Blockchain: Research and Applications 5 (2024) 100194

Contents lists available at ScienceDirect

Blockchain: Research and Applications

journal homepage: www.journals.elsevier.com/blockchain-research-and-applications

Research Article

Partial pre-image attack on Proof-of-Work based blockchains

Hamza Baniata *, Attila Kertesz

Department of Software Engineering, University of Szeged, H-6720 Szeged, Hungary

ARTICLE INFO

Keywords:
Blockchain
Proof-of-Work
Security
Partial pre-image attack
Hash functions

ABSTRACT

Blockchain is a type of distributed ledger technology that consists of a growing list of records, called blocks, that are securely linked together using cryptography. Each blockchain-based solution deploys a specific consensus algorithm that guarantees the consistency of the ledger over time. The most famous, and yet claimed to be the most secure, is the Proof-of-Work (PoW) consensus algorithm. In this paper, we revisit the fundamental calculations and assumptions of this algorithm, originally presented in the Bitcoin white paper. We break down its claimed calculations in order to better understand the underlying assumptions of the proposal. We also propose a novel formalization model of the PoW mining problem using the Birthday paradox. We utilize this model to formalize and analyze partial pre-image attacks on PoW-based blockchains, with formal analysis that confirms the experimental results and the previously proposed implications. We build on those analyses and propose new concepts for benchmarking the security of PoW-based systems, including Critical Difficulty and Critical Difficulty per given portion. Our calculations result in several important findings, including the profitability of launching partial pre-image attacks on PoW-based blockchains, once the mining puzzle difficulty reaches a given threshold. Specifically, for any compromised portion of the network ($q < 0.5$; honest majority assumption still holds), the attack is formally proven profitable once the PoW mining puzzle difficulty reaches 56 leading zeros.

1. Introduction

Bitcoin [1] is the first solution that addresses several open issues of a desire to have a distributed cryptocurrency system online. Challenges that delayed the proposal of such a reliable system include the security, privacy, and full decentralization requirements. That is, the success of a given online distributed cryptocurrency solution is attributed to the reliable provision of high security and privacy measures without using a Trusted Third Party (TTP). Several techniques were deployed within the Bitcoin proposal to comprehensively address those issues, including Proof-of-Work (PoW) [2], robust hashing functions [3], Merkle Trees [4], and distributed timestamps [5]. The combination of those techniques resulted in the emergence of the so-called blockchain technology.

The requirements set out in Nakamoto's paper have been reliably proven, both theoretically and experimentally. As we can see, Bitcoin has been increasingly used and adopted for more than 13 years now. Not only Bitcoin, but also the blockchain technology in its generality, was used in a wide variety of applications, including Cloud and Fog Computing [6], distributed e-voting [7,8], e-Health [9,10], IoT [11], smart contracts [12], digital identity [13], and IoV [14].

Upon the proposal of Bitcoin, several main security-related assumptions were made depending on the postulated high security of the cryptography methods utilized. As long as these main assumptions were not violated, the claimed high security of Bitcoin (and similarly any PoW-based blockchain) shall remain dependable.

In this paper, we discuss and simplify those main security-related assumptions. Specifically, we discuss an alternative PoW mining method other than those presented in the original paper [1]. To do so, we present a novel formalization of the PoW mining problem using the Birthday Paradox concepts [15]. We present the attack model, its implications, and related open issues. Meanwhile, we systematically discuss two novel security benchmarks for PoW-based blockchains.

The remainder of this paper is structured as follows. Section 2 provides the necessary background on blockchain, hash functions, and mining. We simplify Bitcoin's assumptions and calculations in Section 3, and formalize PoW mining using the Brithday paradox concepts in Section 4. The attack model, implications, and benchmarking methods are presented in Section 5. Finally, we conclude our paper in Section 6. To facilitate our discussions in the remainder of the paper, we list the main notation and abbreviations used in Table 1.

* Corresponding author.
E-mail address: baniatah@inf.u-szeged.hu (H. Baniata).

https://doi.org/10.1016/j.bcra.2024.100194
Received 25 August 2023; Received in revised form 29 November 2023; Accepted 22 February 2024
Available online 6 March 2024

Table 1
Descriptions of notations and abbreviations used throughout the manuscript.

Notation	Description
V	Set of nodes in network $G(V, \epsilon, w)$
ϵ	Set of edges in network G
$e_{i,j}$	Edge $e \in \epsilon$ connecting nodes $i, j \in V$
$w_{i,j}$	Weight on $e_{i,j}$
N	Number of nodes in the set V
M	Set of honest nodes in V
K	Set of faulty/adversarial nodes in V
f	Number of nodes in the set K
T	Total computational power in blockchain
a_i	Computational capacity of $i \in V$
q	Portion of computational power controlled by K out of T
p	Portion of computational power controlled by M out of T
Ψ	Tolerated upper bound fraction of f out of N, or of q out of T
$h(.)$	Hashing function
Φ	Probability the next valid block is found by a miner $\in V$
E	Probability $m_i \in M$ solves the PoW puzzle
Q	Probability $k_j \in K$ solves the PoW puzzle
Ω	Non-zero digits allowed for a correct PoW
r	Leading zero digits required for a correct PoW
Abbreviation	**Description**
DL	Distributed Ledger
CA	Consensus Algorithm
TX	Transaction
B_z	Most recent confirmed block
B_{z+1}	Next block
B_{z-u}	Block at depth u
P2P	Peer-to-Peer
PoW	Proof-of-Work
UTXO	Unspent TX Output
Nonce	Number-used-Once

2. Background

2.1. Blockchain

A blockchain-based system is characterized by its infrastructure, data structures, networking model, and Consensus Algorithm (CA). The considered infrastructure can be formally described by a set of N nodes $V = \{v_1, v_2, ..v_N\}$, usually termed as miners. Data shared between elements of the set V are described according to the application of the system. For example, transactions (TXs) are submitted by end users to the blockchain network so that they are processed and added to its Distributed Ledger (DL). Usually, TXs are shared with all miners triggering them to generate new blocks of data. A block usually consists of a header and a body. The header may consist of data such as the type of block, the type of CA, the timestamp, the hash of the body, and most importantly, the proof of block validity. The body, on the other hand, usually includes a group of TXs and the hash of the previous block body. More technical details can be found in Ref. [16].

As blockchain nodes form a distributed system, these nodes exchange data through a Peer-to-Peer (P2P) network and communicate by sending messages through directly connected links. Blockchain nodes connect to their peers once they are granted access to the network, making them demonstrable as a graph $G = (V, \epsilon, w)$ of a connected giant component, where ϵ is the set of edges in G representing the communication links between the elements of V. Each $e_{i,j} \in \epsilon$ connects exactly two nodes $i, j \in V$ and can be traveled in both directions. Each $e \in \epsilon$ is associated with a distinct non-negative value, namely weight ($w_{i,j}$ or w_e), which represents the transmission time needed to deliver 1 bit of data from node i to node j or vice versa, computed in milliseconds.

Let $K = k_1, k_2, ..., k_f$ be the set of faulty/adversarial nodes in G and $M = j_1, j_2, ..., j_{N-f}$ be the set of honest nodes, Equation (1) is usually assumed valid for blockchain networks.

$$M + K = V \tag{1}$$

Let the total computational power of the network be T. The attacker then controls a portion q of T that can be calculated using Equation (2), where a_i is the computational capacity associated with node $i \in K$.

$$q = \sum_{i=1}^{f} \frac{a_i}{T} \tag{2}$$

Let the remaining portion p of T be controlled by nodes in the set M. p can then be calculated as in Equation (3), where a_j is the computational power of $j \in M$.

$$p = \sum_{j=1}^{N-f} \frac{a_j}{T} \tag{3}$$

Every blockchain-based system must operate a CA in order to maintain the consistency of its DL [17]. As tens of CAs are proposed in the literature, a CA is usually considered valid if it is proven secure under specific formalized circumstances. One of the main benchmarks used to describe the security level of a given CA is its tolerance for adversary nodes out of the total number of nodes, denoted as Ψ. The tolerance benchmark Ψ is typically mathematically represented by an inequality that relates q to T or f to N. As long as Ψ holds, the blockchain is considered secure.

