Received 21 December 2022, accepted 30 December 2022, date of publication 5 January 2023, date of current version 10 January 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3234322

SURVEY

Approaches to Overpower Proof-of-Work Blockchains Despite Minority

HAMZA BANIATA AND ATTILA KERTESZ
Department of Software Engineering, University of Szeged, 6720 Szeged, Hungary

Corresponding author: Hamza Baniata (baniatah@inf.u-szeged.hu)

This work was supported in part by the National Research, Development and Innovation Office through the Artificial Intelligence National Laboratory Programme; in part by the National Project through the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund under Grant TKP2021-NVA-09 and Grant TKP2021-NVA; and in part by the University of Szeged Open Access Fund under Grant 6023.

ABSTRACT Blockchain (BC) technology has been established in 2009 by Nakamoto, using the Proof-of-Work (PoW) to reach consensus in public permissionless networks (Praveen et al., 2020). Since then, several consensus algorithms were proposed to provide equal (or higher) levels of security, democracy, and scalability, yet with lower levels of energy consumption. However, Nakamoto's model (a.k.a. Bitcoin) still dominates as the most trusted model in the described sittings since alternative solutions might provide lower energy consumption and higher scalability, but they would always require deviating the system towards unrecommended centralization or lower levels of security. That is, Nakamoto's model claims to tolerate (up to) $< 50\%$ of the network being controlled by a dishonest party (minority), which cannot be realized in alternative solutions without sacrificing the full decentralization property. In this paper, we investigate this tolerance claim, and we review several approaches that can be used to undermine/overpower PoW-based BCs, even with minority. We discuss those BCs taking Bitcoin as a representative application, where needed. However, the presented approaches can be applied in any PoW-based BC. Specifically, we technically discuss how a dishonest miner in minority, can take over the network using improved Brute-forcing, AI-assisted mining, Quantum Computing, Sharding, Partial Pre-imaging, Selfish mining, among other approaches. Our review serves as a needed collective technical reference (concluding more than 100 references), for practitioners and researchers, who either seek a reliable security implementation of PoW-based BC applications, or seek a comparison of PoW-based, against other BCs, in terms of adversary tolerance.

INDEX TERMS Blockchain, proof-of-work, security, quantum computing, sharding, machine learning.

I. INTRODUCTION

Bitcoin [2] is the first solution that addressed several open issues of a wanting online distributed cryptocurrency system. The most important aspects that delayed the proposal of such a reliable system are the security, privacy, and full-decentralization. That is, the success of a given online distributed cryptocurrency solution is attributed by the reliable provision of high security and privacy measures, without using a Trusted Third Party (TTP). Several techniques were deployed within the proposal of Bitcoin to comprehensively address those issues, including Proofs-of-Work [3], robust hashing functions [4], Merkle Trees [5], and distributed timestamps [6]. The combination of those techniques resulted in the emergence of the so-called Blockchain (BC) technology.

As we can see, Bitcoin has been increasingly used and adopted for more than 13 years now (by the time of writing). Not only Bitcoin, but the BC technology in its generality was also used in a wide variety of applications, including distributed e-voting [7], [8], e-Health [9], [10], [11], IoT [12], smart contracts [13] and IoV [14].

The associate editor coordinating the review of this manuscript and approving it for publication was Yiming Tang.

Upon the proposal of Bitcoin, several main security-related assumptions were made depending on the postulated high security of the utilized cryptography methods. As long as these main assumptions were not violated, the claimed high security of PoW-based BCs shall remain dependable. One of these assumptions is that PoW-based BCs will always be secure against a dishonest miner, as long as its computational power proportion is less than 50% of the total computational power of the network.

In this paper, we discuss probable alternative PoW mining methods/attacks, other than those presented in the original paper [2]. We focus on cases where participation/profit probability is higher than claimed in [2] due to their reliance on factors other than the controlled computational power. Specifically, we discuss how to improve the classical Brute-force mining by manipulating the randomness level of the searched domain and by training and utilizing AI models. Additionally, we dive into the intriguing challenge of Quantum Computing [15] methods against PoW-based BC's security, and the applicability of state-of-the-art Sharding [16] approaches. Accordingly, we set main benchmarks and requirements of post-quantum and secure sharding protocols for PoW-based BCs, and we perform experiments to demonstrate the scalability-security trade-off. We further discuss other approaches to undermine PoW-based BCs such as partial pre-image attacks using SAT solvers and selfish mining in pooled BCs. Throughout the paper, we present several related open issues, and we systematically discuss novel, and previously proposed, security benchmarks.

