Sucker punch makes you richer

Rethinking Proof-of-Work security model

Runchao Han*
Monash University and CSIRO-Data61
runchao.han@monash.edu

Zhimei Sui*
East China Normal University
zhimeisui@gmail.com

Jiangshan Yu
Monash University
jiangshan.yu@monash.edu

Joseph Liu
Monash University
joseph.liu@monash.edu

Shiping Chen
CSIRO-Data61
shiping.chen@data61.csiro.au

Abstract

Honest majority is a vital assumption of Bitcoin-style blockchains. However, recent 51% attacks render this assumption unrealistic in practice. In this paper, we analyse two possible 51% attacks launched by a rational miner, who is profit driven. The first attack considers a rational miner, who moves his mining power from a stronger blockchain to a weaker blockchain to launch 51% attacks, provided that (1) the mining power is compatible in both blockchains, and (2) the transferred mining power dominates the weaker blockchain. We say a blockchain is stronger if the total mining power of this blockchain is higher than other (weaker) blockchains. The second attack considers a rational miner, who rents cloud mining power to launch 51% attacks. The former attack is new, and we name it *mining power migration attack*; the latter is called *cloud mining attack*, and it was initially covered by the bribery attack (FC' 16).

We formalise the two attacks by using Markov Decision Process. We then test the feasibility of launching both attacks on leading blockchains in the wild by using our model. We show that both attacks are feasible and profitable. For example, our result shows that with 12.5% mining power of Bitcoin, a rational miner can gain approximately 6% ($18,946.5 USD) extra profit than honest mining, by launching mining power migration attack to double spend a transaction of 3000 BCH (equivalent to $378,930) on BitcoinCash. We also investigate the 51% attack on Ethereum Classic happened in Jan. 2019, by applying our model into this attack to provide some insights to understand more about it.

1 Introduction

Proof-of-work (PoW) based consensus was first introduced by Bitcoin [29]. It allows distributed participants within the consensus agreeing on the same global state. For such cryptocurrencies, Blockchain is the most commonly used data structure for storing transactions. A blockchain records all transactions as a chain of blocks i.e. batched transactions. Anyone can create a block of transactions, and append it into the blockchain. PoW enforces one to solve a computationally hard "crypto-puzzle" to create a block. In particular, one needs to find a random nonce to make the block's hash value smaller than a target value. The target value is dynamically adjusted according to the consensus protocol. This guarantees that new blocks are appended with a stable rate. In PoW, "crypto-puzzle" is achieved by hash algorithms; solving such crypto-puzzles is also called "mining"; and participants who are mining are so called "miners".

Blockchain is exposed to the "fork" problem. Fork refers to the scenario that miners create multiple valid blocks with the same preceded block. In Bitcoin, miners select the longest branch within valid branches. However, the selected branch may be overtaken by the other branch, and all transactions in the overtaken branch will be deem invalid. This introduces the "double spending attack": One may first spend some coins in a branch, then creates another branch (e.g. longer branch in Bitcoin) to erase this transaction and make those spent coins spendable again. In PoW, an attacker should have a majority of mining power i.e. computational resource to create a branch longer than the current one. This is so called the "51% attack".

Security assumptions of PoW-based consensus. Due to the "51% attack", all PoW-based blockchains have to assume that the majority of mining power is honest i.e. behaving normally according to the protocol. This "honest majority" assumption is the key assumption to guarantee the security of all PoW-based blockchains. If this assumption does not hold, then the so called 51% attack would be possible, and an attacker is able to rewrite the blockchain and spend a coin more than once [29].

The "honest majority" is guaranteed relatively rather than absolutely: The difficulty to break the assumption directly relies on the total mining power in the system. The more mining power in the system, the more difficult to control 51% mining power, and the higher the security guarantee will be. To achieve stronger security guarantee, PoW should attract miners to increase its mining power. To attract miners, Bitcoin and other PoW-based blockchains adopt an incentive mechanism. In this incentive mechanism, miners contribute mining power to PoW, and miners who successfully created a block will get some cryptocurrencies as the reward of mining. The mining reward of a block consists of a pre-defined number of cryptocurrencies and transaction fees of transactions in this block.

"Honest majority" in the wild. Recall that the "honest majority" assumption is relative towards the system's mining power. If only one blockchain exists in the world, then all potential miners will contribute their mining power to this blockchain, and controlling 51% mining power could be extremely difficult. However, if multiple blockchains with the same hash algorithm exist simultaneously, the total mining power for this algorithm in the world will be split into these blockchains. This reduces the difficulty of controlling 51% mining power in both of the blockchains.

Unfortunately, the "multiple blockchains" scenario is the fact in the real world rather than the "single blockchain" scenario. Currently, there are over 2,000 cryptocurrencies [1], and many of them share the same hash algorithm. This weakens security guarantees of all PoW-based blockchains, as no single blockchain can take full advantage of its incentive mechanism. In this way, controlling the majority of mining power is no longer impossible for some PoW-based blockchains.

In addition to the "multiple blockchains" scenario, cloud mining service e.g. Nicehash [7] gives another way to rent great mining power in a short time. Bribery attacks [12] (a.k.a. cloud mining attack) considered the possibility of renting cloud mining power to gain a majority of mining power in PoW-based blockchains.

Since 2018, many successful 51% attacks have been identified, as shown in Figure 1. These attacks result in the loss of more than $23 million US dollars. One main stream suspicion [28, 26] is that the attacks are equipped with cloud mining power.

Miners are rational rather than honest. In 2005, before the birth of Bitcoin, BAR (Byzantine, Altruistic, Rational) model [10] was considered in the context of fault tolerance for cooperative services. In this model, a participant can be Byzantine, altruistic, or rational. A Byzantine (a.k.a. malicious) participate would try everything to break the security of the system with any cost; an altruistic (a.k.a. honest) participant will always follow

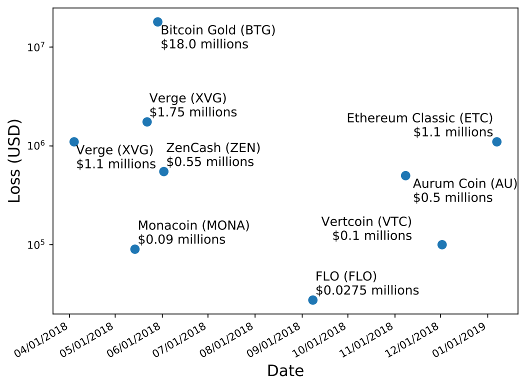


Figure 1: Successful 51% Attacks in 2018-2019 [8]. We omit the 51% attack on Litecoin Cash on June 4, 2018 as the loss is unknown.

the system specification; and a rational attacker would only depart from the protocol specification and being Byzantine if he can gain more profit. As the nature of the incentive mechanism in blockchains is to encourage miners joining the system with reward, so it is reasonable to assume that most miners are rational, rather than Byzantine or altruistic. For example, miners do not attack the blockchain because attacks are not profitable, rather than because miners want to be honest. Once miners can profit from attacking the blockchain, they will attack.

"Sucker punch" v.s. "honest majority" in the presence of rational miners. Given the facts that (1) multiple blockchains may share the same hash algorithm; (2) cloud mining service enables renting mining power temporarily; and (3) miners are rational, we suspect the security guarantee of PoW-based blockchains in the presence of rational miners. More specifically, we doubt the soundness of the "honest majority" assumption and the incentive compatibility in PoW. In this paper, we investigate a variant of 51% attacks which we call "sucker punch attack". In "sucker punch attack", the attack obtains temporary mining power, either from other blockchains with the same hash algorithm or the cloud mining services, to launch 51% attacks on a blockchain and double-spend some money. The "sucker punch attack" exploits our identified facts and directly endangers PoW's "honest majority" assumption. Furthermore, as a variant of 51% attacks, "sucker punch attack" can be profitable, so miners have incentive to launch such attacks. We name the attack using mining power from other blockchains "mining power migration attack", and the attack using cloud mining services "cloud mining attack". Here, the "mining power migration attack" is

[1]https://coinmarketcap.com/all/views/all/

identified by us, and the "cloud mining attack" is previously proposed as the "bribery attack".

We evaluate both the "mining power migration attack" and the "cloud mining attack" using Markov Decision Process (MDP), and surprisingly find that both attacks are practical and profitable on most mainstream cryptocurrencies. For example, we show in Section 5 that, a miner with 12.5% mining power of Bitcoin (BTC) can gain approximately 6% ($18,946.5 USD) extra profit (than honest mining) by double-spending a transaction of 3000 BCH (equivalent to $378,930) on BitcoinCash. This required mining power is in fact not difficult to achieve. At the time of writing (10/May/2019), the top two mining pools in BTC, i.e. BTC.com with 15.5% mining power and AntPool with 14.1% mining power, already achieves this requirement.

In addition, we simulate and analyse the 51% attack on Ethereum Classic (ETC) in Jan. 2019 [5] using our model in order to quantify the incentive and investigate the attacking strategy of the attacker. Our results show that the attacker's net revenue from the attack is close to the value ($100,000) that the attacker gave back to Gate.io. Moreover, we describe the attacker's behaviours during the attack, and these actions are proved to be smart strategies to maximise and stabilise his revenue.

Furthermore, we use our model to show how to eliminate the incentive of such "sucker punch attacks" by adjusting relevant blockchain parameters. We use the 51% attack on ETC as an example, and show that going back to Jan 2019, if cryptocurrency exchanges set the maximum transaction amount to 9,000 ETC (approximately $38,340), the attacker will not get any net revenue. If an exchange wants to allow higher transaction amounts, e.g. 52,800 ETC (approximately $224,928) which is the largest transactions in this attack, then our model recommends that the number of confirmation should be increased from 12 to 18.