For example, the Delegated Byzantine Fault Tolerance (dBFT) algorithm [18] is claimed to be secure as long as $f <= \frac{N-1}{3}$, while PoW [1], is claimed to be secure as long as Condition (4) holds.

$$q < T/2 \tag{4}$$

2.2. SHA-256

A hashing function, or a one-way encryption function, $h(.)$, is a mathematical function that takes a variable-length input string and converts it into a fixed-length binary sequence that is computationally difficult to invert [19]. A hashing function enables the determination of a message's integrity: any change to the message will, with a very high probability, result in a different message digest [20].

As an encryption method, hashing functions have been studied and improved over the years to guarantee the highest possible security. The main categories of attacks on hashing functions can be classified as follows:

1. **Collision attack** [21]. This attack tries to find two inputs producing the same hash value, i.e., a hash collision. Classically, a collision attack is described as follows. Find any two different messages $m1$ and $m2$ such that $h(m1) == h(m2)$.
2. **Pre-image attack** [22]. This attack tries to find a message that has a specific hash value. That is, given only the hash value $h(m)$, a pre-image attacker attempts to recover any m' such that $h(m') == h(m)$.
3. **Second-pre-image attack** [23,24]. Given an input $m1$, try to find another input $m2$ (not equal to $m1$) such that $h(m1) == h(m2)$.
4. **Length extension attack** [25]. In this attack, the attacker can use an available $h(m)$ and the length of m to calculate $h(m||m')$, where m' is an extension forged by the attacker. This attack is successful if the attacker can run it without needing to know the content of m.
5. **Brute-force attack** [26]. Simply put, this attack implies sequential or random testing of a wide range of inputs until finding the correct or desired output. Performing such an attack on hashes is considered the easiest to implement but the hardest in terms of cost. There are several ways to use such an attack as it can be used to run any of the previously described attacks. For example, if an attacker needs to know a message m that is used to output the hash $h(m)$, the attacker may sequentially try all possible inputs, hash each input, and check each hash of each input if it is equal to the

desired $h(m)$. Once an input m' is found where $h(m') == h(m)$, the attacker may provably claim that $m == m'$.

Accordingly, the five main properties of a secure and reliable hashing function [27] are

1. **Fixed size of output**: The function takes variable length input and always outputs a string with the same predefined length.
2. **Pre-image resistant**: Given the output, it should be mathematically inefficient to reverse-engineer the original input.
3. **Second pre-image resistant**: Given the input and output, it should be mathematically inefficient to obtain a second input that produce the same output.
4. **Collision resistant**: It is computationally infeasible to find any two inputs that produce the same output.
5. **Random distribution of outputs**: If any single bit of the input is changed, the function will produce an entirely different output.

SHA-256 [3] was proven secure against all known attacks on hashing functions, except for, trivially, the brute-force attack [28,29]. Specifically, the lower bound complexity of algorithmic pre-image or second-pre-image attacks on SHA-256 was evidently reported to be 2^{256}, while the lower bound complexity of algorithmic collision attacks was evidently reported to be 2^{128} [30]. As such bounds make the SHA-256 compliant with all above mentioned properties, it has been deployed in Bitcoin as the mainly-used hashing function.

2.3. Mining in Proof-of-Work blockchain

The mining problem in PoW-based blockchains (as described in the Bitcoin white paper [1]) is defined as finding a nonce that, together with a given block of data m, produces $h(m)$ that complies with the puzzle difficulty. The puzzle difficulty is defined as the dynamically predefined number of leading zeros r in the produced hash. This produced hash, together with the block of data, is called a valid block.

Remark 1. We assume throughout this paper that there is only one required nonce to search for. The effect of using more than one nonce [31] may need further analysis and modifications.

As it is thus far argued that the only way to mine in such PoW-based blockchains is by conducting a brute-force attack, miners should be confident that a miner who provides a correct PoW has worked for sufficient time prior to proposing the valid block. This mining time window should be sufficient for data to propagate throughout the network before the next valid block is produced by another miner. Accordingly, the consistency of local views of confirmed blocks at different physical locations of the network is maintained. To realize this, the puzzle difficulty is regularly modified by the miners, referring to the average time between consequent blocks confirmed within the preceding two weeks and a hard-coded time window (e.g., 10 minutes in Bitcoin and 15 seconds in Ethereum 1.0). Miners that adhere to the rules of the above description are called honest miners and are called adversary miners otherwise.

3. Simplifying Bitcoin's assumptions and calculations

We can break down the original calculations of Bitcoin into simpler pieces. In the original paper, only one attack model was analyzed, where an attacker controls q. Assuming Equation (1) holds, Equation (5) was adopted.

$$p + q = 1 \tag{5}$$

Abstractly, the main objective of the attacker is to find the next block containing its fraud TX, including a valid PoW, before one of the honest nodes in the network finds a valid, non-fraudulent block. In this model, the two portions of the network q and p, i.e., the attacker and honest portions, respectively, are racing towards finding the next valid block.

The Bitcoin proposal was mainly built upon three implicit and explicit assumptions. Next, we attempt to clarify those assumptions in a simple list of points, which would later help us formalize the mining process by referring to the Birthday paradox.

1. **Attack and honest mining mechanism is unified**: Both types of miners use the same mechanism to find a PoW for any given block (i.e., that does or does not include a fraud TX).
2. **Attack and honest resources requirement is unified**: Generating a valid PoW using such a mechanism requires only computational power.
3. **Honest majority**: At any given moment, $p > q$ is valid. Accordingly, honest miners are expected to have a higher probability of finding the next valid block.

Following these assumptions, it is assumed that the only factor affecting the probability of an attack is p. Because of that, portion value notations (i.e., p and q) were also used to denote the probabilities of successful mining by both M and K, respectively. To clarify this mathematically, let Φ denote the probability that the next valid block is found by any given miner $\in V$ (by searching for all probable nonce values, i.e., nonce = 0 to *max_int* "nearly 4 billion in the C language"), and $\overline{\Phi}$ be the probability that the next valid block is NOT found by that miner, then

$$\Phi + \overline{\Phi} = 1 \tag{6}$$

The probability that a miner $\in M$ finds the next block, denoted as E, should then be calculated using Equation (7), while the probability that a miner $\in K$ finds the next block, denoted as Q, should be calculated using Equation (8).

$$E = p \times \Phi \tag{7}$$

$$Q = q \times \Phi \tag{8}$$

Although it was not mentioned clearly, Bitcoin calculations assumed that $\overline{\Phi}$ is always equal to 0, which resulted in Equation (9) (by summing Equations (7) and (8), with reference to Equation (5), and depending on the correctness of the first two assumptions). This equation justifies why p and q were used instead of E and Q, respectively. This special case was evaluated in the original paper, using Equation (10), to compute the attacker's potential progress λ for a given block at depth z. However, we can see in practice that for high-difficulty values (e.g., currently it is 19–20 leading zeros), miners frequently do not find any nonce that solves the puzzle at hand, and in such a case, they may shuffle TXs in the body, generate new Merkle Root, and start mining again from nonce = 0 to *max_int* as the new Merkle Root implies a new block to be mined. Fig. 1 demonstrates this process with two main mining cycles. The first cycle (with Blue nodes) is the first mining process approached by miners, which indicates the search for a valid nonce. The second cycle (including Orange nodes) is the extension of the mining process in case none of the tested nonce values generate a valid puzzle solution.

Obviously, if Φ is always equal to 1, there is no need for cycle 2. However, this cycle path is taken by miners, which indicates that $\Phi < 1$ and does not agree with the initial Bitcoin proposal assumption. Several other inaccuracies of the formalization and validation in the original Bitcoin proposal were discussed in Refs. [32–34].

$$E + Q = 1 \tag{9}$$

$$\lambda = z\frac{q}{p} \tag{10}$$

By the time of writing the original paper, there was no evident algorithm proving an efficient method to run pre-image attacks on SHA-256.