Motivated by several approaches presented in the literature, we found no previous work that accumulated all available methods to compromise PoW-based BCs despite controlling a minority of the network's computational power. To the best of our knowledge, this is the first technical review that provides this. Our contributions can be summarized in the following points:

- We technically discuss the potential of eight different approaches that can be used to compromise permissionless PoW-based BCs while controlling a minority of the total computation network power.
- We experimentally evaluate the security of PoW-based BCs in different cases, including enhanced Brute-forcing using the Collatz Conjuncture and Sharding. Accordingly, a number of open research questions were provided.

The remainder of this paper is structured as follows: Section II provides necessary background of BC, SHA-256, and PoW mining. Sections III and IV describe methods to improve the efficiency of classical PoW mining using the Collatz conjecture and Machine Learning, respectively. Section V reviews state-of-the-art advances for utilizing Quantum Computing to control PoW-based BCs. Section VI discusses Sharding concepts in public-permissionless PoW-based BCs, where main research questions and experiments demonstrating the scalability-security trade-off were presented. Section VII presents four other mining approaches that were found more profitable than classical mining. Finally, Section VIII concludes our work.

II. BACKGROUND

A. BLOCKCHAIN

A BC-based system is characterized by its infrastructure, data structures, networking model, and Consensus Algorithm (CA). The 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 is described according to the application of the system. For example, transactions (TXs) are submitted by end users to the BC 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 [17].

As BC nodes form a distributed system, those nodes exchange data through a P2P network and communicate by message passing via directly connected links. BC 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 ms.

Let $K = k_1, k_2, \ldots, k_f$ be the set of adversary nodes in G and $M = j_1, j_2, \ldots, j_{N-f}$ be the set of honest nodes, Equation 1 is usually assumed valid for BC networks.

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

Let the network's total computational power be T. The attacker, then, controls a portion q out 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 out 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 BC-based system must operate a CA in order to maintain the consistency of its DL [18]. As tens of CAs were proposed in the literature, a CA is usually considered *valid*, if it was 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 either q with T or f with N. As long as Ψ holds, the BC is considered secure [19].

For example, the Delegated Byzantine Fault Tolerance (dBFT) algorithm [20] was claimed to be secure as long as $f <= \frac{N-1}{3}$, while PoW [2], was claimed to be secure as long as Condition 4 holds.

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

B. 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 [21]. 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 [22].

As an encryption method, hashing functions have been studied and improved over the years to guarantee the highest possible security. Main categories of attacks on hashing functions include Collision attacks [23], Preimage attacks [24], Second-Preimage attacks [25], [26], Length extension attack [27], and Brute-force attacks [28].

Simply put, a Brute-force attack implies sequential, or random, testing of a wide range of inputs, until finding the correct or desired output. Performing such attack on hashes is considered the easiest to implement but the hardest in terms of cost. There are several ways to use such attacks as it can be used to run any of the other described attacks. For example, if an attacker needs to know a message m that was 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, main properties of a secure and reliable hashing function [29] include Fixed size of output, Preimage resistant, Second Preimage resistant, Collision resistant, and Random distribution of outputs.

SHA-256 [4] was proven secure against all known attacks on hashing functions, except for, trivially, the Brute-force attack [30], [31]. Specifically, the lower bound complexity of algorithmic Preimage or Second-Preimage 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} [32]. As such bounds make the SHA-256 compliant with all above mentioned properties, it has been deployed in Bitcoin as the main utilized hashing function.

C. MINING IN PROOF-OF-WORK BLOCKCHAIN

The mining problem in PoW-based BCs (as described in the Bitcoin white paper [2]) is defined as finding a nonce which, 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 pre-defined number of leading Zeros r in the produced hash. This produced hash, together with the block of data is called a valid block [33].

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 [34] may need further analysis and modifications.

As it is thus far argued that the only way to mine is such PoW-based BCs is by conducting a Brute-force attack, miners should be confident a miner that provides a correct PoW, has worked for a 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.

III. IMPROVED BRUTE-FORCE MINING

The original Bitcoin calculations primarily rely on p and q values for determining the probability of a successful attack (on a block at depth z). That is, hardening the puzzle is assumed to have no effect on these probabilities. We present detailed explanation and discussion of mining probabilities in Appendix VIII-B. Accordingly, it is assumed that hardening the puzzle difficulty would equally affect Q and E and, thus, Equation 16 should always hold regardless of the value Ω.