Our contributions. Our contributions are summarised as follows:

1. We build new security foundation of PoW consensus reflecting to the real-world scenario.

2. Under our security model, we propose the "sucker punch attack", a practical variant of 51% attacks on PoW consensus.

3. We formalise the "sucker punch attack", including our identified "mining power migration attack" and the previously proposed "cloud mining attack", using our proposed SPA-MDP model.

4. We implement our SPA-MDP model in Python, and evaluate the impact of different PoW parameters (e.g. the transaction amount and the confirmation blocks) on the estimated cost and the relative revenue of an attack. This allows us to generate recommendations on how to setup the parameters for a given blockchain.

5. With our SPA-MDP model, we provide a feasibility study of the sucker punch attack on several mainstream blockchains from two aspects, namely the estimated cost and the potential net revenue of a successful attack. Our results show that both the "mining power migration attack" and the "cloud mining attack" are practical and profitable.

6. We investigate the 51% attack on Ethereum Classic (ETC) in Jan. 2019, by applying our SPA-MDP model into this attack. Our investigation recovers the attacker's revenue and provides some insights on the attack behaviours.

7. We discuss how to mitigate such sucker punch attacks. In particular, we use our SPA-MDP model to show the maximum transaction amounts and the minimum confirmation blocks that can fully mitigate the inventive of sucker punch attacks.

Paper organisation. This paper is structured as follows. Section 2 compares existing PoW security analysis with ours. Section 3 presents our system model and formalise the two sucker punch attacks (SPAs) in Markov Decision Process (MDP). In section 4, we implement our SPA-MDP model in Python, and evaluate the impact of different parameters on the two attacks. In section 5, we evaluate the security of selected blockchains based on our model, and investigate the recent 51% attack on EthereumClassic (ETC) in January 2019. In Section 6, we discuss potential remedies based on the observations we derived from our model, and using ETC as an example to show how to use our model to provide recommendations to a given blockchain. Section 7 concludes our work.

2 Existing work on PoW-based consensus security

Most research on PoW-based consensus security can be classified to the following four types: 1) the formalisations of PoW-based consensus that extract security assumptions and properties under well-defined security models; 2) the evaluation frameworks that define quantitative metrics from existing formalisations and analyse protocols using these metrics; 3) the attacks that indicate flaws in the protocol design; and 4) the models for understanding attacks using MDP or game theory.

Our research lies in the third type (i.e. proposing attacks), and in addition covers the last type (i.e. evaluating our proposed attacks using MDP). However, our research is different from all of them, starting from the security model. More specifically, we analyse PoW-based consensus in the presence of external mining power (from other blockchains or cloud mining service). The presence of external mining power reflects to the real world, but all existing research does not consider it.

As all security analyses begin with the security model, the claimed conclusions from research not covering our scenario may not work in our security model. For example, existing research assumes the "honest majority". Our analysis shows that this assumption does not hold anymore with external mining power due to the "sucker punch atatck". In addition, we evaluate such "sucker punch attack" using MDP. The evaluation results show that rational miners are also incentivised to launch sucker punch attacks apart from honest mining.

In this section, we review existing research on PoW-based consensus security, and compare them with ours.

2.1 Formalisations of PoW-based consensus

There are several attempts on formalising PoW-based consensus, based on different security models. Garay et al. [17] first formalised Bitcoin's consensus under the synchronous network setting, and first extracts two refined properties, namely the *common prefix* (i.e. the number of blocks from the end of their local chains that two honest parties disagree with) and the *chain quality* (i.e. the ratio of blocks in the blockchain contributed by adversarial miners). After that, Pass et al. [31] extended this formalisation to the bounded asynchronous network setting, then followed by Garay et al. [18] which further considered the dynamic difficulty adjustment of PoW. Following [18], Kiayias et al. [22] extracted another property called the *chain growth* (i.e. the number of blocks appended to the blockchain during any time unit). Zhang et al. [36] proposed a cross-protocol evaluation framework with three new properties for measuring PoW-based consensus' attack resistance, namely the *incentive compability* (i.e. the relative revenue lower bound of honest miners under selfish mining attacks), the *subversion gain* (i.e. the profit upper bound of a adversarial miner performing double spending), and the *censorship susceptibility* (i.e. the profit loss of honest miners under censorship retaliation attacks).

Considering the external mining power will influence the *chain quality*, the *chain growth*, and the three attack-resistance-related properties (i.e. the *incentive compability*, the *subversion gain* and the *censorship susceptibility*). More specifically, external mining power can reduce the *chain quality* because adversarial mining power enters the mining; temporarily speed up the *chain growth* because total mining power increases in a short time; weaken the three attack-resistance-related properties (i.e. reduce the *incentive compability* and increase the *subversion gain* and *censorship susceptibility*) because adversary becomes stronger with external mining power.

As our research focuses on the 51% attack, we mainly evaluate the *chain growth* (i.e. how much external mining power speeds up the growth of the adversary's fork for launching 51% attack) and the *subversion gain* (i.e. how much the adversarial miners can profit from our proposed "sucker punch attacks"). More specifically, our evaluation shows that the adversary can use external mining power to significantly increase the *chain growth* of its own fork faster than the honest fork and perform the "sucker punch attack" with significant *subversion gain* under relatively low requirements.

2.2 Evaluation frameworks on PoW-based consensus security

Gervais et al. [19] first proposed an evaluation framework for quantitatively analyse PoW-based consensus security using MDP, with a focus on the resistance of selfish mining and double-spending. Zhang et al. [36] generalised this framework by 1) formalising the selfish-mining-resistance as *incentive compability* and the double-spending-resistance as *subversion gain*; 2) considering the feather forking attack, generalising it as censorship retaliation attacks and formalising the censorship-retaliation-attack as *censorship susceptibility*; 3) considering the *chain quality* and measuring it using MDP.

We generalise the model from Gervais et al. [19] in order to measure the impact of external mining power on the *subversion gain* from Zhang et al. [36]. To our best knowledge, our research presents the first MDP-based model that can evaluate the 51% attack (quantitatively speaking, *subversion gain*) in the wild (where external mining power exists).

2.3 Proposed attacks on PoW-based consensus

The original Bitcoin whitepaper [29] briefly discussed the 51% attack and proved the "honest majority" security assumption by modelling Bitcoin blockchain growth as Poisson distribution.

However, several research proves that the "honest majority" cannot prevent all attacks. Miller et al. [27] first proposed an attack called "feather-fork" which does not require the adversary to control the majority of mining power. Eyal et al. [16] later proposed the selfish mining, which does not need the majority of mining power to launch as well. With selfish mining, miners can obtain the revenue ratio larger than its mining power ratio, which renders PoW-based consensus incentive-incompatible. Nayak et al. [30] generalised the selfish mining to the stubborn mining, and showed how to combine it with the eclipse attack [20] to amplify the attack gain. Different from our research, such attacks still assume that the "honest majority" security assumption holds. We show in this paper that the "honest majority" does not always hold in the real world, and consequently the more severe "sucker punch attacks" can destroy a great number of blockchains.

Our research first investigates the soundness of the "honest majority" and the incentive compatibility without the "honest majority". In fact, some research notices this problem, but either fails to study it from the security foundation level or fails to evaluate it. Bonneau et al. [12] proposed the "bribery attack" that, a mining pool attracts the majority of mining power using low fee rate in order to perform the 51% attack. It first indicates that rational miners can be incentivised to behave maliciously, but neither considers other blockchains as external mining power nor evaluates the impact or the incentive of the bribery attack. Our work generalises the bribery attack to the "sucker punch attack", and evaluate it thoroughly. Kwon el al. [25] investigated the "fickle mining" that, a miner switches between two blockchains depending on their mining difficulties, then proved that such behaviour can be used for weakening the security guarantee of blockchains with less mining power. It first evaluates attacks across different blockchains (BTC and BCH), but neither doubts nor tries to break the "honest majority". In fact, regardless of the profitability, a miner with approximately $\frac{1}{40}$ BTC mining power can break BCH's honest majority (i.e. achieve the majority of mining power on BCH). We will show the incentive (i.e. whether miners are willing to) and the profitability (i.e. how much incentive miners have) of miners to break the "honest majority" in the following sections.

2.4 Evaluation of attacks on PoW-based consensus

Most research evaluating attacks on PoW-based consensus uses the MDP-based models [33, 19, 23, 36] or the game-theoretic models [16, 15, 13, 30, 24, 25, 21]. Our research adopts the MDP-based model to evaluate our proposed "sucker punch attacks", which we call SPA-MDP. SPA-MDP is similar with models in [33, 19] in the notations and the processes, but in addition supports the presence of external mining power. Model the external mining power is complex, because the number of parameters becomes twofold. We reduce the excessive parameters to limited ones (in later sections), which significantly simplifies the implementation and the simulation of SPA-MDP without losing any correctness of the model.

3 Formalising Sucker Punch Attacks

This section presents our system model, the two aforementioned attacks, and our formalisation of the attacks in Markov Decision Process.

3.1 System setting and notations

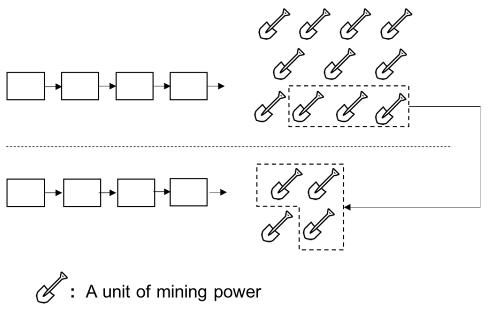


Figure 2: A high-ranked blockchain miner can migrate his mining power to a low-ranked blockchain, and he is able to launch a 51% attack on low-ranked blockchain.

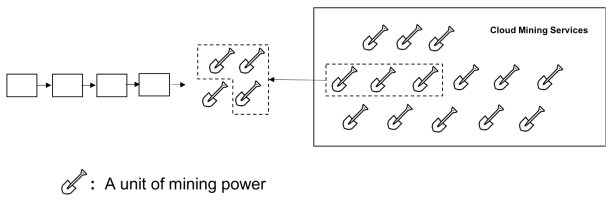


Figure 3: Cloud mining services provide miners extra mining power, so that an adversary can increase his mining power quickly and launch a 51% attack.