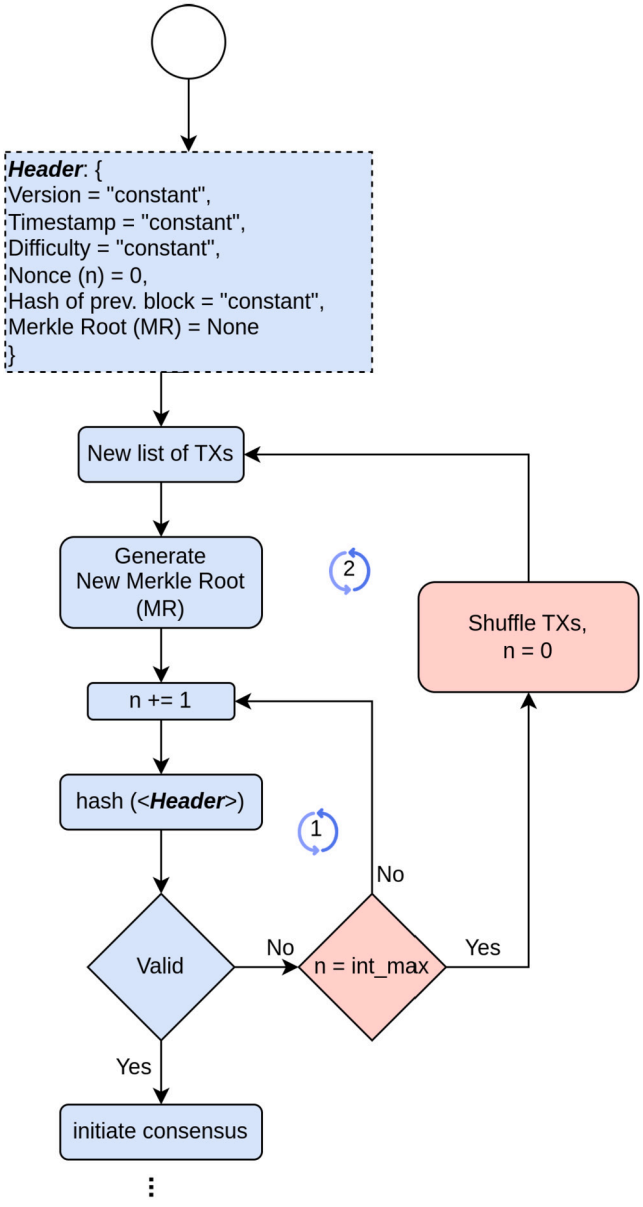

Fig. 1. Mining process in Proof-of-Work based blockchains (e.g., Bitcoin). The first loop (with Blue nodes) indicates the process of changing only the nonce value while keeping all *Header* values constant. The second loop (with Orange nodes) indicates the process of changing the body of the block being mined (thus, the Merkle Root as well) when the maximum nonce value is reached and no valid solution is found.

However, several researchers [22,35–39] attempted and are still attempting to run such attacks. As long as a pre-image attack cannot be run on SHA-256, the only approach available to solve Bitcoin's mining problem is to run a brute-force attack (first assumption), which only requires computational capacity (second assumption).

The Bitcoin paper claims that an attacker cannot create value out of thin air. That is, a user must be initially sent some amount of cryptocurrency. Accordingly, this user can refer to this first TX's output to be used as input for new (fraud or honest) TXs as described earlier. As long as there is a reliable mechanism utilized to prevent an attacker from forging an already confirmed Unspent TX Output (UTXO), this claim is valid.

Since the third assumption cannot be anyhow controlled or guaranteed by any system entity as there is no TTP, it was argued in the original paper that even if this assumption does not hold anymore, it would be more profitable for an attacker to play by the rules. That is, to use their controlled majority portion of computational power to create value for themselves (mint out of thin air), rather than undermine the system and the validity of their own wealth. This argument does not seem to be scientifically convincing as justifications for undermining the system may vary. Nevertheless, if the attacker indeed follows this argument, it is easily discoverable whether the third assumption does not hold anymore (i.e., using tools such as Ref. [40]). Even with an attacker that follows the rules, the system would lose its credibility against user trust once it is found that $q > p$.

4. Formalizing the PoW mining problem using the Birthday paradox

We can formalize the PoW mining problem using the Birthday paradox [15] as follows. Let the set $H = 1, 2, ..., \xi_H$ consist of all probable numbers that, if concatenated with r zeros, would produce a correct hash. A correct hash h is then describable as the concatenation $h = ('0' \times r) + hc$ where $hc \in H$. To simplify our description, let the set H' consist of all correct hashes, and the set H'' consist of all incorrect hashes. The relation between r and Ω is defined in Equation (11).

$$r = 64 - \Omega \tag{11}$$

Remark 2. The number of elements ξ in H and H', denoted as ξ_H and $\xi_{H'}$, respectively, is trivially $2^{4\Omega}$, since the hash is decoded at base 16. ξ_H and $\xi_{H'}$ are bounded for SHA-256 ($1 \leq 2^{4\Omega} \leq 2^{256}$).

Remark 3. The number of elements in H'', denoted as $\xi_{H''}$, is $2^{256} - 2^{4\Omega}$.

Furthermore, let the set $C = 1, 2, ..., \xi_C$ consist of all values that a miner would test as a nonce for a given block before it drops the mining task of this block. The number of elements in C (ξ_C) is equal to the last nonce a miner would test.[1] As of March, 2022,[2] the default last tested nonce in Bitcoin is 2^{32} [41]. Now, let the set HC with length ξ_{HC} be a merged set of H and C. ξ_{HC} can then be defined using Equation (12).

$$\xi_{HC} = \max\{\xi_C, \xi_H\} \tag{12}$$

Let $test(hc_1, hc_2) \rightarrow \{0, 1\}$ be a function that tests pairs of elements $(hc_1, hc_2), \forall hc \in HC$, where its output is 1 if mining hc_1 would produce $h = ('0' \times r) + hc_2$ (or vice versa), and is 0 otherwise. Let MATCH be the event where $test(hc_1, hc_2) \rightarrow 1$, while NON-MATCH be the event where $test(hc_1, hc_2) \rightarrow 0$. We can also define the potential of using the set C to produce a MATCH using Equation (13).

$$\varpi = \frac{\xi_C}{\xi_{HC}} \tag{13}$$

The MATCH probability for any pair of elements in the set HC equals 1 divided by the combined lengths of the sets H' and H'' (i.e., $\xi_{H'} + \xi_{H''} = 2^{256}$ as per Equation (11) and Remarks 2 and 3). For any two pairs, it would be $2/(\xi_{H'} + \xi_{H''})$. Consequently, the probability of a MATCH Φ, and the probability of NON-MATCH $\overline{\Phi}$ can be calculated using Equations (14) and (6), respectively. Note that we use the same notations for MATCH and NON-MATCH as for the previously discussed probabilities of finding and not finding a valid block. This is because, essentially, both are the same. Specifically, we define MATCH $= 1$ if a tested nonce that belongs to the set of nonce values tested (up to max_int), produces a valid hash. Thus, the probability of MATCH is equal to the probability that the next valid block is found.

[1] C and C++Integer Limits: docs.microsoft.com/en-us/cpp/c-language/cpp-integer-limits?view=msvc-170.

[2] Github.com/bitcoin/bitcoin/blob/640eb772e55671c5dab29843cebe42ec35cb703f/src/rpc/mining.cpp.