As honest miners search sequentially for a nonce that fulfils the puzzle difficulty, we propose that the attacker searches randomly for such a nonce. Assuming the outputs of the hash function and the inputs of the attacker were *equivalently* Truly Random [35], an adversary miner should theoretically have higher chances to find a puzzle solution before an honest miner does.

We propose Algorithm 3, representing a dummy attack procedure, to check if this assumption is correct. Similarly to the Finney Attack [36], Algorithm 3 implies the attacker prepares TX_2 that consists incorrect data. The attacker then waits for a new block (B_z) to arrive. Once B_z is received, a transaction within (say TX_1) is modified so that TX_2 becomes correct. The Collatz_Update function (Algorithm 1) exploits the Collatz Conjecture [37], [38] to guarantee a Truly Random search for a correct $nonce_A$. An attacker that uses our proposed algorithm might have an advantage over honest miners, in finding a correct nonce if, and only if, the following conditions were fulfilled:

1) The first value of $nonce_A$ was accurately selected for both B_z and B_{z+1}, and
2) The randomness uniformity of the Collatz Conjecture maps to the randomness uniformity of SHA-256 output range (i.e. relatively more than that of a sequential search).

More sophisticated randomization approaches using, specifically, the Collatz Conjecture were proposed in the literature. For example, several pseudo-random number generators that could pass the NIST Test Suite were proposed in [39]. Out of these generators, one was proven to generate uniformly distributed output. On the other hand, the randomness margin of the SHA-256 hashing function was reported to be 0.734 in [40], which raises Research Question 1, whose answer shall be the perfect candidate to substitute our randomization approach.

***RQ** 1:* What is the randomization approach, whose outputs best map to the SHA-256 randomness uniformity?

We implemented Algorithms 3, 1, and 2 using Python3, on top of which we built a simulator that consecutively generates 1,000 Bitcoin blocks for each given pair of unique parameterization (r, q). We tested all $r \in [0, 1, \ldots, 14]$ and all $q \in [0, 0.1, \ldots, 0.45]$, resulting in 675 pairs and a total of 675,000 attempts to attack the simulated Bitcoin BC. Accordingly, we calculated the experimental attack success rate using the proposed algorithms by dividing the number of successful attempts by the total number of attack attempts, per unique pair. We ran our code on the Google Cloud Platform using a VM of type c2-standard-4 (4 Intel Cascade Lake vCPUs clocked at 3.8GHz with 16GB of RAM), running Ubuntu 18.04 LTS. Our code is publicly available at Github.[1] Part of the results of our experiments are presented in Figure 1 and detailed in Figure 2. In the table, we further added the average success rates for $r \geq 8$ (since this was the least difficulty used since the launch of Bitcoin). We then compared the computed average with its relevant q value to check if the proposed approach provided higher attack success rate than theoretical expectations.

It is easily notable from the results that Algorithm 3 provides less success attack rate, than the probability computed by the original Bitcoin calculations. This confirms that Algorithm 3 is not a suitable candidate to attack a PoW network in its current form. Our results also confirm that Equation 16 holds regardless of the value of Ω even when the first assumption is violated (i.e. in our case: partially). On the other hand, increasing the difficulty does indeed lower the attack success probability, using Algorithm 3 in its current form, equally to the decrement in honest mining probability as we obtained a Poisson Distribution.

[1] github.com/HamzaBaniata/Collatz

Algorithm 1 Collatz_Update

```
Input: $nonce_A$, tried_ONE_already
Output: $nonce_A$
if $nonce_A$ is even then
    $nonce_A = nonce_A / 2$
else
    $nonce_A = (3 \times nonce_A) + 1$
if $nonce_A == 1$ and tried_ONE_already then
    $nonce_A$ = random.randint(5, maximum_int)
return $nonce_A$
```

Algorithm 2 Forge_block

```
Input: $TX$, $B_z$
Output: $B_z$
$nonce_A$ = random.randint(5, maximum_int)
tried_ONE_already = False
while $\neg h(nonce_{B_z} \wedge TX) \Rightarrow valid\ B_z$ do
    $nonce_{B_z}$ = Collatz_Update ($nonce_{B_z}$, ted_ONE_already)
    if $nonce_{B_z} == 1$ then
        tried_ONE_already == True
return $B_z$
```