We consider a world of more than one blockchain, and some of the blockchains share the same type of mining algorithms. For simplicity, we define two blockchains, BC_1 and BC_2, sharing the same mining algorithm. We call the blockchain with more mining power a *stronger blockchain*, and another one *weaker blockchain*. Since they share the same mining algorithm, the mining difficulty of a stronger blockchain is higher than the weaker one, as they need to create the same number of blocks within the same time period. In this paper, BC_1 is the stronger blockchain, and BC_2 is the weaker one. We also consider rentable cloud mining power from organisations such as NiceHash, and the cloud mining power is compatible with the used mining algorithm.

We consider a rational miner, who is profit driven. When considering mining power migration attack, we assume that the rational miner already has a considerable portion (less than 50%) of mining power on BC_1. When considering cloud mining power attack, we assume that the attacker has some initial investment to rent sufficient cloud mining power. Later in Section 5, we will analyse the soundness of such an environment. Figure 2 and Figure 3 present examples of the two attacks.

For the simplicity of analysis, we also assume that the non-attacking mining power in the two blockchains are constant during the attack. That is, during the time period of the attack, only the attacker is able to move its mining power back and forth between the two blockchains, or to put his rented mining power into the BC_2. This assumption is reasonable as most of the time the mining power is fairly stable at least within a small time period.

Table 1: Notations of Parameters

Notations	definition
D_1	The difficulty of BC_1
D_2	The difficulty of BC_2
d	The fraction of BC_1's difficulty towards BC_2's difficulty ($d = \frac{D_1}{D_2}$)
$H_{a,1}, H_{h,1}$	The honest and adversarial mining power on BC_1, respectively
$H_{a,2}, H_{h,2}$	The honest and adversarial mining power on BC_2, respectively
H_a, H_h	The total honest and adversarial mining power, respectively ($H_a = H_{a,1} + H_{a,2}$, $H_h = H_{h,1} + H_{h,2}$)
h_1	The fraction of the adversarial mining power towards the honest mining power on BC_1 ($h_1 = \frac{H_a}{H_{h,1}}$)
h_2	The fraction of the adversarial mining power towards the honest mining power on BC_2 ($h_2 = \frac{H_a}{H_{h,2}}$)
R_1	The mining reward of a block of BC_1
R_2	The mining reward of a block of BC_2
r	The fraction of BC_1's mining reward of a block towards BC_2's ($r = \frac{R_1}{R_2}$)
v_{tx}	The amount of the attacking transactions
γ	The propagation parameter of the adversary
pr	Renting price of a hash algorithm when mining[2]
β	The fraction of migrated mining power by the adversary

More formally, we define the system environment as follows. A summary of notions is available in Table 1. Let BC_1 and BC_2 be the stronger blockchain and the weaker blockchain in Section 3.1, where BC_2 is the attacker's target. Let pr be the price of renting a unit of mining power (e.g. hash per second) for a time unit. Let D_1 and D_2 be the difficulties, R_1 and R_2 be the mining rewards of BC_1 and BC_2, respectively. We have $D_1 > D_2$ and $R_1 > R_2$. We define the fractions of the difficulties and mining rewards as: $d = \frac{D_1}{D_2}$, $r = \frac{R_1}{R_2}$.

For *mining power migration attack*, we define mining-related parameters for two blockchains. Let $H_{h,1}$, $H_{h,2}$ be the honest mining power, and $H_{a,1}$, $h_{a,2}$ be the adversarial mining power of BC_1 and BC_2, respectively. Let $H_1 = H_{h,1} + H_{a,1}$ and $H_2 = H_{h,2} + H_{a,2}$ be the total mining powers on BC_1 and BC_2, respectively. Note that $H_1 > H_2$ according to the security model. Let $h_1 = \frac{H_a}{H_{h,1}}$ and $h_2 = \frac{H_a}{H_{h,2}}$ be the fractions of the adversarial mining power towards the honest mining power on BC_1 and BC_2. Let $\beta = \frac{H_{a,2}}{H_a}$ be the fraction of migrated mining power by the adversary.

For *cloud mining attack*, since the mining power is not coming from another blockchain, we only consider the target blockchain BC_2. Let $H_{h,2}$ be the honest mining power, and H_a be the rentable mining power from the cloud mining service. Let $h_2 = \frac{H_a}{H_{h,2}}$ be the fraction of rented mining power out of rentable mining power. Let $\beta = \frac{H_{a,2}}{H_a}$ be the fraction of rented mining power towards the rentable mining power. Let pr be the price of renting mining power (in $\$/h/s$).

Let $\gamma \in [0,1]$ be the propagation parameter of the adversary, i.e. the connectivity of the adversary within the network. When the adversary and the honest miners release blocks simultaneously, γ equals to the fraction of the network that agrees on the adversary's block.

3.2 The SPA-MDP model

We design SPA-MDP, a Markov Decision Process (MDP) to formalise the two sucker punch attack (SPA) processes. It takes blockchain parameters as input, and produces the optimal relative revenue of the adversary. In detail, our MDP model is a four-element tuple $M := (S, A, P, R)$ where:

- S is the state space containing all possible states of an adversary.
- A is the action space containing all possible actions performed by an adversary.
- P is the stochastic transition matrix presenting the probabilities of all state transitions.
- R is the reward matrix presenting the rewards of all state transitions.

We detail each element of M below, and present an overview on the state transitions and reward matrices of SPA-MDP in Table 2.

3.2.1 The State Space S

The state space S is defined as a tuple $S := (l_h, l_a, \beta, fork)$, where l_h and l_a are the length of the honest and the adversarial blockchain, respectively; β is the portion of mining power at BC_2 allocated by the adversary; and $fork$ represents the state of the adversarial branch.

Table 2: State transitions and reward matrices of SPA-MDP. Notations are summarised in Table 1.

State × Action	Resulting State	Probability	Reward				Condition
			$\text{cost}_{\text{multichain}}$	$\text{cost}_{\text{cloudmining}}$	blockreward	v_{tx}	
(l_h, l_a, β, $fork$), ADOPT	(0, 0, β, ir)	1	0	0	0	0	$l_h > l_a \geq nConfirm$
(l_h, l_a, β, $fork$), OVERRIDE	(0, 0, β, ir)	1	0	0	l_aR_2	v_{tx}	$l_a > l_h \geq nConfirm$
(l_h, l_a, β, $fork$), WAIT	(l_h, l_a +1, β, p)	$\frac{\beta h_2}{\beta h_2+1}$	$\frac{-\beta h_2 R_1}{d(1+\beta h_2)}$	$\frac{-\beta h_2 D_2 pr}{1+\beta h_2}$	0	0	$l_h < nConfirm$
	(l_h+1, l_a, β, p)	$\frac{1}{\beta h_2+1}$	$\frac{-\beta h_2 R_1}{d(1+\beta h_2)}$	$\frac{-\beta h_2 D_2 pr}{1+\beta h_2}$	0	0	$l_h < nConfirm$
(l_h, l_a, β, $fork$), WAIT_INC	(l_h, l_a+1, β +0.2, p)	$\frac{(\beta+0.2)h_2}{(\beta+0.2)h_2+1}$	$\frac{-(\beta+0.2)h_2R_1}{d(1+(\beta+0.2)h_2)}$	$\frac{-(\beta+0.2)h_2D_2pr}{1+(\beta+0.2)h_2}$	0	0	$l_h < nConfirm$
	(l_h+1, l_a, β +0.2, p)	$\frac{1}{(\beta+0.2)h_2+1}$	$\frac{-(\beta+0.2)h_2R_1}{d(1+(\beta,+0.2)h_2)}$	$\frac{-(\beta+0.2)h_2D_2pr}{1+(\beta+0.2)h_2}$	0	0	$l_h < nConfirm$
(l_h, l_a, β, $fork$), WAIT_DEC	(l_h, l_a+1, β -0.2, p)	$\frac{(\beta-0.2)h_2}{(\beta-0.2)h_2+1}$	$\frac{-(\beta-0.2)h_2R_1}{d(1+(\beta-0.2)h_2)}$	$\frac{-(\beta-0.2)h_2D_2pr}{1+(\beta-0.2)h_2}$	0	0	$l_h < nConfirm$
	(l_h+1, l_a, β -0.2, p)	$\frac{1}{(\beta-0.2)h_2+1}$	$\frac{-(\beta-0.2)h_2R_1}{d(1+(\beta-0.2)h_2)}$	$\frac{-(\beta-0.2)h_2D_2pr}{1+(\beta-0.2)h_2}$	0	0	$l_h < nConfirm$
(l_h, l_a, β, $fork$), MATCH	(l_h, l_a +1, β , ir)	$\frac{\beta h_2+\gamma}{\beta h_2+1}$	$\frac{-\beta h_2 R_1}{d(1+\beta h_2)}$	$\frac{-\beta h_2 D_2 pr}{1+\beta h_2}$	$\frac{(l_a+1)R_2\beta h_2}{\beta h_2+\gamma}$	v_{tx}	$l_h = l_a \geq nConfirm$
	(l_h +1, l_a, β , r)	$\frac{1-\gamma}{\beta h_2+1}$	$\frac{-\beta h_2 R_1}{d(1+\beta h_2)}$	$\frac{-\beta h_2 D_2 pr}{1+\beta h_2}$	0	0	$l_h = l_a \geq nConfirm$
(l_h, l_a, β, $fork$), MATCH_INC	(l_h, l_a +1, β +0.2, ir)	$\frac{(\beta+0.2)h_2+\gamma}{(\beta+0.2)h_2+1}$	$\frac{-(\beta+0.2)h_2R_1}{d(1+(\beta+0.2)h_2)}$	$\frac{-(\beta+0.2)h_2D_2pr}{1+(\beta+0.2)h_2}$	$\frac{(l_a+1)R_2(\beta+0.2)h_2}{(\beta+0.2)h_2+\gamma}$	v_{tx}	$l_h = l_a \geq nConfirm$
	($l_h + 1$, l_a, β +0.2 , r)	$\frac{1-\gamma}{(\beta+0.2)h_2+1}$	$\frac{-(\beta+0.2)h_2R_1}{d(1+(\beta+0.2)h_2)}$	$\frac{-(\beta+0.2)h_2D_2pr}{1+(\beta+0.2)h_2}$	0	0	$l_h = l_a \geq nConfirm$
(l_h, l_a, β, $fork$), MATCH_DEC	(l_h, l_a +1, β -0.2, ir)	$\frac{(\beta-0.2)h_2+\gamma}{(\beta-0.2)h_2+1}$	$\frac{-(\beta-0.2)h_2R_1}{d(1+(\beta-0.2)h_2)}$	$\frac{-(\beta-0.2)h_2D_2pr}{1+(\beta-0.2)h_2}$	$\frac{(l_a+1)R_2(\beta-0.2)h_2}{(\beta-0.2)h_2+\gamma}$	v_{tx}	$l_h = l_a \geq nConfirm$
	(l_h +1, l_a, β -0.2 , r)	$\frac{1-\gamma}{(\beta-0.2)h_2+1}$	$\frac{-(\beta-0.2)h_2R_1}{d(1+(\beta-0.2)h_2)}$	$\frac{-(\beta-0.2)h_2D_2pr}{1+(\beta-0.2)h_2}$	0	0	$l_h = l_a \geq nConfirm$