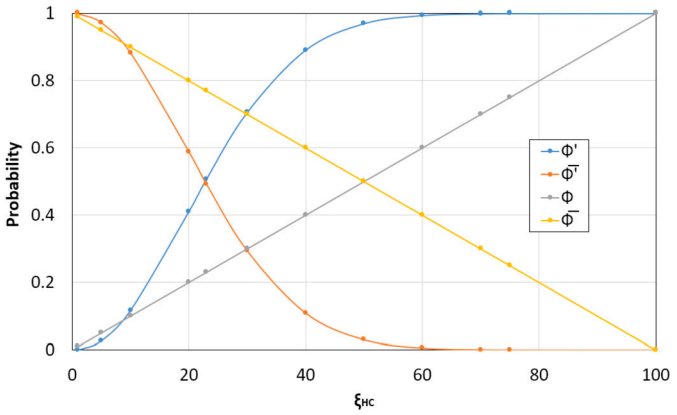

Fig. 2. Graphs showing the approximate MATCH and NON-MATCH probability for $\xi_{HC} \in [0, 100]$ and $\xi_{H'} + \xi_{H''} = 100$, using Equation (14) (Gray), Equation (6) (Yellow), Equation (15) (Orange) and Equation (16) (Blue).

$$\Phi = \frac{\xi_{HC}}{\xi_{H'} + \xi_{H''}} \tag{14}$$

However, $\overline{\Phi}$ can be calculated according to the Birthday paradox using Equation (15), which results in Φ being calculated using Equation (16). Fig. 2 represents the approximate MATCH and NON-MATCH probability distributions for sampling $\xi_{HC} \in [0, 100]$ and $\xi_{H'} + \xi_{H''} = 100$ using all these equations.

$$\overline{\Phi'} = \prod_{y=1}^{\xi_{HC}-1} \frac{(\xi_{H'} + \xi_{H''}) - y}{(\xi_{H'} + \xi_{H''})} \tag{15}$$

$$\Phi' = 1 - \overline{\Phi'} \tag{16}$$

Naturally, both Equations (14) and (16) can be deployed in Equations (7) and (8), which can be used for both honest and attacker miners, respectively, as long as the first two assumptions hold. Putting it differently, if both honest and attacker parties have equal MATCH probabilities, Equation (9) would still hold. However, in the case where one or both of the first two assumptions are violated, Q would require further considerations depending on the attack model (thus Equations (8) and (9) may not be valid).

5. Partial pre-image attack

5.1. Attack model

This attack is a special-case and relaxed version of the pre-image attack: given r bits of a hash value $h(m1)$, find an input $m2$ such that $h(m2)$ matches $h(m1)$ in the specified r bits [42]. This attack is claimed in the literature to be inefficient for SHA-256 due to the high output distribution randomness of SHA-256. Mapping this attack into PoW mining described earlier, Equation (11) defines r, while $h(m1)$ can be any hash that consists of r leading zeros (i.e., any element in the set H'). One approach to running this attack is to find $h(m2) = ('0' \times r) + s$ of a constant $m2$ by only searching for a correct $s \in H$ instead of searching in all 2^{256} possible outputs of SHA-256.

A few previous works investigated such an interesting approach. For example, Heusser [41,43] proposed encoding the nonce search as a decision problem solved by an SAT solver [44] in such a way that a satisfiable $m2$ contains a valid nonce. The key ingredients in the algorithm are a non-deterministic nonce and the ability to take advantage of the known structure of a valid $h(m2)$ referring to an element in H' using assume statements. Although it was not formalized, the method's efficiency was reported to be negatively proportional to Ω and, hence, it (potentially) gets more efficient with increasing the puzzle difficulty. A miner that uses this approach shall then find puzzle solutions quicker than a miner that uses the classical brute-force mining approach. Note that this approach implies that the attacker would use a different mining mechanism and (partially) different resources. As a result, a successful attack using this approach violates both the first and the second assumptions.

Other works investigated the utilization of SAT solvers for other purposes, such as in Ref. [45] where cryptanalysis was conducted against MD4 and MD5 hash functions and for other applications [46,47]. However, there is an unjustifiable lack of research work in, specifically, exploiting SAT solvers to perform partial pre-image attacks on PoW-based solutions.

In this section, we investigate the potential of the approach presented in Refs. [41,43] in particular. If this approach is provably implemented, we can redefine Q referring to our formalization model of PoW mining, as shown in Equation (17).

$$Q' = \frac{q\varpi\ \xi_{HC}}{\xi_{H'}} \tag{17}$$

Remark 4. We have optimistically extracted Equation (17) from Equation (14), which provides less probability for a successful attack. If Equation (17) is to be extracted from Equation (16), it would provide a higher probability for the attack.

5.2. Implications

The default configuration in PoW-based blockchains, increases the difficulty as generating new blocks gets faster. If the probability of a successful attack using the approach discussed above gets higher with increased difficulty, the increment in difficulty might then lead to swift collapse of the Bitcoin's system security before a hard fork of the core protocol is available. The hard fork shall be able to detect such adversary behavior within the network and dynamically adjust the difficulty modification mechanism. However, even with such adjustment, the majority of honest nodes converting to this method might lead to decreasing difficulty, which makes the classical mining mechanism arise as an attack in a scenario similar to an Oscillating Universe [48]. A proven secure, practical, and comprehensive method for detecting and addressing such attacks is then urgently needed.

We can predict the implications of using such an approach for attacking PoW-based blockchains by comparing the honest majority success probability in Equation (7) versus the attacker's success probability in Equation (17). Consequently, an attacker can have an advantage over honest nodes as follows:

$$\begin{aligned} E \ &<?\ Q' \\ &\Rightarrow p \times \frac{\xi_{HC}}{\xi_{H'} + \xi_{H''}} \ <?\ q \times \frac{\varpi \xi_{HC}}{\xi_{H'}} \\ &\Rightarrow \frac{1}{2^{256}} \ <?\ \frac{q}{p} \times \frac{\varpi}{2^{4\Omega}} \end{aligned}$$

which gives:

$$\frac{1}{\varpi \times 2^{4r}} < \frac{q}{p} \tag{18}$$

Inequality (18) is, in fact, a direct utilization example of Equation (10). This inequality represents a condition upon which a successful attack can be conducted (using the approach discussed above). To realize our following calculations, we take the Bitcoin system as a representative solution, where PoW is utilized. Only for attack probability demonstration purposes[3] do we take the pool that controls the highest hash rate per year as K and the remaining pooled and non-pooled miners as M. Assuming that $\varpi = 1$ (i.e., $\xi_H \leq \xi_C$), we present in Fig. 3 historical data[4] for p, q and the accompanying values of q/p. More in-depth anal-

[3] We are not accusing any party as our intention is to clarify our math analysis for the presented security threat.

[4] BTC GUILD: btc.com/stats/pool?percent_mode=2013#pool-history.

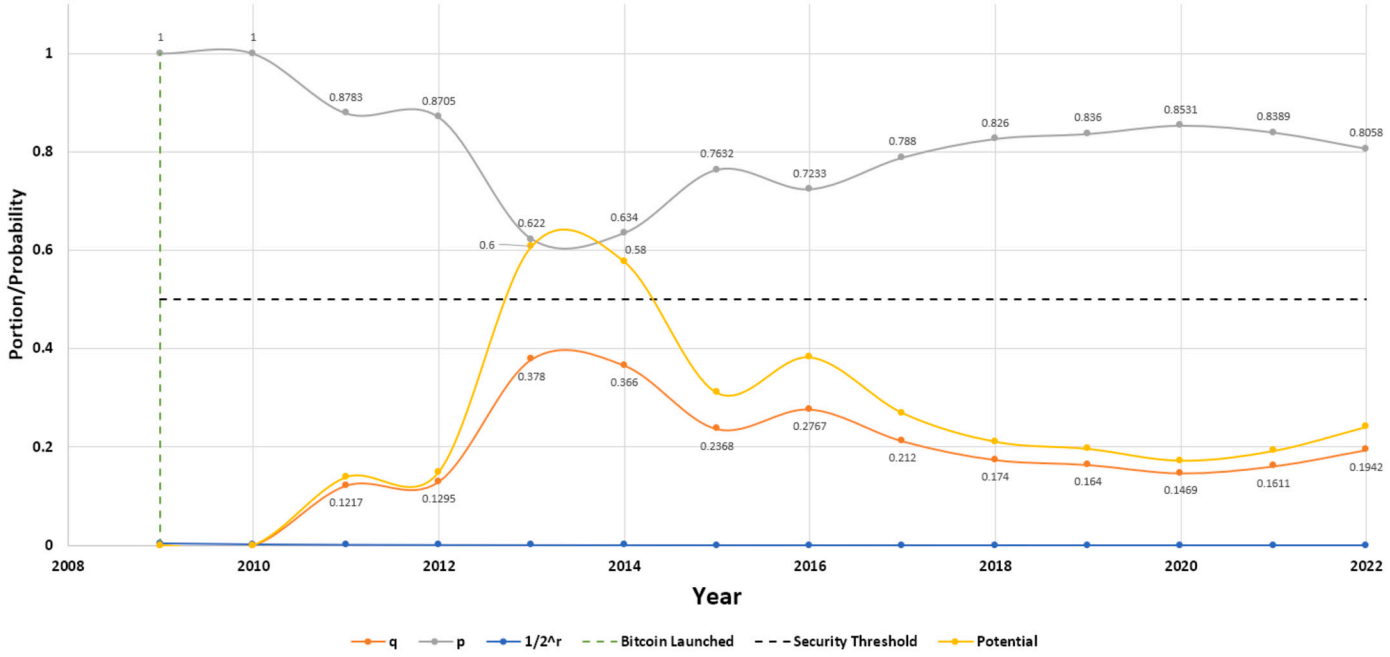

Fig. 3. Historical data for maximum portion controlled by the dominating mining pool per year.

ysis regarding mining pools can be found in Ref. [49]. It can be noticed from the figure that Inequality (18) has held indeed since the year 2010 (thus associated with the emergence of mining pools), while the maximum portion historically ever controlled by a single mining pool was reached in the year 2013.

