale transactions are underway, each mining entity $i$ makes a rational decision whether to continue mining on the public (longest) branch ("honest mining") or whether to mine on Alice's fork ("whale mining"). At this point, the remaining fraction of mining power $\beta$ can be divided into two partitions: whale miners and honest miners. Say $\gamma_i = 1$ if miner $i$ decides to whale mine, and $\gamma_i = 0$ otherwise. Then a fraction $q = \alpha + \sum_{i=1}^k \gamma_i \cdot \beta_i$ of the total mining power in the network is devoted to whale mining, and a fraction $p = \sum_{i=1}^k (1 - \gamma_i) \cdot \beta_i = 1 - q$ is devoted to honest mining.

The whale attack is carried out in two phases: a *pre-mining* phase and a *race* phase. We describe the attack somewhat informally here; it is specified fully in Appendix A.

**Pre-mining phase.** In this phase, Alice first tries to build a private branch that is longer than the public blockchain. Once she has done so, she issues a transaction $tx_B$ paying some entity Bob and waits for that transaction to be incorporated in the public blockchain $n$ levels deep, all the while continuing to mine on her private branch. Once $tx_B$ is $n$ levels deep, that transaction is "confirmed" and Alice then broadcasts her private branch—which we refer to now as *Alice's fork*—along with a second transaction $tx_A$ that pays out from the same address as $tx_B$ and so is a double-spending attempt; the attack then transitions to the race phase. (For simplicity, we assume that $tx_A$ pays out to another address that Alice owns.) Note that $tx_A$ is not valid if it is incorporated on the main branch, and so Alice's hope is that it will be incorporated on her fork and that her fork will then overtake the public chain. To incentivize other miners to mine on her fork, Alice ensures that $tx_A$ has a large transaction fee.

In more detail, fix a parameter $\ell$ with $1 \leq \ell \leq n+1$. The attack begins by having Alice try to mine a private branch (as in [3, 13]) that is $\ell$ blocks longer than the public branch. Once she does so, she issues transaction $tx_B$, and other miners work to incorporate it on the public branch. Meanwhile, Alice continues to mine on her fork while waiting for $tx_B$ to reach $n$ confirmations. Let $m$ denote the number of additional blocks mined by Alice when $tx_B$ reaches $n$ confirmations. The probability $P(m)$ for a given value of $m$ is [11, Section 4]:

$$P(m) = \binom{m+n-1}{m} \alpha^m \beta^n. \tag{1}$$

Once $tx_B$ is confirmed, Alice publishes $tx_A$ and her heretofore private branch containing $m+\ell$ blocks. If $m+\ell \leq n$ (in other words, Alice's branch is not longer than the main branch), the attack transitions to the race phase.

Note there is a tradeoff here: a larger value of $\ell$ will require more time before Alice obtains a lead of $\ell$ blocks over the public branch, but ensures a longer fork when Alice releases her double-spent transaction.

**Race phase.** In this phase, Alice's fork and the main branch enter into a race. Alice will continue to mine on her fork, and in addition will issue whale transactions that are not valid on the public chain but are valid on Alice's fork.[3] We assume for simplicity that Alice continues to issue whale transactions as her fork grows.

The race phase can be modeled as a biased random walk. The initial state is $z = n-(m+\ell)$, where this denotes the lead of the original public branch. (If $z < 0$ then Alice's fork is already longer, and so we assume $z \geq 0$ in what follows.) In each block-creation step, $z$ increases by 1 with some probability $p$ and decreases by 1 with probability $q = 1-p$, where $p$ and $q$ denote the relative fractions of mining power devoted to the original public branch and Alice's fork, respectively. In accordance with Rosenfeld's analysis [11, Section 3], the probability that $z$ eventually reaches the state $-1$ (in other words, the probability that Alice's fork eventually overtakes the original public branch) is

$$a_z = \min(q/p, 1)^{z+1}. \tag{2}$$

Of course, if $z$ ever becomes too large then Alice may simply abort the attack.

[3] This can be achieved in several ways, e.g., via a sequence of transactions that can all be traced back to the payer address used in $tx_B$.

While it would be interesting to analyze the expected profit Alice obtains in general, we are more interested in determining the regime where the attack succeeds with probability close to 1. This allows us to determine if a whale attack is potentially feasible, without having to make any assumptions about Alice's risk tolerance.