Algorithm 3 Attack

```
Input: $B_z$ = Most recently confirmed valid block
Output: $B_{z+1}$ = Forged valid block
Start:
Construct forged $TX_2$
Modify on $TX_1$ in $B_z$ ($\Rightarrow TX_1'$) to make $TX_2$ valid
$B_z'$ = Forge_block ($TX_1'$, $B_z$)
$B_{z+1}$ = Forge_block ($TX_2$, $B_{z+1}$)
return $B_{z+1}$
```

If Algorithm 1 was optimized by answering RQ1, an improved Brute-force attack can be successful on PoW-based BCs. As a result, honest nodes would maintain lesser probability to find a puzzle solution through time (due to the difficulty increment), while the attacker's success probability Q needs to be redefined. The work of Bhattacharjee et al. [41] would be a good start to find an answer for RQ1.

IV. MACHINE LEARNING

Machine Learning (ML) is a subfield of Artificial Intelligence (AI), which is broadly defined as the capability of a machine to imitate intelligent human behavior [42]. This capability is realized using different types of algorithms that are usually categorized into Classification and Regression algorithms. The main difference between Regression and Classification algorithms is that Regression algorithms are used to predict continuous values such as price, salary, age, while Classification algorithms are used to predict/classify discrete values such as Male or Female, True or False, Spam or Not Spam [43]. Nevertheless, the general aim of both

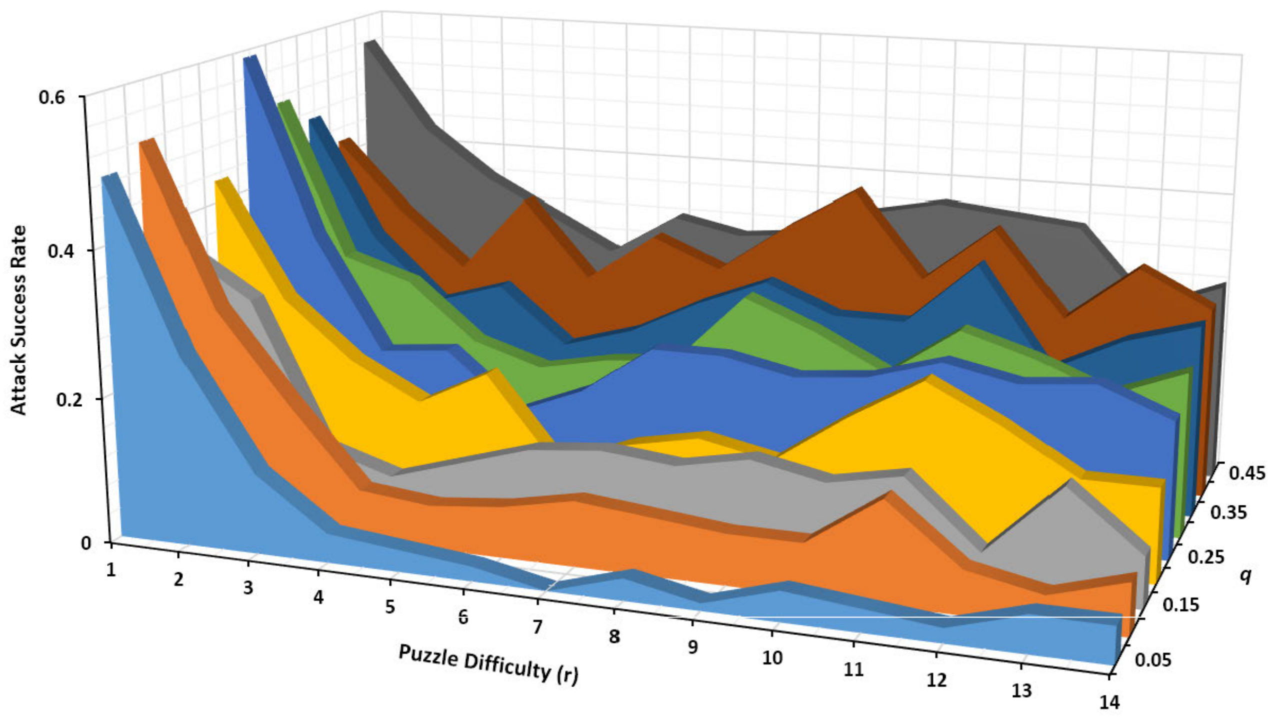


FIGURE 1. **Attack success rates when using Algorithm 3 with different parameterization of puzzle difficulty and portion controlled by the attacker (q).**

categories is to precisely capture the relations between inputs and outputs, so that the prediction accuracy is increased with more training.