For simplicity, we choose β from $(0.0, 0.2, 0.4, 0.6, 0.8, 1.0)$. The state $fork$ of the adversarial branch has three values, defined as follows:

- irrelevant ($fork = ir$) means the adversarial branch is published and confirmed in network. This indicates a successful attack.
- relevant ($fork = r$) means the adversarial branch is published but the honest chain is confirmed by the network. This indicates that the attack is unsuccessful at present. (Note that the adversary can keep trying and may succeed in the future.)
- private ($fork = p$) means the adversarial branch is private and only the adversary is mining on it. This indicates that an attack is in process.

3.2.2 The Action Space A

An adversary can perform the following actions:

- **ADOPT** The adversary accepts the branch of the honest network and discards his own adversarial branch, which means the adversary aborts his attack.
- **OVERRIDE** The adversary publishes his adversarial branch (which is longer than the honest one). Consequently, the honest branch is overridden, and the payment transaction from the adversary is successfully reverted.
- **MATCH** The adversary publishes his fork with the same length as the honest blockchain. This action has three variants $MATCH$, $MATCH\_INC$, and $MATCH\_DEC$, where $\_INC$ and $\_DEC$ represent the increase and decrease of malicious mining power ratio β, respectively.
- **WAIT** The adversary keeps mining on his own fork. This action can be performed in two scenarios. One is $l_h < nConfirm$, indicating that the merchant is still waiting for the payment confirmation. Another one is that after the adversary publishes his blockchain by $MATCH$, $l_a \leq l_h$ still holds. In addition, there are three types in this action $WAIT$, $WAIT\_INC$, $WAIT\_DEC$, which represent the adversarial mining power adjustment, similar to that of **MATCH**.

3.2.3 The State Transition Matrix P

The State Transition Matrix P is defined as $S \times A \times S'$: $Prob(S \times A \Rightarrow S')$, where the participant in the state S does the action A to transit his state to S' with the probability $Pr(S \times A \Rightarrow S')$. In the double-spending context, the state transition is invoked by a new block (during **WAIT**), a blockchain branch selection (by **OVERRIDE** or **MATCH**) or quitting the attack (by **ADOPT**).

In MDP, an action can trigger a state S to transit to another state S' with some probability. Note that the resulting state S' can have multiple possibilities $S'_1, S'_2, \cdots, S'_n$, and $\sum_{i=1}^{n} Pr(S \times A \to S'_i) = 1$.

The state transition matrix P (Table 2) describes all state transition possibilities. P is a 3-dimensional matrix $(S \times A \times S')$, where S is the current state, S' is the resulting state, and A is the action triggering this state transition. The value of this matrix is the possibility of the corresponding state transition.

Without Eclipsed Mining Power ($A \in \{$ **WAIT_INC**, **WAIT_DEC** $\}$) When $A \in \{$ **WAIT_INC**, **WAIT_DEC** $\}$, the adversary is mining his private adversarial branch alone. The next state update is triggered by a newly created block either by the honest network or by the adversary, with a probability directly associated to their mining power on BC_2, namely $H_{a,2} = \beta H_a$ of the adversary and $H_{h,2}$ of the honest network.

Therefore, the probability that the adversary gets the next block ($l_a \rightarrow l_a + 1$) is

$$\begin{aligned} P(l_a \rightarrow l_a+1) &= \frac{H_{a,2}}{H_{a,2}+H_{h,2}} \\ &= \frac{\beta H_a}{\beta H_a + H_{h,2}} = \frac{\beta h_2}{\beta h_2 + 1} \end{aligned} \tag{1}$$

And vice versa for $l_h \rightarrow l_h + 1$.

$$P(l_h \rightarrow l_h+1) = 1 - P(l_a \rightarrow l_a+1) = \frac{1}{\beta h_2 + 1} \tag{2}$$

With Eclipsed Mining Power ($A \in \{$ **MATCH_INC**, **MATCH_DEC** $\}$) Besides the mining power owned by the adversary, the eclipsed mining power of $\gamma H_{h,1}$ mines on the adversarial blockchain after a **MATCH** attempt. Therefore, the possibility of $l_a \rightarrow l_a + 1$ becomes

$$\begin{aligned} P(l_a \rightarrow l_a+1) &= \frac{H_{a,2}+H_{eclipsed}}{H_{a,2}+H_{h,2}} \\ &= \frac{\beta H_a + \gamma H_{a,2}}{\beta H_a + H_{h,2}} = \frac{\beta h_2 + \gamma}{\beta h_2 + 1} \end{aligned} \tag{3}$$

And vice versa for $l_h \rightarrow l_h + 1$.

$$P(l_h \rightarrow l_h+1) = 1 - P(l_a \rightarrow l_a+1) = \frac{1-\gamma}{\beta h_2 + 1} \tag{4}$$

3.2.4 The Reward Matrix R

The Reward Matrix R is defined as $S \times A \times S' : Re(S \times A \Rightarrow S')$, where the participant in the state S transits to a new state S' with the reward $Re(S \times A \Rightarrow S')$. The reward from a double-spending attack contains two parts: the block reward (including transaction fees) of the published longer blockchain and the double-spending transaction. To calculate the net reward, we also need to consider the cost of launching an attack. Here, we define the reward $Re(S \times A \Rightarrow S')$ as a tuple (*cost*, *blockreward*, v_{tx}) to fit into the MDP model, where *cost* represents the cost of launching a double-spending attack, *blockreward* represents the reward from the mined blocks including transaction fees, and v_{tx} represents the reward from the double-spent transaction.

With a state transition $S \times A \rightarrow S'$, an adversarial will get some reward, which can be of a positive or negative value. The state transition matrix R is a 3-dimension matrix $(S \times A \times S')$, where S, A and S' are the same as P, but the value of this matrix is the reward of the corresponding state transition.

In our context, the reward consists of two parts, namely the block reward *blockreward* on BC_2 and the reward gained from the double-spent transaction.

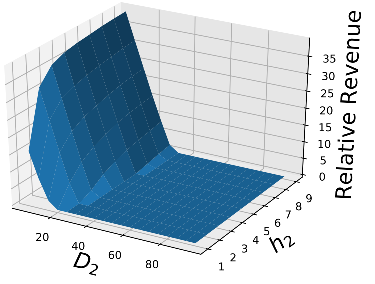
Cost To calculate net revenue, we also need to consider a "negative reward", which is the cost of launching attacks. We use $cost_{multichain}$ and $cost_{cloudmining}$ to denote the cost of launching the mining power migration attack and the cloud mining attack, respectively. Compared to honest mining on BC_1, the cost $cost_{multichain}$ of mining power migration is mainly from the loss of block rewards from BC_1 due to the migrated mining power. Consequently, the cost can be computed as hashrate $\cdot$ time $\cdot$ difficulty $\cdot R_1$, which is the estimated mined block multiplies the block reward of BC_1. For **ADOPT** and **OVERRIDE** actions, the time of finishing a state transition is negnigible. Meanwhile, for **WAIT**-style and **MATCH**-style actions, the state transition is triggered by mining a new block, so the time it takes is depending on the difficulty of mining a block. Therefore, the $cost_{multichain}$ under **WAIT**-style and **MATCH**-style actions can be computed as follows:

$$\begin{aligned} cost_{multichain}(l_a \rightarrow l_a+1) &= cost_{multichain}(l_h \rightarrow l_h+1) \\ &= \text{hashrate} \cdot R_1 \cdot \text{time} \cdot \frac{1}{\text{difficulty}} \\ &= -\beta H_a \cdot R_1 \cdot \frac{D_2}{H_{h,2}+\beta H_a} \cdot \frac{1}{D_1} \\ &= \frac{-\beta h_2 R_1}{d(1+\beta h_2)} \end{aligned} \tag{5}$$