## 3 Analysis

In this section we establish informal bounds on the expected cost to carry out the whale attack with success probability close to 1. Since Alice's profit is contingent on the value of the double-spend being greater than the sum of the whale transactions, the main questions we are trying to answer are "How large do the whale transactions need to be?" and "How many whale transactions are needed?"

### 3.1 Simplifications and Assumptions

The attack described in the previous section is simplified in several ways, and is not presented in full generality. Our analysis also relies on various simplifying assumptions. We now briefly mention and justify them:

1. *We assume the relative power of all miners is known.*
2. *Mining entities consider only their own mining power and Alice's mining power when making rational decisions.* We make minimal assumptions about the sophistication of mining entities in evaluating their profits. We simply assume that each mining entity considers its own mining power and Alice's mining power, and evaluates whether it makes sense to "defect" and mine on Alice's fork under the assumption that the remaining miners continue to mine on the public branch. (I.e., we simply evaluate whether the status quo of Alice mining on her fork and the other miners mining on the public branch is a Nash equilibrium for each of the other miners. We do not examine a possible "cascading" effect which might make it profitable for miner $j$ to defect given that miner $i$ defects, etc.) This serves to establish an upper bound on the cost of the attack, since it underestimates the fraction of miners who might switch to mine on Alice's fork.
3. *Mining entities are not "sticky."* We allow miners to change their strategy after each block-creation event. We do not assume any cost due to "stickiness" for switching strategies.
4. *Mining entities will choose the (even marginally) most profitable mining strategy.*
5. *Whale mining power is kept constant throughout the race phase.* We assume that in the course of Alice's attack, she adjusts the excess transaction fee $\delta$ so as to keep the mining power on her fork constant. This is in contrast to other strategies, such as keeping $\delta$ fixed throughout the race phase. The strategy we choose to analyze allows Alice to better predict the number of blocks it will take for her attack to succeed, since the race phase can then be modeled as a steady-state stochastic process.

6. *Alice issues whale transactions for every block on her fork until the attack succeeds.* Since mining entities always make new rational decisions following each block event, Alice issues whale transactions until her branch is longer than the original branch. This implies that she never aborts[4] the attack and that her budget is unbounded.

Although some of the above assumptions may not hold, we believe that in most cases they are conservative and have the effect of increasing the cost of the attack. Nevertheless, in Section 4 we discuss open questions related to relaxing the above assumptions.

### 3.2 How Large Should Whale Transactions Be?

The first step in evaluating the cost of the whale attack is to determine what values $\delta$ are appropriate for incentivizing a desired proportion of the network to whale mine. To do this, we examine the decision faced by a rational miner.

Suppose miner $m$ has mining power $\beta_m$ and must decide whether to whale mine or not, under the assumption that the remaining miners (except Alice) will mine on the public branch as dictated by the protocol.[5] If $m$ chooses to continue mining on the public branch (i.e., $\gamma_m = 0$), then $m$ receives block rewards only if the whale attack fails. From Equation 2, the probability that the whale attack fails is $1 - a_z = 1 - \min(q/p, 1)^{z+1}$, where $q = \alpha$ and $p = \beta$ here because only Alice will mine on her fork. Conditioned on the whale attack failing, $m$ receives the next block rewards (which include both the reward for mining a new block as well as any transaction fees) with probability $\beta_m/\beta$. It follows that $m$'s expected profit is given by

$$\frac{(1 - a_z) \cdot \beta_m}{p} = \frac{\left(1 - \left(\frac{\alpha}{\beta}\right)^{z+1}\right) \cdot \beta_m}{\beta}. \tag{3}$$

(Recall we assume $\alpha < \beta$.)

On the other hand, suppose $m$ decides to whale mine (i.e., $\gamma_m = 1$). This means that $m$ receives block rewards only if the whale attack succeeds. The probability that the whale attack succeeds is $a_z = \min(q/p, 1)^{z+1}$, where $q = \alpha + \beta_m$ and $p = 1 - q = \beta - \beta_m$. Conditioned on the whale attack succeeding, $m$ receives block rewards with probability $\beta_m/q$. It follows that $m$'s expected profit

[4] It is worth remarking that even if Alice aborts the attack, the only "costs" to Alice are the block rewards she gave up by not mining on the public branch; the whale transaction fees are not paid if Alice's fork never overtakes the public branch.