Utilizing ML for providing an alternative, or at least more efficient PoW mining, is usually thought of as a successful attack on the used hashing function in particular. On the other hand, the randomness margin of a given hash function is defined as the ratio of number of random rounds to the total number of rounds [40].

SHA-256, which is used in most PoW-based BCs including Bitcoin, was reported to have a randomness margin of 73.4% [40]. This is considered a very good level, compared to other functions, since collisions are very hard to find in this case. Accordingly, it is used because it relatively provides high levels of security. However, this randomness level means practically that up to 26.6% of any produced hash string is expected to be non-random (i.e. is affected by the input). Specifically, we identified this as the expected maximum improvement (henceforth notated as θ_{max}), an ML-based PoW mining method can provide.

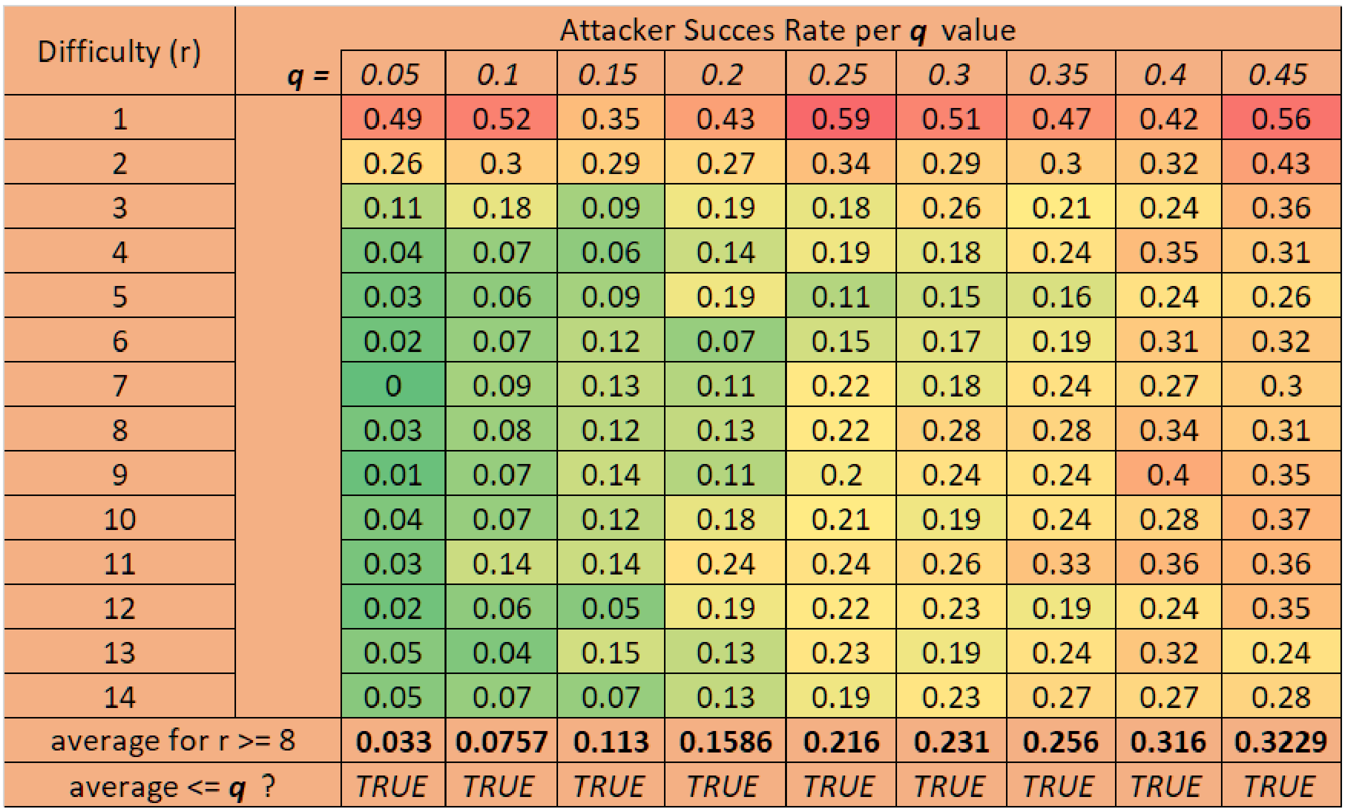
We have recently investigated this assumption in [44] and proposed a revolutionary approach where ML is used to improve the classical PoW mining efficiency. Since ML relies basically on learning the relationships between different inputs and outputs, then using ***only*** an ML model to predict a PoW puzzle solution would be feasible if, and only if, the following two conditions apply:

1) There are detectable relations between inputs and outputs of the utilized hashing function that the ML model can learn.
2) $\theta > q$.

The first condition implies that if there were no detectable relations between inputs and outputs then the model would always predict fully random values, which contradicts our prediction requirement. Furthermore, if there were obvious detectable relations then the utilized hashing function is not secure enough to be deployed in PoW-based BCs in the first place. However, in the case of SHA-256, the reported high randomness margin implies there is a relation percentage of up to θ_{max} between each given input and its output.

The second condition is self-explanatory. If the probability of finding PoW puzzle solutions using classical mining is higher than that using ***only*** a trained ML model, then miners will always use the classical mining as it would provide higher profits. Assuming we have an optimally trained ML model,[2] this model should be able to precisely predict up to θ_{max} nonce values for (sufficiently big) given tested data. We can also describe it differently as an optimally trained model would predict nonce values closer (to correct values) than initial search values of classical miners, in θ_{max} of tests. Since it is practically impossible to train such a model, then

[2]This is hypothetical since we would need large enough training data, such that the data would represent all probable inputs (i.e. infinite possibilities) and outputs (i.e. 2^{256} possibilities).



Difficulty (r)	Attacker Succes Rate per ***q*** value									
	q =	*0.05*	*0.1*	*0.15*	*0.2*	*0.25*	*0.3*	*0.35*	*0.4*	*0.45*
1		0.49	0.52	0.35	0.43	0.59	0.51	0.47	0.42	0.56
2		0.26	0.3	0.29	0.27	0.34	0.29	0.3	0.32	0.43
3		0.11	0.18	0.09	0.19	0.18	0.26	0.21	0.24	0.36
4		0.04	0.07	0.06	0.14	0.19	0.18	0.24	0.35	0.31
5		0.03	0.06	0.09	0.19	0.11	0.15	0.16	0.24	0.26
6		0.02	0.07	0.12	0.07	0.15	0.17	0.19	0.31	0.32
7		0	0.09	0.13	0.11	0.22	0.18	0.24	0.27	0.3
8		0.03	0.08	0.12	0.13	0.22	0.28	0.28	0.34	0.31
9		0.01	0.07	0.14	0.11	0.2	0.24	0.24	0.4	0.35
10		0.04	0.07	0.12	0.18	0.21	0.19	0.24	0.28	0.37
11		0.03	0.14	0.14	0.24	0.24	0.26	0.33	0.36	0.36
12		0.02	0.06	0.05	0.19	0.22	0.23	0.19	0.24	0.35
13		0.05	0.04	0.15	0.13	0.23	0.19	0.24	0.32	0.24
14		0.05	0.07	0.07	0.13	0.19	0.23	0.27	0.27	0.28
average for r >= 8		**0.033**	**0.0757**	**0.113**	**0.1586**	**0.216**	**0.231**	**0.256**	**0.316**	**0.3229**
average <= ***q*** ?		*TRUE*	*TRUE*	*TRUE*	*TRUE*	*TRUE*	*TRUE*	*TRUE*	*TRUE*	*TRUE*

FIGURE 2. Attack success rates when using Algorithm 3 with different parameterization of puzzle difficulty and portion controlled by the attacker (q). The color scale (Green-Yellow-Red) is correlated with the attack success rate value on a hypothetical fuzzy spectrum (Low-Medium-High).

it is always expected that θ is bounded as in Inequality 5.

$$0 \leq \theta < \theta_{max} \tag{5}$$

Nonce cannot be found immediately using only trained ML models, due to the high randomness level of the hash function. Thus, we targeted the improvement of the classical mining by training a model, taking its predictions as *initial* nonce values, and then start mining classically starting from this predicted nonce. The predicted nonce values should then be closer to correct values than the initial nonce value a classical miner would start with (i.e. 0×0). Such improvement implies that a miner would search a narrower domain until it finds a correct nonce, resulting in a relatively faster, and more efficient, block production process compared to classical mining.