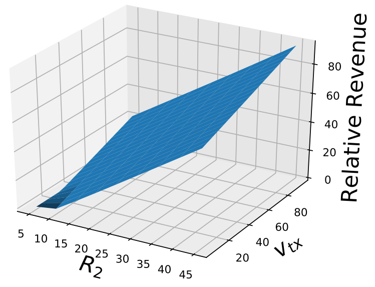
When conducting double-spending attacks by cloud mining, the cost is from renting the cloud mining power. The cloud mining power is priced as $\$/(h/s)/s$, which means the price of renting a mining power unit for a time unit. We denote the cloud mining power price as pr. Similar with the mining power migration, only **WAIT**-style and **MATCH**-style actions take a non-negnigible time period. Therefore, either the cost of **ADOPT** or **OVERRIDE** is 0, and the cost of **WAIT**-style and **MATCH**-style actions is computed as

$$\begin{aligned} R_{BC_1}(l_a \rightarrow l_a+1) &= R_{BC_1}(l_h \rightarrow l_h+1) \\ &= \text{hashrate} \cdot \text{price} \cdot \text{time} \\ &= -\beta H_a \cdot Pr \cdot \frac{D_2}{H_{h,2}+\beta H_a} \\ &= \frac{-\beta h_2 D_2 Pr}{1+\beta h_2} \end{aligned} \tag{6}$$

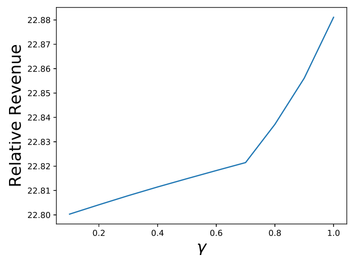
Block Reward on BC_2 The adversary can get the block reward *blockreward* on BC_2 only when his private adversarial branch is broadcasted and accepted by the honest



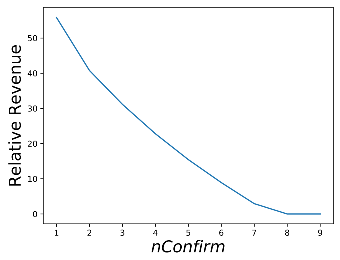
(a) Impact of h_2 and D_2.



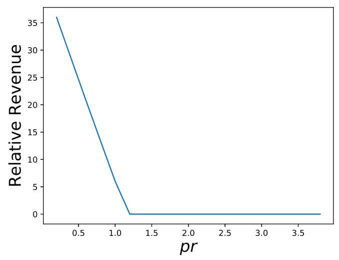
(b) Impact of v_{tx} and R_2.



(c) Impact of γ.



(d) Impact of $nConfirm$.



(e) Impact of pr.

Figure 4: Impacts of Parameters.

network. Therefore, only **OVERRIDE** and the winning scenarios of **MATCH**-style actions have a positive reward, while $blockreward = 0$ under other scenarios.

When performing **OVERRIDE**, the adversarial blockchain of length l_a is directly accepted, so $blockreward = l_a R_2$. When performing **MATCH**-style actions, the adversary needs to get the next block so that his blockchain overrides the honest one. Therefore, $blockreward = (l_a + 1)R_2$ holds for the winning scenarios.

Reward from the Double-Spending Transaction v_{tx} Similar with *blockreward*, the adversary gets the double-spent money only when it successfully overrides the honest branch. Therefore, v_{tx} equals to the transaction amount for **OVERRIDE** and the winning scenarios of **MATCH**-style actions, while $v_{tx} = 0$ for other scenarios.

4 Evaluating model parameters

We implement our SPA-MDP in Python, and theoretically evaluate the impact of different parameters on the two attacks. We show the impact of different parameters on the relative revenue of an attacker. Later in the next section, we will make use of the real-world data to demonstrate the feasibility of these attacks.

4.1 Experimental methodology

4.1.1 Experimental setting

Our model is implemented in Python 2.7 and relies on the *pymdptoolbox* library [14], which is migrated from the Matlab MDP Toolbox. All experiments run on a MacBook Pro with a 2.2 GHz Intel Core i7 Processor, a 16 GB DDR4 RAM and 256 SSD storage disk.

4.1.2 Markov decision process

The proposed MDP model is infinite due to the unbounded l_a and l_h, so we convert the model to the finite MDP for the implementation. This is done by giving an upper bound *limit* for l_a and l_h. We apply the *ValueIteration* algorithm [32] with a discount value of 0.9 and an epsilon value of 0.1 in our MDP-based model.

4.2 Impacts of parameters

We classify the parameters into five aspects, namely 1) mining status related parameters, 2) incentive related parameters, 3) adversary network related parameters, 4) the vigilance of the merchant related parameters, and 5) mining power price related parameters. The mining status related parameters include mining difficulty (D_1 and D_2) and the ratio of adversarial mining power (h_1 and h_2).

The incentive related parameters includes mining reward (R_1 and R_2) and the value v_{tx} of an adversarial transaction. The adversarial network related parameters include the propagation parameter γ of the adversary. The vigilance of the merchant related parameters include the number $nConfirm$ of required block confirmations. The mining market related parameters include pr.

We evaluate the impact of each aspect on the relative revenue of an adversary in both types of attacks, by utilising the *Control Varieties Method*. Table 5 in Appendix A summarises the parameter values used for the evaluation. For aspects with multiple parameters (e.g. the mining status and the incentive), we correlate them by using a 3D surface. Since both attacks have common parameters D_2, h_2, R_2, v_{tx}, γ and $nConfirm$, we evaluate them on *mining power migration attack* only to avoid the repetition.

4.2.1 Impact of mining status

Figure 4a shows the impact of mining-related parameters on the relative revenue. We observe that the relative revenue increases monotonically with D_2 decreasing and h_2 increasing.

When D_2 decreases, mining on BC_2 will be easier, so that migrating to BC_2 will be more profitable. This encourages both types of our attacks on BC_2. When h_2 increases, launching both types of attacks on BC_2 will be more possible to succeed, so the net revenue of attacks will increase. This also encourages 51% attacks on BC_2. Therefore, both decreasing D_2 and increasing h_2 incentivise 51% attacks on BC_2.

4.2.2 Impact of incentive-related parameters

Figure 4b shows the impact of incentive-related parameters on the relative revenue. We observe that increasing R_2 and v_{tx} leads the adversary to profit more.

When R_2 increases, mining BC_2 will be more profitable, and 51% attacks on BC_2 will also be more profitable. This encourages both types of 51% attacks on BC_2. The 51% attack generates v_{tx} out of thin air, so v_{tx} is the direct revenue of the 51% attack, and increasing v_{tx} directly increases the relative revenue. Therefore, both increasing R_2 and v_{tx} incentivise 51% attacks on BC_2.

4.2.3 Impact of adversary network

Figure 4c shows the impact of γ on the relative revenue. In particular, we can see that the relative reward increases slightly with γ increasing. Interestingly, when the attacker's propagation parameter $\gamma = 0.7$, the curve slope increases.

According to our model, γ counts only when the adversary launches the $MATCH$ action. When $h_2 \geq 1$, the adversary can always launch the 51% attack, regardless of the reward. Therefore, the $MATCH$ action is an infrequent choice compared to $OVERRIDE$, so the influence of γ is negligible in our case.

The slope change is suspected to be when $\beta H_a + \gamma H_{h,2} \geq (1-\gamma)H_{h,2}$. At that point, the allocated mining power from the adversary plus his eclipsed honest mining power outperforms the un-eclipsed honest power. Consequently, the adversary is confident to override the small blockchain by $MATCH$ action.

4.2.4 Impact of the vigilance of the merchant

Figure 4d shows the impact of $nConfirm$ on the relative revenue. We observe the relative revenue decreases monotonically with $nConfirm$ increasing, and finally reaches 0.

More block confirmations require the adversary to keep mining secretly for a longer time. This leads to greater cost for launching the 51% attack through both types of attacks, and discourages 51% attacks on BC_2.

4.2.5 Impact of mining power price

The impact of the mining power price pr is shown in Figure 4e. We observe that the relative revenue decreases sharply with pr increasing, and finally reaches 0.

When the price of renting mining power is low, the related blockchains are vulnerable to the cloud mining attack as the attack cost is also low. Increasing pr leads to greater cost of launching 51% attack through renting cloud mining power, which will discourage this kind of 51% attacks on BC_2.

5 Sucker Punch Attack feasibility in the wild

In this section we evaluate our model with the real-world blockchains, including Bitcoin, BitcoinCash, Ethereum, EthereumChassic, Monero, and ByteCoin. In addition, we also revisit and use our model to explain the 51% attacks on EthereumChassic in January 2019 [34].

5.1 Testing blockchains in the wild

In order to evaluate the feasibility and profitability of both attacks, we apply our model to real-world blockchains. First, we evaluate the *mining power migration attack* on 3 pairs of top-ranked blockchains with the same hash algorithm, i.e., (1) Bitcoin (BTC) and BitcoinCash (BCH) with Sha256d, (2) Ethereum (ETH) and EthereumChassic (ETC) with Ethash, and (3) Monero(XMR) and ByteCoin (BCN) with CryptoNight. Second, we evaluate the *cloud mining attack* on 15 PoW blockchains chosen from the top 100 blockchains by their market caps [9]. Unfortunately, our result shows that both attacks are feasible and profitable on almost all selected blockchains.

5.1.1 Mining Power Migration Attack on BTC/BCH, ETH/ETC and XMR/BCN

We evaluate the profitability and feasibility of 3 pairs of top-ranked cryptocurrencies with the same hash algorithm: BTC/BCH, ETH/ETC, and XMR/BCN. Table 6 in Appendix A summarises the blockchain data we use as the input of our model [3]. By permuting the adversary mining power H_a and the transaction value v_{tx}, our experiments reveal their relationship with the relative revenue.

As shown in Figure 5, it is surprisingly easy and profitable for a miner of BTC/ETH to launch a 51% attack on BCH/ETC, but it is difficult and unprofitable for a XMR miner to attack BCN. In detail, the requirement and the profitability of a 51% attack are summarised as follow:

- With approximately 12.5% mining power of BTC ($5000E + 15h/s$), an adversary can gain 6% (150 BCH, or \$18,946.5 USD) extra profit (than honest mining) by double-spending a transaction of 3000 BCH (equivalent to \$378,930).
- With approximately 11.27% mining power of ETH ($16E + 12h/s$), the adversary can gain 1.33% (600 ETC, or \$2,556 USD) extra profit by double-spending a transaction of 90000 ETC (equivalent to \$383,400).
- With approximately 43.05% mining power of XMR ($4E + 8h/s$), the adversary can gain 0.67% (1,000,000 BCN or \$619 USD) extra profit by double-spending a transaction of 600,000,000 BCN (equivalent to \$247,600).