Nevertheless, even when Inequality (18) holds, the attacker may not be able to catch up with the honest majority with a significantly dominating computing capacity controlled by an honest majority. That is, such dominance shall compensate for the system's decreasing ability of attack tolerance due to the increment of the difficulty. It would be beneficial, then, to benchmark and redefine a threshold for such a case. The aim of this benchmark is to provide exact critical r and q values upon which an attacker can successfully run the attack. Since violating the third assumption would not require further violations to control the network, we will only compute the Q' values while the third assumption still holds (i.e., using Equation (17) while $p > q$). As long as $Q' < p$, the discussed partial pre-image attack should not be considered as a major threat. Note that we assume that Φ for the honest majority equals 1, which is the best case possible for M.

We test two cases where $\xi_{H'}$ equals 2^{Ω} (hypothetical) and $2^{4\Omega}$ (realistic). The results for computing Q' in those cases are presented in Figs. 4 and 5, where the red arrows point to the correlated p value for each q curve.

To check where the Bitcoin system currently stands against the presented attack, the puzzle difficulty r is currently 19 digits (out of 64 hash digits as of August-2023[5]) and has been around this level for almost 3 years [50]. To successfully run the discussed attack, the attacker would need to further violate the third assumption, which has never been the case, or wait until r exceeds a critical value. It is worth noting as well that the analysis of space-time trade-offs [51] for both mining approaches (i.e., brute-force and partial pre-image) is highly important. Specifically, ϖ and q in Equation (17) represent the storage capacity and the computational capacity of the attacker, respectively. Although Bitcoin's original proposal assumed unlimited storage resources (in its Section 11), we expect Q' to be more constrained due to (realistic) limited resources assumption, resulting in further reduction of successful attack probability.

5.3. Benchmarking

In light of the results presented so far, we propose the concepts of **Critical Difficulty** ($\hat{r}$) and **Critical Difficulty per Portion** $\hat{r}$ (q).

Definition 1. Critical Difficulty ($\hat{r}$) is the lowest PoW difficulty r after which Equation (9) would not hold anymore.

We state according to our analysis that $\hat{r} = 56$. Referring to Remark 4, $\hat{r}$ value might be different if Equation (17) is extracted from Equation (16). We leave, as an open issue, the determination and proof of the $\hat{r}$ equation.

As shown in Figs. 4 and 5, once the puzzle difficulty r exceeds $\hat{r}$, Equation (9) does not hold anymore. Specifically, the following two experimental observations provided in Ref. [41] are conditionally proven by our formalization and analysis:

1. The proposed algorithm gets more efficient with increasing the puzzle difficulty.
2. The search time of the proposed algorithm is not linearly correlated with the nonce range.

The results shown in Figs. 4 and 5 prove that, strictly, as long as $r > \hat{r}$, the first observation is indeed true. Taking it from another perspective, let the efficiency of the algorithm be defined with reference to the maximum number of nonce values the algorithm needs to try to obtain a solution. Increasing the difficulty r implies the decrement of Ω leading to the decreased number of required nonce values to be tested (Equation (12)), while the algorithm's potential to find the solution is maintained (Equation (13)).

For the second observation, this is indeed expected since increasing ξ_C simultaneously increases both the search time and the probability of finding a solution (see Equations (13) and (17)). That is, higher ξ_C requires the deployed solver to search for more paths for a solution, as it uses the DPLL backtracking algorithm which shall consume more processing time. To clarify the correlation, we assume two hypothetical

[5] explorer.btc.com/btc/block/737856.

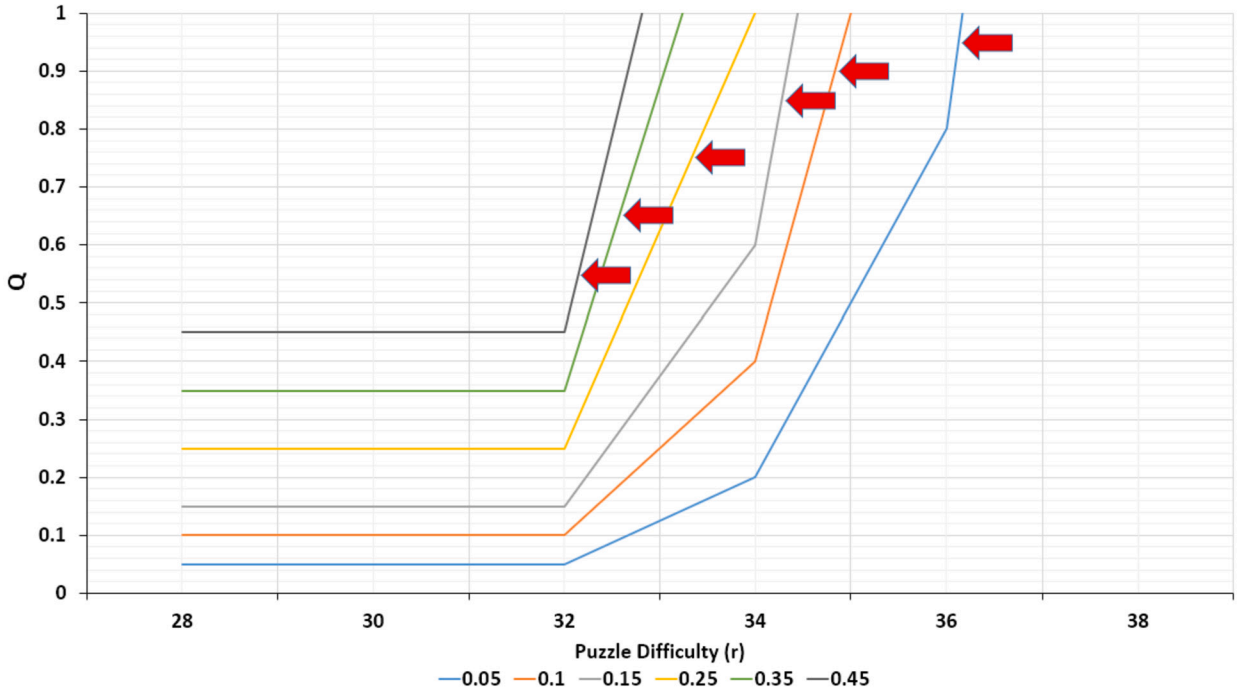

Fig. 4. Partial pre-image attack probability on Bitcoin's Proof-of-Work based system, where $0 < r < 64$, $q \leq 0.45$, $\xi_{H'} = 2^{\Omega}$, and $\varpi = 1$. Red arrows indicate the horizontal grid line for the relevant honest proportion p.