[5] In fact, this ignores the case where the public branch and Alice's fork have the same length, and the protocol allows miners to choose arbitrarily which branch to mine on. We also ignore here the case where Alice's fork is longer than the public branch, and so the whale attack succeeds immediately. Taking these into account will only increase the probability that the attack succeeds.

when whale mining is given by

$$\frac{a_z \cdot \beta_m}{q} \cdot (\delta + 1) = \frac{\left(\frac{\alpha+\beta_m}{\beta-\beta_m}\right)^{z+1} \cdot \beta_m}{\alpha + \beta_m} \cdot (\delta + 1). \tag{4}$$

(Recall that the normal block reward is normalized to 1, and the whale block reward is $\delta + 1$.)

Miner $m$ will choose $\gamma_m \in \{0, 1\}$ that maximizes its expected profit, i.e., it will whale mine when Equation 4 is greater than Equation 3. Solving for $\delta$ for wihch this holds, we see that whale mining is profitable for miner $m$ when

$$\delta > f(\alpha, \beta_m) \overset{\text{def}}{=} \frac{\left(1 - \left(\frac{\alpha}{\beta}\right)^{z+1}\right)}{\beta} \cdot \frac{\alpha + \beta_m}{\left(\frac{\alpha+\beta_m}{\beta-\beta_m}\right)^{z+1}} - 1, \tag{5}$$

which is equivalent to

$$\delta > \frac{\Pr[\text{whale attack fails} \mid \gamma_m = 0]}{\Pr[\text{honest block} \mid \gamma_m = 0]} \cdot \frac{\Pr[\text{whale block} \mid \gamma_m = 1]}{\Pr[\text{whale attack succeeds} \mid \gamma_m = 1]} - 1.$$

Below we tabulate minimum values of $\delta$ that make whale mining more profitable than honest mining, as a function of the lead $z$ of the original branch, Alice's fractional mining power $\alpha$ (rows), and $m$'s fractional mining power $\beta_m$ (columns).

$z = 6$

| | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 |
|---|---|---|---|---|---|---|---|
| 0.20 | 682.40 | 140.20 | 32.33 | 7.54 | 1.29 | 0 | 0 |
| 0.25 | 149.56 | 34.54 | 8.11 | 1.44 | 0 | 0 | 0 |
| 0.30 | 37.00 | 8.74 | 1.61 | 0 | 0 | 0 | 0 |
| 0.35 | 9.38 | 1.78 | 0 | 0 | 0 | 0 | 0 |
| 0.40 | 1.88 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.45 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$z = 5$

| | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 |
|---|---|---|---|---|---|---|---|
| 0.15 | 962.74 | 213.41 | 55.96 | 15.89 | 4.36 | 0.76 | 0 |
| 0.20 | 226.76 | 59.50 | 16.95 | 4.69 | 0.87 | 0 | 0 |
| 0.25 | 63.47 | 18.12 | 5.07 | 1.00 | 0 | 0 | 0 |
| 0.30 | 19.39 | 5.47 | 1.13 | 0 | 0 | 0 | 0 |
| 0.35 | 5.84 | 1.25 | 0 | 0 | 0 | 0 | 0 |
| 0.40 | 1.28 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.45 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$z = 4$

| | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 |
|---|---|---|---|---|---|---|---|

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0.15 | 239.90 | 70.46 | 23.41 | 8.09 | 2.57 | 0.44 | 0 |
| 0.20 | 74.86 | 24.91 | 8.66 | 2.79 | 0.53 | 0 | 0 |
| 0.25 | 26.55 | 9.27 | 3.03 | 0.63 | 0 | 0 | 0 |
| 0.30 | 9.89 | 3.28 | 0.73 | 0 | 0 | 0 | 0 |
| 0.35 | 3.46 | 0.80 | 0 | 0 | 0 | 0 | 0 |
| 0.40 | 0.78 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.45 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$z = 3$

| | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 |
|---|---|---|---|---|---|---|---|
| 0.10 | 170.83 | 55.88 | 21.50 | 8.88 | 3.63 | 1.25 | 0.12 |
| 0.15 | 59.18 | 22.80 | 9.45 | 3.89 | 1.38 | 0.18 | 0 |
| 0.20 | 24.21 | 10.07 | 4.18 | 1.52 | 0.25 | 0 | 0 |
| 0.25 | 10.71 | 4.48 | 1.67 | 0.32 | 0 | 0 | 0 |
| 0.30 | 4.75 | 1.80 | 0.39 | 0 | 0 | 0 | 0 |
| 0.35 | 1.85 | 0.42 | 0 | 0 | 0 | 0 | 0 |
| 0.40 | 0.34 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.45 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$z = 2$

| | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 |
|---|---|---|---|---|---|---|---|
| 0.05 | 75.72 | 27.73 | 12.47 | 6.10 | 3.01 | 1.36 | 0.42 |
| 0.10 | 29.29 | 13.20 | 6.49 | 3.23 | 1.49 | 0.50 | 0 |
| 0.15 | 13.98 | 6.90 | 3.46 | 1.62 | 0.58 | 0 | 0 |
| 0.20 | 7.31 | 3.69 | 1.76 | 0.66 | 0.01 | 0 | 0 |
| 0.25 | 3.89 | 1.88 | 0.73 | 0.05 | 0 | 0 | 0 |
| 0.30 | 1.95 | 0.78 | 0.08 | 0 | 0 | 0 | 0 |
| 0.35 | 0.75 | 0.07 | 0 | 0 | 0 | 0 | 0 |
| 0.40 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$z = 1$

| | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 |
|---|---|---|---|---|---|---|---|
| 0.05 | 7.50 | 4.06 | 2.36 | 1.36 | 0.71 | 0.27 | 0 |
| 0.10 | 4.29 | 2.51 | 1.47 | 0.79 | 0.32 | 0 | 0 |
| 0.15 | 2.65 | 1.56 | 0.86 | 0.38 | 0.03 | 0 | 0 |
| 0.20 | 1.64 | 0.91 | 0.41 | 0.05 | 0 | 0 | 0 |
| 0.25 | 0.94 | 0.43 | 0.07 | 0 | 0 | 0 | 0 |
| 0.30 | 0.41 | 0.05 | 0 | 0 | 0 | 0 | 0 |
| 0.35 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Notice that more powerful miners are more easily enticed to whale mine. More interestingly, we observe that for certain parameters a miner $m$ may prefer to whale mine *even if Alice does not issue any whale transactions* (and even when $\alpha + \beta_m < 0.5$)! We are not aware of this observation being made before. Intuitively, the reason for this is that miner $m$ earns a larger fraction of the rewards (because it can mine a larger fraction of the blocks) when mining on Alice's fork, plus there is some probability that blocks mined on the main branch will become worthless. Thus, as Alice's fork becomes more likely to overtake the main branch, the expected payoff to miner $m$ increases and in some cases exceeds the expected payoff of mining on the main branch even in the absence of whale transactions.

Moreover, if we differentiate $f(\alpha, \beta_m)$ (cf. Equation 5) with respect to $\alpha$, we see that $f$ is strictly decreasing for $\alpha \in [0, 0.5)$. As expected, this means that increasing Alice's mining power $\alpha$ will decrease the required $\delta$ and so will increase the effectiveness of the attack. Differentiating with respect to $\beta_m$, we see that $f$ is strictly decreasing for $\beta_m \in [0, 0.5)$. This means that if whale mining is profitable for a miner $m$, then it is also profitable for all miners who are at least as powerful. Although we have analyzed the decision of miner $m$ under the assumption that other miners continue to mine on the public branch, these observations actually show that the required $\delta$ is (possibly) even smaller than indicated: miner $m$ can reason that all other miners that are at least as powerful as itself will also switch to mining on Alice's branch, which (from the point of view of miner $m$) effectively increases $\alpha$, which in turn lowers the required $\delta$. Determining the final equilibrium is an interesting open question.

## 3.3 How Many Whale Transactions are Needed?

We have so far analyzed whether Alice can make it more profitable for other miners to deviate by carrying out a whale attack, but have not yet analyzed whether carrying out the attack is profitable for Alice! To do so, we need in particular to evaluate Alice's net payout, i.e., how many whale transactions she needs to issue in order to successfully complete the attack.