The required mining power is not difficult to achieve. For example, the top three mining pools[4] in ETH are Ethermine (27.7%), Sparkpool (22.2%), f2pool2 (12.5%); and the top three mining pools in BTC are BTC.com (23.0%), AntPool (16.4%), and F2Pool (11.6%). In addition, the adversary may establish a powerful mining pool from scratch via the bribery attack [12].

Table 3[5] provides a summary on the mining power distribution of a selection of blockchains. In this table, the "portion" represents the ratio of a stronger blockchain over a weaker blockchain, where the stronger blockchain is the first row of each mining algorithm, and all other rows of the same mining algorithm are weaker chains. For a stronger chain, the "Top Miners" represents the percentage of mining power that the top mining pools control in the stronger chain. For weaker chains, the "Top Miners" show the ratio of mining power of a top miner over the total mining power of the weaker chain. For example, the top 1 mining pool in ETH controls 27.7% mining power, and this amount of mining power about 4.563 times of the total mining power in the entire ETC network.

However, an XMR miner does not profit much from the mining power migration attack. This is due to the fact that the total available mining power in Monero is only about 2.8 times of the mining power in the BCN, although their market caps differ greatly. In comparison, the total available mining power in BTC is about 27.8 times of the total mining power in BCH; and the total available mining power in ETH is about 16.4 times of the total mining power in ETC.

5.1.2 Cloud mining attack on top 15 PoW blockchains

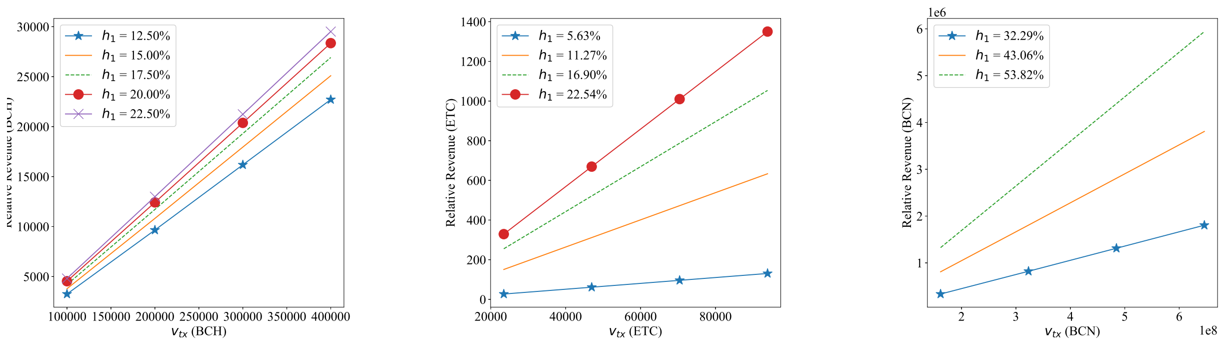
We evaluate the *cloud mining attack resistance* of 15 selected PoW blockchains from Top 100 blockchains by their market cap (Figure 6). We fetched the blockchain data from Coinmarketcap [9] on 19 February 2019, and the renting price of mining power from NiceHash on 07 April 2019. We set $v_{tx} = \$500,000$, and $h_2 = 2$. That is, we assume the rentable mining power is twice of the honest mining power ($h_2 = 2$) in these chains, and the double-spending transaction amount is \$500,000. $nConfirm$ is the value recommended by the according community. We reveal the expected revenue based on these data, and the result is summarised in Figure 6. The result shows that, unfortunately, all selected blockchains are vulnerable towards the *cloud mining attack*, for example:

- the attacker needs approximately \$2,000 to launch *cloud mining attack* on ETC for an hour, and the relative revenue will be \$33899 if successful;
- the attacker needs approximately \$2,600 to launch the attack on BCH for an hour, and the relative revenue will be \$117198 if successful;
- the attack needs approximately \$730 to launch the attack on Electroneum for one hour, and the relative revenue will be \$6222 if successful.

[3]The data was fetched on 05 March 2019 from `https://whattomine.com/` and `https://coinmarketcap.com`

[4]https://www.etherchain.org/charts/topMiners. Data collected on 19/Feb/2019.

[5]Data collected on 19/Feb/2019.



(a) BTC and BCH (b) ETH and ETC (c) XMR and BCN

Figure 5: Testing real-world blockchains.

Table 3: Summary of the blockchains sharing the same hash algorithm.

Type	Mining Algorithm	Coin	Rank	Hashrate (h/s)	Portion	Top Miners		
						#1	#2	#3
ASIC-resistant	Ethash	Ethreum (ETH)	3	1.42E+14	N/A	27.7%	22.2%	12.5%
		EthereumClassic (ETC)	18	8.62E+12	1647.4%	456.3%	365.7%	205.9%
	CryptoNight	Monero (XMR)	14	9.29E+08	N/A	37%	26%	12%
		ByteCoin (BCN)	39	3.35E+08	277.3%	102.6%	72.1%	33.3%
	Equihash	Zcash (ZEC)	20	3.36E+09	N/A	33.4%	19.2%	17.8%
		BitcoinGold (BTG)	26	3.17E+06	111111.1%	37111.1%	21333.3%	19777.8%
		Komodo (KMD)	55	4.48E+07	7518.8%	2511.3%	1443.6%	1338.3%
		Aion (AION)	84	7.22E+05	1000000.0%	334000.0%	192000.0%	178000.0%
ASIC-friendly	Sha256d	Bitcoin (BTC)	1	4.00E+19	N/A	23%	16.4%	11.6%
		BitcoinCash (BCH)	4	1.44E+18	2777.8%	638.8%	455.6%	322.2%
	Scrypt	Dogecoin (DOGE)	23	3.76E+14	N/A	18.0%	16.0%	10.0%
		Litecoin (LTC)	8	2.77E+14	135.7%	24.4%	21.7%	13.6%
	X11	Dash (DASH)	15	2.32E+15	N/A	13.0%	11.0%	11.0%
		WaltonChain (WTC)	73	1.14E+15	203.5%	26.5%	22.4%	22.4%

5.2 Investigating the 51% attack on ETC at Jan. 2019

Ethereum Classic (ETC) is a PoW-based blockchain distributed computing platform, and ETC uses the same hash algorithm with Ethereum (ETH). As shown in the previous section, ETC and ETH are vulnerable to our migration attacks, and ETC is also vulnerable to the cloud mining attack. In fact, a 51% attack happened to ETC on 7 Jan, 2019. NiceHash, a cloud mining service, is a suspected mining power source, though the actual source is still a mystery.

In this section, we apply our SPA-MDP to investigate the 51% attack on ETC. We estimate the net revenue of the attacker, describe the attack strategy adopted by the attacker, and compare the revenue between the mining power renting and the mining power migration.

5.2.1 The attack details

At the beginning of 2019, a 51% attack on ETC resulted in the loss of more than 1.1 million dollars [5]. The attack started from 0:40am UTC, Jan. 7th, 2019 and ended at 4:20am UTC, Jan. 7th, 2019 approximately, lasted 4 hours. The attacker launched a coin withdrawal transaction on the Gate.io exchange [3], then launched double-spending attacks multiple times. As a consequence, 12 transactions out of all attempts were successfully double-spent, listed in Table 4.

The source of the mining power for this attack remains uncertain due to the anonymity of miners. However, the NiceHash cloud mining platform is highly suspected

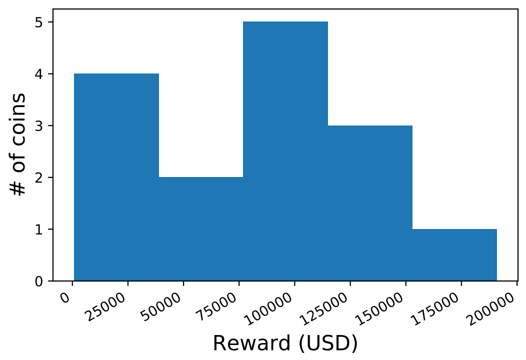


Figure 6: Expected revenue of 15 selected PoW blockchains.

Table 4: All double-spent transactions during the 51% attack on ETC [11]. In this attack, 12 transactions were double-spent from two accounts.

Trans. ID (in short)	From	To	Amount (ETC)	Height	waiting time (#block)
0x1b47a700c0	0x3ccc8f7415	0xbbe1685921	600	7249357	-
0xbba16320ec	0x3ccc8f7415	0x2c9a81a120	4000	7254430	**5073**
0xb5e0748666	0x3ccc8f7415	0x882f944ece	5000	7254646	216
0xee31dffb66	0x3ccc8f7415	0x882f944ece	9000	7255055	409
0xfe2da37fd9	0x3ccc8f7415	0x2c9a81a120	9000	7255212	157
0xa901fcf953	0x3ccc8f7415	0x2c9a81a120	15700	7255487	275
0xb9a30cee4f	0x3ccc8f7415	0x882f944ece	15700	7255554	67
0x9ae83e6fc4	0x3ccc8f7415	0x882f944ece	24500	7255669	115
0xaab50615e3	0x3ccc8f7415	0x53dffbb307	5000	7256012	343
0xd592258715	0x07ebd5b216	0xc4bcfee708	26000	7261492	**5480**
0x9a0e8275fc	0x07ebd5b216	0xc4bcfee708	52800	7261610	118
0x4db8884278	0x07ebd5b216	0xc4bcfee708	52200	7261684	74
					Total: 219500 ETC

[28, 26]. One day before the attack, an anonymous person rents all available Ethash (the hash algorithm used by ETH and ETC) mining power from NiceHash.