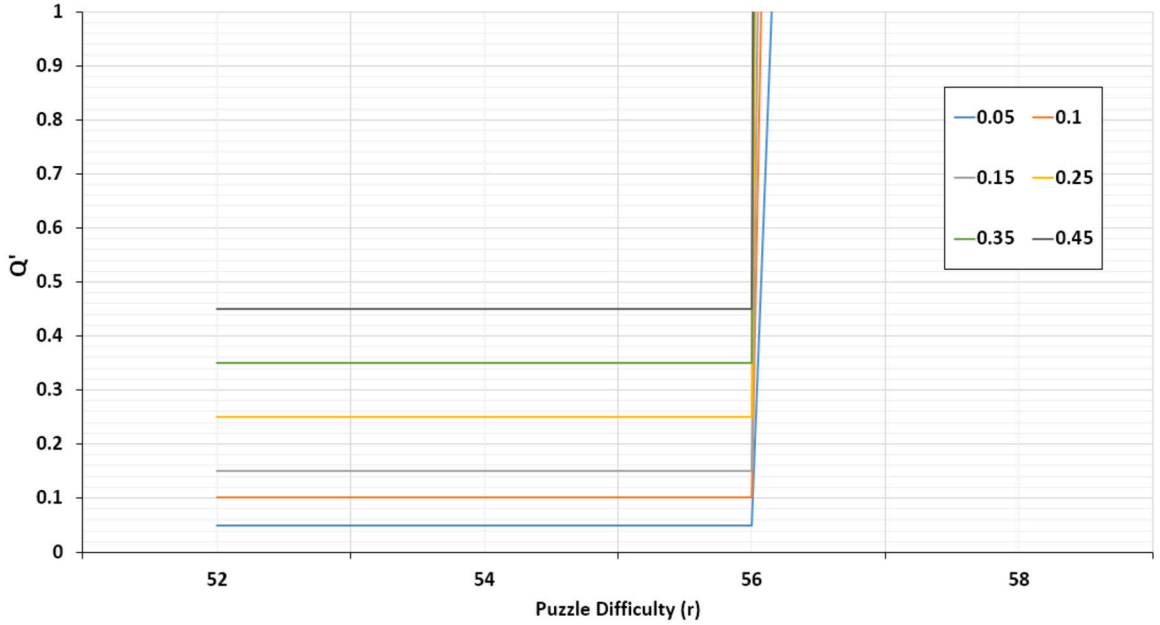

Fig. 5. Partial pre-image attack probability on Bitcoin's Proof-of-Work based system, where $0 < r < 64$, $q \leq 0.45$, $\xi_{H'} = 2^{4\Omega}$, and $\varpi = 1$.

sets $C1$ and $C2$ with lengths ξ_{C1} and ξ_{C2}, respectively, where $\xi_{C1} < \xi_{C2}$. The search using $C1$ shall take time $T1 < T2$, yet if a MATCH is not found in $C1$, it is more likely to be found sooner using $C2$ as per Equation (13).

To demonstrate how our equations and definitions can be used, we forecast r in order to predict when Bitcoin would reach $\hat{r}$ (for a hypothetical $\hat{r} = 32$) depending on the historical evolution of r through time. We extracted the historical data (number of leading zeros (r) and the hash rate of the network, measured from January 2010 until March 2022) using Google's BigQuery[6] database. Using a linear trending approach, we can expect r to reach $\hat{r}$ by the year 2040. Similarly, we can expect that r would reach $\hat{r} = 56$ by the year 2073. Fig. 6 presents the data we extracted along with our forecast.

Definition 2. Critical Difficulty per Portion ($\hat{\mathbf{r}}$ ($\mathbf{q}$)) is the lowest PoW difficulty r at which Q becomes greater than, or equal to, p.

[6] https://cloud.google.com/bigquery/public-data/.

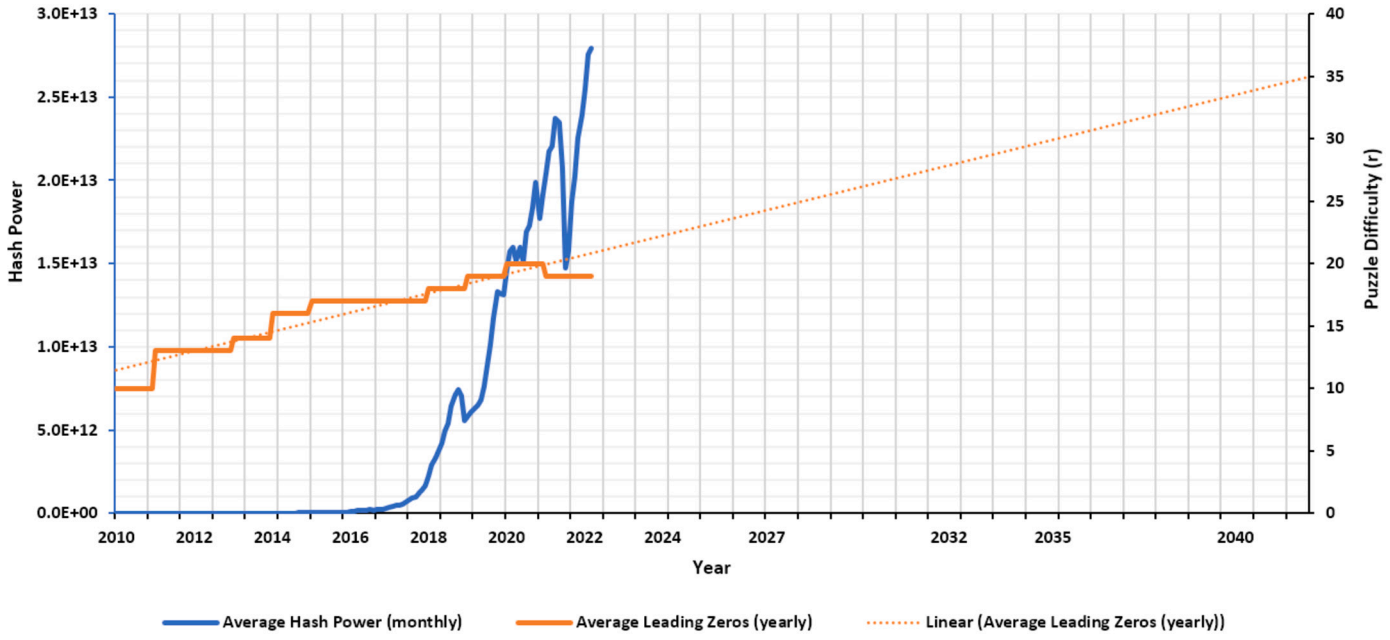

Fig. 6. Bitcoin's monthly hash rate averages (Blue curve correlated with the primary y-axis on the left), yearly puzzle difficulty (r) averages (Orange curve correlated with the secondary y-axis on the right) for the period Jannuary, 2010 – March, 2022, and linear forecast of r until $r = 35$.

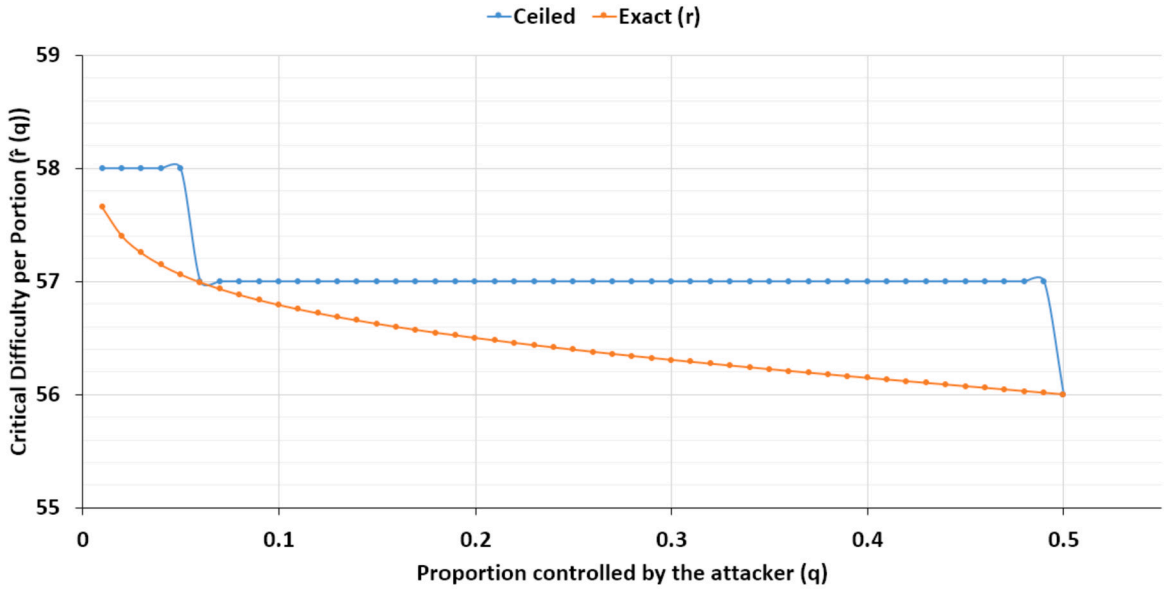

Fig. 7. Exact and ceiled Critical Difficulty per Portion controlled by an attacker $\hat{r}(q)$, where $\log_2 \xi_C = 32$.