To recap, the initial state in the race phase is the lead $z$ of the original public branch over Alice's branch; $z$ decreases by 1 with probability $q$, the proportion of power devoted to mining on Alice's fork, and increases by 1 with probability $p = 1 - q$, the proportion of power devoted to mining on the public branch. (Recall we assume that Alice will keep the whale mining power constant by adjusting $\delta$ as necessary, and that she will continue doing so until her branch overtakes the original branch.) The race phase is thus analagous to the Gambler's Ruin problem. Using elementary theory of random walks, one can show that the expected number of block-creation events before reaching $z = -1$ is given by $E \overset{\text{def}}{=} \frac{z+1}{2q-1}$ assuming $q > 0.5$. Now, say $E$ block-creation events have occured when Alice's fork overtakes the public branch, with $E_A$ of those on Alice's fork and $E' = E - E_A$ of those on the public branch. Since $E_A = E' + z + 1$, we see that the expected number of block-creation events on Alice's branch (and hence the expected number of whale transactions issued by Alice) is

$$E_A = \frac{E + z + 1}{2} = \frac{z+1}{4q-2} + \frac{z+1}{2} .$$

Now that we have established bounds on appropriate values for whale transaction fees and have calculated the expected number of whale transactions, we can in principle analyze how much the whale attack costs. This is not trivial to do analytically, since (by our assumption) the whale-transaction fee always changes to maintain a fixed fraction of mining power on Alice's fork. For that reason, we determine the cost of the attack by simulation.

### 3.4 Simulation

| AntPool | F2Pool | BTCC Pool | BW.COM | BitFury |
|---|---|---|---|---|
| 18.8% | 18.2% | 16.2% | 13.8% | 9.4% |

| HaoBTC | SlushPool | ViaBTC | BitClub Net | Kano CKPool |
|---|---|---|---|---|
| 6.4% | 5.9% | 4.4% | 3.7% | 3.1% |

**Table 2:** Distribution of mining power among the ten largest pools (95% of the total) from July 30–August 2, 2016 (source: `https://blockchain.info/pools`).

We use as our distribution of mining power the distribution among the ten largest pools in the Bitcoin network (cf. Table 2), where we equate Alice with the largest of those pools, so $\alpha = 0.188$. Since we are interested in the expected cost to carry out the attack with success probability 1, we only consider cases in which the whale mining power $q > 0.5$. For example, we run simulations in which $\delta$ is set so that all pools at least as large as the BTCC Pool will whale mine, i.e., $q = 0.188 + 0.162 + 0.162 = 0.532$. Table 3 presents the cost of the whale attack in terms of $\delta$ under different parameters of $q$ and $z$. Our simulation takes into account the fact that with probability $\frac{\alpha}{q}$, Alice reclaims her whale transaction fee.

| $q$ | z = 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0.532 | 2.93e+23 | 3.09e+22 | 8.03e+21 | 1.10e+22 | 2.57e+24 | 2.50e+21 | 4.40e+20 |
| 0.670 | 999.79 | 464.74 | 307.71 | 267.72 | 56.09 | 17.64 | 3.63 |
| 0.764 | 768.09 | 291.86 | 109.89 | 40.16 | 12.73 | 2.48 | 0 |
| 0.828 | 1265.14 | 417.85 | 135.80 | 42.32 | 11.60 | 1.65 | 0 |
| 0.887 | 1205.00 | 390.63 | 123.93 | 37.23 | 9.46 | 1.00 | 0 |
| 0.931 | 1806.67 | 540.75 | 159.34 | 44.66 | 10.69 | 1.12 | 0 |
| 0.968 | 2178.58 | 628.13 | 178.19 | 48.29 | 11.23 | 1.15 | 0 |
| 0.999 | 2598.64 | 723.92 | 198.92 | 52.33 | 11.89 | 1.22 | 0 |

**Table 3:** Simulated attack cost (sum of the whale-transaction fees) for different settings of the whale mining power $q$ and the lead of the public branch at the start of the race phase $z$. Values shown are averages across $10^6$ simulations for each pair of $q$ and $z$.