5.2.2 Analysing the attacking strategy

According to the actual attack happened, the attacker continuously increased the value of new transactions throughout the attack (except the last double spending of the first account). It is suspected that this behavior belongs to the strategies used by the attacker to maximise and stabilise his revenue, with the following reasons.

First, launching multiple small double-spending attempts can stabilise the expected revenue. The double-spending attack may fail in a limited time period, even if the adversary controls more than 50% of the computing power. Compared to a one-off attempt, the revenue will be more stable if dividing a transaction to multiple smaller transactions.

Second, this strategy may be used for avoiding the risk management system of the cryptocurrency exchanges. Most cryptocurrency exchanges run their own risk management system to combat the misbehaviors, like the fraudulent payments and the abnormal login attempts. A huge coin withdrawal transaction is highly possible to trigger the risk management system, while multiple small transactions would be overlooked. Meanwhile, a big transaction may lead to longer confirmation time, and a longer attack period is easier to be detected. Therefore, defeating the risk control system is naturally a part of the attacker's strategy. According to the Gate.io report [5], the risk management system ignored transactions from the attacker, as the attack was decently prepared - they registered and real-name authenticated the account on Gate.io more than 3 months before the attack. Slowly increasing the transaction value is also highly suspected as an approach for reverse-engineering the threshold of invoking the risk management system.

In addition, we investigate the waiting time between each two attacks (quantified by using the number of blocks). The waiting time varies mostly from 67 blocks to 409 blocks. Interestingly, there are two much bigger gaps of more than 5000 blocks before the transactions 0xbba16320ec and 0xd592258715. The first gap is after the first attack, and the second gap is before the attacker changed to another account to send double-spending transactions. The first gap may be because the attacker was cautious when first launching the double-spending attack. The attacker launched a double-spending transaction of only 600 ETC coins, which is much smaller than his following transactions. After the first attack, the attacker waited for a long time to confirm that the attack is successful, then he started to increase the transaction value. The second gap may be because the attacker ran out of money in his first account 0x3ccc8f7415, and managed to change to another account 0x07ebd5b216. The last transaction 0xd592258715 sent by 0x3ccc8f7415is is right before the second gap. It's value is 5000 ETC coins, which is much smaller than its previous transaction of 24500 ETC coins. After the transaction 0xd592258715, the attacker changed to his another account 0x07ebd5b216, which caused the time gap of 5480 blocks.

5.2.3 Estimating the revenue of the attacker

We use our model to estimate the revenue of the attacker. To analyse this attack, we collect attack-related data during the time period of the attack (on 07/01/2019). Table 7 in Appendix A summarises the experimental data.

With Gate.io, the required number $nConfirm$ of block confirmation is 12, which is also recommended by of ETH community and ETC community [6]. The price

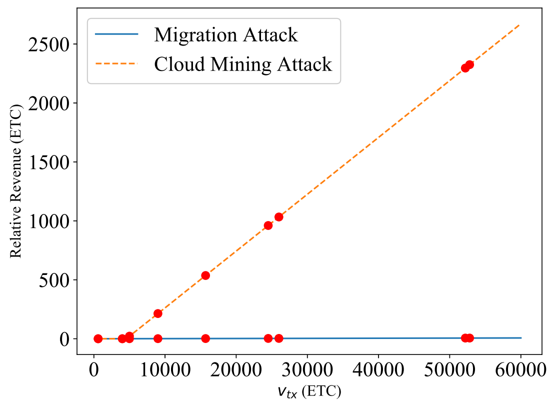


Figure 7: Replay of the ETC Double-Spending Attack. The blue line denotes the performance of *Cloud Mining Attack*, we marked the amount of attack transactions on it in red. The orange line denotes the performance of *Mining Power Migration Attack* as the contrast experiment.

of ETC on that day was \$5.32, and the price of BTC was \$4061.47. The mining difficulty of ETC was 131.80E+12, and the ratio h_2 of attacker's mining power over the honest mining network was about 1.16, i.e., the attacker approximately controls 53.7% mining power during the attack. The reward of successfully mining a block is 4 ETC coins, and the price of renting Nicehash mining power on that day is 3.8290 BTC/TH/day. As there is no data on the attacker's connectivity w.r.t. propagating his blocks, and the impact of γ is relatively small (as previously discussed), we assume that $\gamma = 0.3$.

We permute and mark the transaction values used by the attacker. We also plot the same curve in the mining power migration scenario to compare the profitability of two mining power sources. The result is shown in Figure 7.

The result shows that when the transaction value is over 5000 ETC, double-spending is more profitable than by honest mining. Having a transaction (or a set of transactions) of value over 5000 ETC (approximately 26,000 USD at the time of attack) should not be difficult for an attacker, so the incentive of launching double-spending attacks is very strong.

Moreover, our results give the estimated net revenue of the attacker: \$84773.40. It is approximate to \$100,000 - the value that attacker returned to Gate.io after the attack [4]. Summing all relative revenues of successful transactions, the total relative revenue of the attacker derived from our model is approximately 9000 ETC coins. Recall that the attacker controlled $p = 53.7\%$ of ETC mining power. the probability P of a successful 51% attack is one minus the possibility of failing to attack. The failing scenario is that the adversary mines $n < nConfirm$ blocks when the honest network has mined $nConfirm$ blocks, where $nConfirm$ is the number of confirmation blocks. Mining can be modeled as a binomial distribution model $B(n + nConfirm, p)$, where $n + nConfirm$ blocks will be mined and the adversary mines the next block with the probability p. Therefore, given $nConfirm = 12$, P is calculated as:

$$\begin{aligned} P &= 1 - \sum_{i=0}^{12} C_{i+12-1}^{i} p^i (1-p)^{12} \\ &= 56.48\% \end{aligned} \tag{7}$$

Our model produces the expected net revenue of a single attack, regardless whether it is successful or not. In our case, only successful attacks were observed, but the failed attacks also contribute to the theoretical expected net revenue. The successful attacks contribute to the expected net revenue of 9000 ETC coins, and their possibilities of success are 56.48%. Accordingly, the failed attacks contribute to the expected net revenue of $\frac{9000}{56.48\%}(1 - 56.48\%) = 6934.85$ ETC coins. Therefore, the expected total net revenue is $9000 + 6934.85 = 15934.85$ ETC coins, which is equivalent to \$84773.40 at the time of attack.

The expected net revenue based on our model is \$15226.6 less than the value returned by the attacker. This is because the success of an attack is probabilistic, and our model provides the mathematical expectation rather than the accurate value of the net revenue. According to the probability theory, when the number of attack attempts is small, the real value and the theoretical value will be biased.

Compared to the mining power migration, cloud mining is much more profitable. This means that for the ETH/ETC pair, renting mining power to attack ETC is much cheaper than migrating mining power from ETH. The reason may be the GPU friendliness of the ETH/ETC mining algorithm. ETH and ETC use Ethash [1] as the hash algorithm of PoW. Ethash is a memory-hard function, making it GPU-friendly while ASIC-resistant [2]. As GPU is not dedicated hardware, its mining power can be migrated to any blockchains. Therefore, renting mining power for ETH/ETH is much cheaper compared to renting mining power with dedicated hardware such as ASICs.

6 Detecting and preventing Sucker Punch Attacks

This section discusses potential short term and long term methods to detect and prevent the two sucker punch attacks. We also show how to make use of our model to dynamically adjust some parameters to prevent potential sucker punch attacks.

6.1 Quick remedies

We first discuss several quick remedies for cryptocurrency exchanges to reduce the damage of 51% attacks. It consists of detecting potential attack attempts, and reacting upon detection through conventional risk management techniques.

6.1.1 Detecting 51% attacks

For the sucker punch attacks, the attacker needs to move a considerable amount of mining power from somewhere, such as the other blockchain or a cloud mining service.

This gives us an opportunity to detect the anomaly state where a "large" portion of mining power suddenly disappears. The threshold of "large" is blockchain specific according to the risk management rules. For example, a blockchain which cares less on such attacks can set the threshold to 100% of its current total mining power. That is, after moving this amount of mining power into this blockchain, the new comer will control 50% mining power in total. However, this will not detect an attacker who gains 90% mining power from cloud, and 10% from another source. A more cautious blockchain may set a tighter threshold, e.g. 5%, however, this may cause many false positive alarms.

There are two limitations of this method. First, it may introduce false positive detections, and it is hard to identify which blockchain will be the victim upon detection. Second, it is not cost effective, as it requires significant communication overhead to monitor all possible stronger blockchains and cloud mining services in real-time.

6.1.2 Reactions

Upon detection, a potential victim can react to manage potential risks. Several reactions can be taken to reduce the potential damage from the sucker punch attacks. The first reaction is to increase the number $nConfirm$ of block confirmations. As shown in Figure 4d, in our experiment setting (Table 5), with the increase of required number of confirmations, the related revenue decreases. Second, decreasing the maximum amount of cash out in a single transaction. As shown in Figure 4b, the higher the transaction value is, the more relative revenue an attacker can gain. Thus, decreasing the maximum value of a transaction for cash out would discourage a rational miner to launch sucker punch attacks.

Our model can be used to provide recommendations on the above mentioned parameters. For example, Figure 8 shows the impact of the value v_{tx} of transactions and the number $nConfirm$ of confirmations on the 51% on ETC. This analysis is produced by using our SPA-MDP model, and all other parameters are set up according to the attack

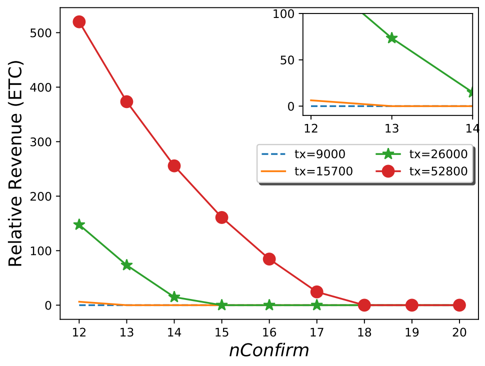


Figure 8: Impacts of v_{tx} and $nConfirm$ on the ETC attack.

happened. This shows that if the value of transactions was limited to 9,000 ETC (approximately $38340.0) per transaction, then the attacker will not get any net revenue. On the other side, if the exchange wants to allow a maximum of 52,800 ETC (approximately $224928.0), then our model recommends that the $nConfirm$ should be increased to 18 to eliminate the incentive of a rational miner to launch such attacks.