Remark 5. Our definition of **Critical Difficulty per Portion** $\hat{r}\ (q)$ assumes that Φ for the honest majority equals 1, which is the best case possible for M.

Taking the optimistic worst case for K by using Equation (17) (see Remark 4) to calculate Q, the equation of $\hat{r}\ (q)$ for Bitcoin can be derived as follows:

$$
\begin{aligned}
Q' = p \Rightarrow q \times \frac{2^{\log_2 \xi_C}}{2^{\log_2 \xi_{H'}}} &= p \\
\Rightarrow 2^{\log_2 \xi_C - \log_2 \xi_{H'}} &= p/q \\
\Rightarrow \ln 2^{\log_2 \xi_C - \log_2 \xi_{H'}} &= \ln p/q \\
\Rightarrow (\log_2 \xi_C - \log_2 \xi_{H'}) \times \ln 2 &= \ln p/q \\
\Rightarrow \log_2 \xi_C - \log_2 \xi_{H'} &= \frac{\ln p/q}{\ln 2} \\
\Rightarrow \log_2 \xi_{H'} &= \log_2 \xi_C - \log_2 p/q \\
\Rightarrow \log_2 \xi_{H'} &= \log_2 \frac{q \times \xi_C}{p}
\end{aligned}
$$

Since $\log_2 \xi_{H'} = 256 - 4r$:

$$\hat{r}(q) = (256 - \log_2 \frac{q \times \xi_C}{p})/4 \tag{19}$$

Similar equations can be further derived from Equation (19), e.g., Critical Portion per Difficulty $\hat{q}\ (r)$ and Critical Difficulty per Portion per Depth of attacked block $\hat{r}(q, z)$. We attempt to use the ceiling operation in Equation (19) to estimate the first r value at which the inequality $p \leq Q$ is guaranteed valid. However, the Floor operation can be used instead to estimate the lowest r value tolerated by the system for a given q.

We also used $\log_2 \xi_C = 32$ similar to its value for honest mining. Note that decreasing $\log_2 \xi_C$ trivially increases $\hat{r}$, which further secures the system and, thus, conforms with the results presented in Ref. [41]. Specifically, the experiments presented in Ref. [41] configured ξ_C to equal 1000 and 10,000, which gives $\log_2 \xi_C \approx 10$ and 13, respectively. We calculate $\hat{r}(q)$ for $q \in [0.01, 0.5]$ and $\log_2 \xi_C = 32$. The exact and ceiled results are depicted in Fig. 7.

In summary, it is clear from Equations (17), (18), and (19) that the presented attack is a practical case where Equation (5) cannot be substituted by Equation (9). Additionally, it is clear from the results presented that violating only the first two main assumptions indeed weakens the security of Bitcoin and similar systems more than it was originally expected. In particular, the presented attack is a practical case where violating only those two assumptions can lead to a successful security attack on PoW-based blockchains, even if this case is widely thought to be non-existent or non-practical.

6. Conclusion

In this paper, we discuss the security of PoW-based blockchain referring to the calculations presented in the original Bitcoin white paper. With this, we observe that the accuracy of these calculations depends mainly on specific assumptions, which, in some cases, may not be correct. To test our observations, we propose a novel formalization model of the PoW mining problem using the Birthday paradox. Building on this model, we deduce the implications of partial pre-image attacks on PoW-based blockchains. We present a generalized attack model that includes a modified mining probability of the attacker, which depends on other parameters than the compromised portion of the network (q). Additionally, we propose new benchmarks to assess the security level of PoW-based systems against the attack studied. This work formally confirms some experimental results and observations previously presented without explanation. Furthermore, our work predicts an exact difficulty value which, once reached, allows for compromising the system regardless of the proportion controlled by the attacker. Specifically, for any $q < 0.5$ (honest majority assumption still holds), the attack is formally proven profitable once the PoW mining puzzle difficulty reaches 56 leading zeros.

Funding

The research leading to these results has received funding from the National Research, Development and Innovation Office within the framework of the Artificial Intelligence National Laboratory Programme (MILAB), and from the national project TKP2021-NVA-09 implemented with the support provided by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund.

CRediT authorship contribution statement

Hamza Baniata: Conceived and designed the analysis - literature review, model design and validation, Collected the data, Performed the analysis, Wrote the paper - wrote the first version of the manuscript. Attila Kertesz: Performed the analysis - validated the mathematical equations, Wrote the paper - proof-read and enhanced the readability of the manuscript, Supervision, coordination.

Declaration of competing interest

The authors whose names are listed immediately below certify that they have NO affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

[1] S. Nakamoto, Bitcoin: a peer-to-peer electronic cash system, http://www.bitcoin.org/bitcoin.pdf, 2008. (Accessed 15 August 2023).
[2] M. Jakobsson, A. Juels, Proofs of work and bread pudding protocols, in: B. Preneel (Ed.), Secure Information Networks, Springer, Boston, MA, 1999, pp. 258–272, https://doi.org/10.1007/978-0-387-35568-9_18.
[3] F.I.P.S. Pub, Secure hash standard (shs), Fips Pub, 2012, 180 (4).
[4] R.C. Merkle, A digital signature based on a conventional encryption function, in: P. Carl (Ed.), Conference on the Theory and Application of Cryptographic Techniques, Springer, Berlin, Heidelberg, 1988, pp. 369–378, https://doi.org/10.1007/3-540-48184-2_32.
[5] H. Massias, X.S. Avila, J-J. Quisquater, Design of a secure timestamping service with minimal trust requirement, https://nakamotoinstitute.org/library/secure-timestamping-service, 1999. (Accessed 15 August 2023).
[6] H. Baniata, A. Anaqreh, A. Kertesz, PF-BTS: a privacy-aware fog-enhanced blockchain-assisted task scheduling, Inf. Process. Manag. 58 (1) (2021) 102393, https://doi.org/10.1016/j.ipm.2020.102393.
[7] N. Kshetri, J. Voas, Blockchain-enabled e-voting, IEEE Softw. 35 (4) (2018) 95–99, https://doi.org/10.1109/MS.2018.2801546.
[8] Z. Liao, S. Cheng, RVC: a reputation and voting based blockchain consensus mechanism for edge computing-enabled iot systems, J. Netw. Comput. Appl. 209 (2023) 103510, https://doi.org/10.1016/j.jnca.2022.103510.
[9] A. Hasselgren, K. Kralevska, D. Gligoroski, et al., Blockchain in healthcare and health sciences—a scoping review, Int. J. Med. Inform. 134 (2020) 104040, https://doi.org/10.1016/j.ijmedinf.2019.104040.
[10] X. Xiang, J. Cao, W. Fan, Decentralized authentication and access control protocol for blockchain-based e-health systems, J. Netw. Comput. Appl. 207 (2022) 103512, https://doi.org/10.1016/j.jnca.2022.103512.
[11] M. Samaniego, U. Jamsrandorj, R. Deters, Blockchain as a service for IoT, in: 2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), IEEE, 2016, pp. 433–436, https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData.2016.102.
[12] S. Wang, L. Ouyang, Y. Yuan, et al., Blockchain-enabled smart contracts: architecture, applications, and future trends, IEEE Trans. Syst. Man Cybern. Syst. 49 (11) (2019) 2266–2277, https://doi.org/10.1109/TSMC.2019.2895123.
[13] H. Baniata, A. Kertesz, Prifob: a privacy-aware fog-enhanced blockchain-based system for global accreditation and credential verification, J. Netw. Comput. Appl. 205 (2022) 103440, https://doi.org/10.1016/j.jnca.2022.103440.
[14] T. Jiang, H. Fang, H. Wang, Blockchain-based Internet of vehicles: distributed network architecture and performance analysis, IEEE Int. Things J. 6 (3) (2019) 4640–4649, https://doi.org/10.1109/JIOT.2018.2874398.
[15] K. Suzuki, D. Tonien, K. Kurosawa, et al., Birthday paradox for multi-collisions, in: M.S. Rhee, B. Lee (Eds.), International Conference on Information Security and Cryptology, Springer, Berlin, Heidelberg, 2006, pp. 29–40, https://doi.org/10.1007/11927587_5.
[16] M. Swan, Blockchain: Blueprint for a New Economy, O'Reilly Media, Inc., 2015.
[17] A. Kertesz, H. Baniata, Consistency analysis of distributed ledgers in fog-enhanced blockchains, in: R. Chaves, D.B. Heras, A. Ilic, et al. (Eds.), Euro-Par 2021: Parallel Processing Workshops, Springer, Cham, 2022, pp. 393–404, https://doi.org/10.1007/978-3-031-06156-1_31.
[18] Q. Wang, R. Li, S. Chen, et al., Formal security analysis on dbft protocol of neo, Distrib. Ledger Technol. 2 (1) (2023) 1–19, https://doi.org/10.1145/3568314.
[19] T. Porter, M. Gough, How to Cheat at VoIP Security, Syngress, 2011.
[20] L. Johnson, Security Controls Evaluation, Testing, and Assessment Handbook, Academic Press, Cambridge, Massachusetts, 2019.
[21] M.M.J. Stevens, Attacks on Hash Functions and Applications, Leiden University, 2012.
[22] Y. Sasaki, L. Wang, K. Aoki, Preimage attacks on 41-step SHA-256 and 46-step SHA-512, Cryptology ePrint Archive, 2009, https://ia.cr/2009/479.
[23] T. Isobe, K. Shibutani, Preimage attacks on reduced tiger and SHA-2, in: O. Dunkelman (Ed.), Fast Software Encryption. FSE 2009, in: Lecture Notes in Computer Science, vol. 5665, Springer, Berlin, Heidelberg, 2009, pp. 139–155, https://doi.org/10.1007/978-3-642-03317-9_9.
[24] J. Kelsey, B. Schneier, Second preimages on n-bit hash functions for much less than 2 n work, in: R. Cramer (Ed.), Advances in Cryptology - EUROCRYPT 2005, in: Lecture Notes in Computer Science, vol. 3494, Springer, Berlin, Heidelberg, 2005, pp. 474–490, https://doi.org/10.1007/11426639_28.
[25] D.M.A. Cortez, A.M. Sison, R.P. Medina, Cryptographic randomness test of the modified hashing function of SHA256 to address length extension attack, in: Proceedings of the 2020 8th International Conference on Communications and Broadband Networking, ACM, 2020, pp. 24–28, https://doi.org/10.1145/3390525.3390540.
[26] L. Bošnjak, J. Sreš, B. Brumen, Brute-force and dictionary attack on hashed real-world passwords, in: Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (Mipro), IEEE, 2018, pp. 1161–1166, https://doi.org/10.23919/MIPRO.2018.8400211.
[27] Huaqun Guo, Xingjie Yu, A survey on blockchain technology and its security, Blockchain Res. Appl. 3 (2) (2022) 100067, https://doi.org/10.1016/j.bcra.2022.100067.