To interpret our results, recall that $\delta$ is a lower bound on the value of whale transaction fees for whale mining to be more profitable for other miners. Thus, the cost of the attack is marginally more than the sum of the whale transactions in our simulations. Finally, recognize that the table does not give the complete picture with regard to rationality of the attack from the point of view of Alice, as we need to evaluate whether Alice's net gain is greater than she would have achieved by following the protocol honestly. On the other hand, note that when the whale attak is successful, Alice reaps all the block rewards from her $m + \ell$ pre-mined blocks.

## 4 Discussion

Our simulations return a number of interesting results. Immediately, we can see the impact that pre-mining has on the cost of the whale attack. While pre-mining $\ell$ blocks may take a long time, the time required can be improved if Alice employs selfish mining strategies [13].

Our results also show that centralization of mining power increases the incentive for other miners to be influenced by a whale attack since, as shown in Section 3.2, larger pools are more easily bribed than smaller pools. In our simulation, the three largest pools (which include Alice) already combine for a majority of whale mining power. Since $q = 0.532$ is only slightly above a majority, the cost of the attack is exorbitant. However, simply adding the fourth largest pool to give $q = 0.670$ dramatically reduces the cost of the attack. From Table 3 for $z = 6$, we see that Alice's cheapest option is to aim for $q = 0.764$. Attempting to bribe the smaller pools, which would allow $z$ to converge faster, would not be cost efficient. Now, consider if mining were completely decentralized, and the largest pools wielded less than 0.01 of the mining power—the whale attack would then be incredibly costly for Alice to carry out.

Finally, consider that Alice only wields $\alpha = 0.188$ of the mining power in our simulations. In the past, mining pools have occasionally enjoyed much larger shares of mining power, even exceeding a majority on some occasions. A larger attacker could dramatically reduce the cost of the attack. Taking this a step further, our assumptions from Section 3 already induce an upper bound on the cost. We address these assumptions below, and discuss how they might fail to hold in practice.

- A more sophisticated mining entity who considers the decisions of other mining entities could dramatically lower the necessary $\delta$ for whale mining to be rational. In practice, cooperative mining entities would achieve similar effects, since they could certainly account for each other's mining power when evaluating their collective profit.
- In practice, if a mining entity mines a large whale block, it would likely be in its best interest to "stick" to whale mining. It may even be rational for them to issue their own whale transactions to ensure the success of Alice's branch. From Alice's perspective, the best case (other than if she were to reclaim every whale transaction) would be to have different mining entities each mine a single whale block. If these entities combine for a majority of whale mining power, it is probable that further whale transactions would not be needed at all.

- In practice, a marginal profit for whale mining over honest mining may not be sufficient, and we would need to consider the "cost of deviation."
- In practice, it is not necessary for Alice to keep whale mining power constant, especially if Alice does not require the whale attack to succeed with probability 1.
- Alice may choose to abort the attack if her fork falls too far behind the public branch, since continuing the whale attack would not be rational if the attack unluckily takes longer than expected.

## 5 Conclusion

Cryptocurrencies fail to fit into established theoretical frameworks for secure distributed systems. Instead, their security relies on the assumption that a majority of miners, as measured by their computational resources, will behave honestly. In this regard, researchers have uncovered many deviant mining strategies that rational miners may employ.

In this work, we presented the whale attack, in which a minority attacker increases her chances of double-spending by incentivizing rational miners mining on her fork. Our work is primarily a proof-of-concept showing that a whale attack can be feasible for a minority attacker. We leave open the challenges of modeling the cost of the attack more precisely and exploring the strategy space when combining the whale attack with other mining attacks.

## Acknowledgments

We thank Elijah Soriah and Andrew Miller for their valuable feedback, and the faculty and students of the CAAR REU program for the wonderful experience. Work of Kevin Liao was supported by NSF Research Experience for Undergraduates (REU) award CNS-1560193. Work of Jonathan Katz was supported in part by NSF award CNS-1561209.

## References

1. Joseph Bonneau. Why buy when you can rent? Bribery attacks on Bitcoin-style consensus. In *Financial Cryptography and Data Security Workshops (BITCOIN, VOTING, and WAHC)*, volume 9604 of *Lecture Notes in Computer Science*, pages 19–26. Springer, 2016.
2. Miles Carlsten, Harry Kalodner, S. Matthew Weinberg, and Arvind Narayanan. On the instability of Bitcoin without the block reward. In *ACM Conference on Computer and Communications Security*, pages 154–167. ACM, 2016.
3. Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography and Data Security*, Lecture Notes in Computer Science, pages 436–454. Springer, 2014.