In addition, decreasing the maximum frequency of cash out would also limit the potential damage from an attacker, as it reduces the daily withdraw limit.

Last, if a potential attack is considered very likely, then the potential victim can halt the cash out temporarily, to increase the cost of the attack.

6.2 Long term solutions

Though easy to deploy, aforementioned quick remedies are not sufficient. First, they sacrifice the usability of blockchains. Second, all of them only minimise the effect of the potential attacks, rather than eliminating them.

Improving the PoW protocol from the protocol-level is also a promising approach to combat our attacks. There are limited works aiming at minimising the effects of powerful miners being malicious. For example, RepuCoin [35] aims at mitigating the 51% attacks in PoW protocols by introducing the "reputation". In RepuCoin, the weight of each miner is decided by the reputation rather than the mining power. The reputation of a miner depends on the mining power, but also takes the past contribution of miners into consideration. In this way, "sucker punch" cannot raise the reputation in a short time period, and the "sucker punch"-style attacks become much harder.

7 Conclusion

Honest majority is the most important assumption of PoW-based blockchains. However, this assumption does not always hold in practice. We designed our SPA-MDP model to present two possible cases that can break this assumption, including migrating mining power from one chain to the other, and renting cloud mining power.

Our evaluation provided an estimated amount of the cost and the net revenue of each attack. We showed that it is feasible to launch both attacks, and rational miners do have a reason to do so since they are purely profit driven. By using our model, we also provided an analysis on the strategy the attacker uses, in the recent 51% attacks on ETC in January 2019.

References

[1] Dagger Hashimoto, 2019.

[2] Ethash Design Rationale, 2019.

[3] Gate.io - the gate of blockchain assets exchange, 2019.

[4] Gate.io Got Back 100k USD Value Of ETC From The ETC 51% Attacker, 2019.

[5] Gate.io research: Confirmed the etc 51% attack and attacker's accounts - gate.io news, 2019.

[6] How many confirms is considered 'safe' in Ethereum?, 2019.

[7] Nicehash - Largest Crypto-Mining Marketplace, 2019.

[8] The Anatomy Of A 51% Attack And How You Can Prevent One, 2019.

[9] Top 100 Cryptocurrencies by Market Capitalization, 2019.

[10] Aiyer, A. S., Alvisi, L., Clement, A., Dahlin, M., Martin, J., and Porth, C. BAR fault tolerance for cooperative services. In *Proceedings of the 20th ACM Symposium on Operating Systems Principles 2005, SOSP 2005, Brighton, UK, October 23-26, 2005* (2005), pp. 45–58.

[11] Blog, T. C. Deep Chain Reorganization Detected on Ethereum Classic (ETC), 2019.

[12] Bonneau, J. Why Buy When You Can Rent? - Bribery Attacks on Bitcoin-Style Consensus. In *Financial Cryptography and Data Security - FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers* (2016), pp. 19–26.

[13] Carlsten, M., Kalodner, H. A., Weinberg, S. M., and Narayanan, A. On the Instability of Bitcoin Without the Block Reward. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016* (2016), pp. 154–167.

[14] Chadès, I., Chapron, G., Cros, M.-J., Garcia, F., and Sabbadin, R. MDPtoolbox: a multi-platform toolbox to solve stochastic dynamic programming problems. *Ecography 37*, 9 (2014), 916–920.

[15] Eyal, I. The Miner's Dilemma. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015* (2015), pp. 89–103.

[16] Eyal, I., and Sirer, E. G. Majority Is Not Enough: Bitcoin Mining Is Vulnerable. In *Financial Cryptography and Data Security - 18th International Conference, FC 2014, Christ Church, Barbados, March 3-7, 2014, Revised Selected Papers* (2014), pp. 436–454.

[17] Garay, J. A., Kiayias, A., and Leonardos, N. The Bitcoin Backbone Protocol: Analysis and Applications. In *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* (2015), pp. 281–310.

[18] Garay, J. A., Kiayias, A., and Leonardos, N. The Bitcoin Backbone Protocol with Chains of Variable Difficulty. In *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I* (2017), pp. 291–323.

[19] Gervais, A., Karame, G. O., Wüst, K., Glykantzis, V., Ritzdorf, H., and Capkun, S. On the Security and Performance of Proof of Work Blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016* (2016), pp. 3–16.

[20] Heilman, E., Kendler, A., Zohar, A., and Goldberg, S. Eclipse Attacks on Bitcoin's Peer-to-Peer Network. In *24th USENIX Security Symposium, USENIX Security 15, Washington, D.C., USA, August 12-14, 2015.* (2015), pp. 129–144.

[21] Judmayer, A., Stifter, N., Zamyatin, A., Tsabary, I., Eyal, I., Gazi, P., Meiklejohn, S., and Weippl, E. Pay-to-win: Incentive attacks on proof-of-work cryptocurrencies. Cryptology ePrint Archive, Report 2019/775, 2019.

[22] Kiayias, A., Russell, A., David, B., and Oliynykov, R. Ouroboros: A provably secure proof-of-stake blockchain protocol. In *Annual International Cryptology Conference* (2017), Springer, pp. 357–388.

[23] Kiffer, L., Rajaraman, R., and Shelat, A. A Better Method to Analyze Blockchain Consistency. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018* (2018), pp. 729–744.

[24] Kwon, Y., Kim, D., Son, Y., Vasserman, E. Y., and Kim, Y. Be selfish and avoid dilemmas: Fork after withholding (FAW) attacks on bitcoin. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017* (2017), pp. 195–209.

[25] Kwon, Y., Kim, H., Shin, J., and Kim, Y. Bitcoin vs. Bitcoin Cash: Coexistence or Downfall of Bitcoin Cash? *CoRR abs/1902.11064* (2019).

[26] Messamore, W. Nicehash to smaller cryptocurrency miners : If you can't beat 51% attackers who lease our hash power, join them, 2019.

[27] Miller, A. Feather-forks: enforcing a blacklist with sub-50% hash power, 2013.

[28] Morris, D. Z. The Ethereum Classic 51% attack is the height of crypto-irony, 2019.

[29] Nakamoto, S., et al. Bitcoin: A peer-to-peer electronic cash system.

[30] Nayak, K., Kumar, S., Miller, A., and Shi, E. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016* (2016), pp. 305–320.

[31] Pass, R., Seeman, L., and Shelat, A. Analysis of the Blockchain Protocol in Asynchronous Networks. In *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* (2017), pp. 643–673.

[32] Ross, S. M. *Introduction to stochastic dynamic programming*. Academic press, 2014.

[33] Sapirshtein, A., Sompolinsky, Y., and Zohar, A. Optimal Selfish Mining Strategies in Bitcoin. In *Financial Cryptography and Data Security - 20th International Conference, FC 2016, Christ Church, Barbados, February 22-26, 2016, Revised Selected Papers* (2016), pp. 515–532.

[34] Silva, M. D. Ethereum classic is under attack, 2019.

[35] Yu, J., Kozhaya, D., Decouchant, J., and Verissimo, P. Repucoin: Your reputation is your power. *IEEE Transactions on Computers* (2019).

[36] Zhang, R., and Preneel, B. Lay Down the Common Metrics: Evaluating Proof-of-Work Consensus Protocols' Security. In *Proceedings of the 40th IEEE Symposium on Security and Privacy* (2019), S&P, IEEE.

A Experimental data

Table 5: Values of parameters for evaluating the SPA-MDP model.

	Notation	Default	Permuted
Mining Status	D_1	100	N/A
	D_2	10	np.arange(5, 100, 5)
	h_1	0.1	N/A
	h_2	2.0	np.arange(1, 10, 1)
Incentive-Related Parameters	R_1	50	N/A
	R_2	5	np.arange(5, 50, 5)
	v_{tx}	100	np.arange(5, 100, 5)
Adversary Network	γ	0.3	np.arange(0.1, 1.0, 0.1)
the Vigilance of the Merchant	$nConfirm$	4	np.arange(1, 10, 1)
Mining Power Price	pr	2	np.arange(0.2, 4, 0.2)

Table 6: Data of BTC/BCH, ETH/ETC and XMR/BCN for experiments.

(a) BTC and BCH

	BTC	BCH
Difficulty	6071846049920.0	199070336984
Price (USD)	3585.99	126.31
Algorithm	Sha256d	Sha256d
Hashrate(h/s)	39997.52E+15	1444.26E+15
Coins per Block	12.5	12.5

(b) ETH and ETC

	ETH	ETC
Difficulty	1.91E+15	122025268093982
Price (USD)	118.53	4.26
Algorithm	Ethash	Ethash
Hashrate (h/s)	142.00E+12	8.62E+12
Coins per Block	2	4

(c) XMR and BCN

	XMR	BCN
Difficulty	113361254717.0	40879087965
Price (USD)	43.64	0.000619
Algorithm	CryptoNight	CryptoNight
Hashrate (h/s)	9.29E+08	3.35E+08
Coins per Block	3.075	987.26

Table 7: Details of relevant blockchains and mining power prices at the time of attack (Jan. 7th, 2019). The data is from coinmarketcap [9].

ETC Price	\$5.32
BTC Price	\$4061.47
Difficulty	131.80E+12
h_2	1.16
Coins per Block	4
Nicehash Price	3.8290 BTC/TH/day