[28] H. Gilbert, H. Handschuh, Security analysis of SHA-256 and sisters, in: M. Matsui, R.J. Zuccherato (Eds.), Selected Areas in Cryptography. SAC 2003, in: Lecture Notes in Computer Science, vol. 3006, Springer, Berlin, Heidelberg, 2003, pp. 175–193, https://doi.org/10.1007/978-3-540-24654-1_13.
[29] M. Juliato, C. Gebotys, SEU-resistant SHA-256 design for security in satellites, in: 2008 10th International Workshop on Signal Processing for Space Communications, IEEE, 2008, pp. 1–7, https://doi.org/10.1109/SPSC.2008.4686705.
[30] H. Handschub, H. Gilbert, Evaluation report security level of cryptography–SHA-256, https://www.cryptrec.go.jp/exreport/cryptrec-ex-1045-2001.pdf, 2002. (Accessed 15 August 2023).
[31] R. Benkoczi, D. Gaur, N. Nagy, et al., Quantum bitcoin mining, Entropy 24 (3) (2022) 323, https://doi.org/10.3390/e24030323.
[32] A.P. Ozisik, B.N. Levine, An explanation of Nakamoto's analysis of double-spend attacks, arXiv, preprint, arXiv:1701.03977, 2017.
[33] C. Grunspan, R. Pérez-Marco, Double spend races, Int. J. Theor. Appl. Finance 21 (08) (2018) 1850053, https://doi.org/10.1142/s021902491850053x.
[34] M. Rosenfeld, Analysis of hashrate-based double spending, arXiv, preprint, arXiv:1402.2009, 2014.
[35] J. Zhong, X. Lai, Preimage attacks on reduced DHA-256, Cryptology ePrint Archive 2009, https://ia.cr/2009/552.
[36] H.N. Bhonge, M.K. Ambat, B.R. Chandavarkar, An experimental evaluation of SHA-512 for different modes of operation, in: Proceedings of the 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), IEEE, 2020, pp. 1–6, https://doi.org/10.1109/ICCCNT49239.2020.9225559.
[37] G. Rowe, Has SHA256 been broken by Treadwell Stanton Dupont?, https://crypto.stackexchange.com/questions/74251/has-sha256-been-broken-by-treadwell-stanton-dupont, 2019. (Accessed 15 August 2023).
[38] J.J. Kearney, C.A. Perez-Delgado, Vulnerability of blockchain technologies to quantum attacks, Array 10 (2021) 100065, https://doi.org/10.1016/j.array.2021.100065.
[39] R. Preston, Applying Grover's algorithm to hash functions: a software perspective, arXiv, preprint, arXiv:2202.10982, 2022.
[40] List of mining pools http://miningpoolstats.stream. (Accessed 15 August 2023).
[41] J. Heusser, Sat solving-an alternative to brute force bitcoin mining, Technical report, 2013, https://jheusser.github.io/2013/02/03/satcoin.html.
[42] C. Dobraunig, M. Eichlseder, F. Mendel, Security evaluation of SHA-224, SHA-512/224, and SHA-512/256, Institute for Applied Information Processing and Communications, Graz University of Technology, 2015.
[43] N. Manthey, J. Heusser, Satcoin–Bitcoin mining via sat, SAT COMPETITION 2018, 2018, p. 67.
[44] W. Gong, X. Zhou, A survey of sat solver, in: Proceedings of the AIP Conference Proceedings, vol. 1836, AIP Publishing LLC, 2017, 020059, https://doi.org/10.1063/1.4981999.
[45] I. Mironov, L. Zhang, Applications of SAT solvers to cryptanalysis of hash functions, in: A. Biere, C.P. Gomes (Eds.), Theory and Applications of Satisfiability Testing - SAT 2006, in: Lecture Notes in Computer Science, vol. 4121, Springer, Berlin, Heidelberg, 2006, pp. 102–115, https://doi.org/10.1007/11814948_13.
[46] F. Massacci, Using Walk-SAT and Rel-SAT for cryptographic key search, in: Proceedings of the 16th International Joint Conference on Artifical Intelligence (IJCAI), vol. 1, 1999, pp. 290–295.
[47] B.W. Bloom, Sat solver attacks on cubehash, Technical report, Citeseer, 2010.
[48] R. Durrer, J. Laukenmann, The oscillating universe: an alternative to inflation, Class. Quantum Gravity 13 (5) (1996) 1069–1088, https://doi.org/10.1088/0264-9381/13/5/021.
[49] N. Tovanich, N. Soulié, N. Heulot, et al., The evolution of mining pools and miners' behaviors in the bitcoin blockchain, IEEE Trans. Netw. Serv. Manag. 19 (3) (2022) 3633–3644, https://doi.org/10.1109/TNSM.2022.3159004.
[50] H.T. Heinonen, A. Semenov, Recycling hashes from reversible bitcoin mining to seed pseudorandom number generators, in: K. Lee, L. Zhang (Eds.), Blockchain - ICBC 2021, in: Lecture Notes in Computer Science, vol. 12991, Springer, Cham, 2021, pp. 103–117, https://doi.org/10.1007/978-3-030-96527-3_7.
[51] V. Gheorghiu, M. Mosca, Benchmarking the quantum cryptanalysis of symmetric, public-key and hash-based cryptographic schemes, arXiv, preprint, arXiv:1902.02332, 2019.