4. Juan Garay, Aggelos Kiayias, and Nikos Leonardos. The Bitcoin backbone protocol: Analysis and applications. In *Advances in Cryptology—Eurocrypt 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 281–310. Springer, 2015.
5. Nicolas Houy. The economics of Bitcoin transaction fees, 2014. GATE Working Paper 1407, available at `http://dx.doi.org/10.2139/ssrn.2400519`.
6. Kerem Kaskaloglu. Near zero Bitcoin transaction fees cannot last forever. In *Intl. Conference on Digital Security and Forensics (DigitalSec)*, pages 91–99. The Society of Digital Information and Wireless Communications, 2014.
7. Joshua A. Kroll, Ian C. Davey, and Edward W. Felten. The economics of Bitcoin mining, or Bitcoin in the presence of adversaries. In *12th Workshop on the Economics of Information Security (WEIS)*, 2013.
8. Malte Möser and Rainer Böhme. Trends, tips, tolls: A longitudinal study of Bitcoin transaction fees. In *Financial Cryptography and Data Security Workshops (BITCOIN, WAHC, and Wearable)*, volume 8976 of *Lecture Notes in Computer Science*, pages 19–33. Springer, 2015.
9. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.
10. Kartik Nayak, Srijan Kumar, Andrew Miller, and Elaine Shi. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 305–320. IEEE, 2016.
11. Meni Rosenfeld. Analysis of hashrate-based double spending, 2014. Manuscript available at `https://arxiv.org/abs/1402.2009`.
12. Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. Optimal selfish mining strategies in Bitcoin. In *Financial Cryptography and Data Security*, Lecture Notes in Computer Science. Springer, 2016. Available at `https://arxiv.org/abs/1507.06183`.
13. Yonatan Sompolinsky and Aviv Zohar. Bitcoin's security model revisited, 2016. Manuscript available at `https://arxiv.org/abs/1605.09193`.
14. Jason Teutsch, Sanjay Jain, and Prateek Saxena. When cryptocurrencies mine their own business. In *Financial Cryptography and Data Security*, Lecture Notes in Computer Science. Springer, 2016.

## A Whale Attack Algorithm

**Algorithm 1** Whale attack

```
1: procedure Reset
2:     $original\_branch \leftarrow$ longest branch
3:     $Alice\_branch \leftarrow$ longest branch
4:     $l\_count \leftarrow 0$                          ▷ $len(Alice\_branch) - len(original\_branch)$.
5:     Issue $tx_A$ on $Alice\_branch$.
6:     Mine at head of $Alice\_branch$.

7: procedure Pre-mine($l$, $n$)
8:     Reset
9:     while $l\_count < l$ do
10:        $new\_block \leftarrow$ Listen                ▷ Listen for block creation event.
11:        if $new\_block$ on $Alice\_branch$ then
12:            $l\_count \leftarrow l\_count + 1$
13:        else if $l\_count = 0$ then             ▷ $len(Alice\_branch) < len(original\_branch)$.
14:            Reset
15:        else                                    ▷ $len(Alice\_branch) \geq len(original\_branch)$.
16:            $l\_count \leftarrow l\_count - 1$
17:    Issue $tx_B$ on $original\_branch$.
18:    $n\_count \leftarrow 0$
19:    $m \leftarrow 0$

20:    while $n\_count < n$ do
21:        $new\_block \leftarrow$ Listen
22:        if $new\_block$ on $Alice\_branch$ then
23:            $m \leftarrow m + 1$
24:        else
25:            $n\_count \leftarrow n\_count + 1$
26:    Publish $Alice\_branch$.
27:    if $m + l \leq n$ then                      ▷ $len(Alice\_branch) \leq len(original\_branch)$.
28:        Race $(n - (m + l))$

29: procedure Race($z$)
30:    Issue $tx$ on $Alice\_branch$.
31:    while $z > -1$ do
32:        $new\_block \leftarrow$ Listen
33:        if $new\_block$ on $Alice\_branch$ then
34:            $z \leftarrow z - 1$
35:            Issue $tx$ on $Alice\_branch$.
36:        else
37:            $z \leftarrow z + 1$
```