We trained regression ML models, including linear and polynomial -based models, using randomly chosen 500k+ confirmed Bitcoin blocks, and we tested them using the remaining 300k+ confirmed blocks (total is about 800k+ confirmed blocks since Bitcoin was launched in 2009). We proposed Equation 6 to estimate the probability of solving PoW puzzles using our approach (for a given ML model with θ, check Appendix VIII-B for more detials on the used notations).

$$Q = q + (\lambda \, . \, \theta) \tag{6}$$

We have then experimentally evaluated our approach by setting up a Bitcoin miner predicting nonce values for the the testing data using the trained models. Consequently, we calculated the absolute differences between all predicted and actual values (loops), and then counted the number of loops that are less than the original nonce values. Finally, we calculated the total experimental success score by finding the ratio of: loops less than original values, to the total number of tested blocks.

The results we obtained validated our proposed Equation 6. That is, Equation 6 presents a theoretical probability estimation for a given model with a given controlled proportion, while the computed success scores present experimental statistics for the tested model. Our evaluation results are provided in Table 1, which clearly shows that the differences between theoretical estimations to experimental success scores were negligible.

Subconsequently, we computed the maximum tolerable q by a PoW-based BC system. Specifically, we concluded that $q < 0.354$ must always hold, such that even with a model providing θ_{max}, dishonest miners would still have less than 50% participation probability. In other words, the presented ML-assisted mining approach allows to overpower the system despite controlling a minority of $q \geq 0.354$.

V. QUANTUM COMPUTING

In addition to SHA-256 for securing the ledger against alteration attacks, asymmetric cryptography is used (e.g. the ECDSA in Bitcoin) to secure users' data and TXs [45].

TABLE 1. Results for ML-based improved PoW mining.

	Training Set	Testing Set	θ	$Q(q=0.5)$	$SuccessScore$
SGDRegressor	300	200	18.8%	68.8%	**68.5%**
PolynomialFeatures			20.5%	70.5%	**69.5%**
SGDRegressor	500k	300k	18.8%	68.8%	**64.3%**
PolynomialFeatures			20.5%	70.5%	**70.5%**

Both security measures were shown to be vulnerable against Quantum Computing (QC), which offers fundamentally different solutions to computational problems and enables more efficient problem-solving than what is possible with classical computations [15]. For example, a quantum computer with a clock speed of 100 MHz could factor a 129-bit RSA cipher in a few seconds [46]. QC utilizes the fundamentals of Quantum physics and its definitions and applications to provide more efficient and fast computational paradigms. To directly hit our point, fine-tuned and well equipped QC machines (or Quantum Computers) can be used to decipher secrets, originally encrypted using an asymmetric method, or even to find hash collisions much faster than classical computing methods. Detailed explanation of QC foundations, algorithms, gates, qubits, and countermeasures can be found in [47] and [48].

As argued in [47], it is only a matter of "when" and not "if" QC will become functional as a threat to BC technology in general. Sattath [49] further affirmed that a crucial argument in the analysis of PoW-based BCs security breaks down when QC-based mining is performed. For example, they stated that the chances of a successful QC-based attack, i.e. finding a block using QC, grow quadratically with the number of Grover iterations [50] applied. During the past decade, big IT companies have been spending much time and funds to provide both reliable QC services and QC-resistant cryptographic methods [51]. Examples include IBM [52], Intel [53], Microsoft [54], and Google [55].

Several previous works have investigated the potential of QC attacking, specifically, the PoW mining problem. For instance, Tessler and Byrnes [56] and Aggarwal et al. [57] studied and explained how the Grover's Algorithm [50] can be used to perform PoW mining. They also well explained QC attacks and they discussed the probability and the profitability of a successful attack. The authors have also discussed using Shor's algorithm [58] to find the private keys from a public key using QC. Specifically, there are a total of 64 rounds of hashing in the SHA256 protocol and each round can be done using an optimized circuit with 683 Toffoli quantum gates [59] (i.e. Controlled Controlled NOT (CCN) gates [46]). In [57], specifically, the number of physical qubits needed to perform PoW mining was defined as a function of overhead in physical qubits, incurred due to quantum error correction, and of the mining difficulty and gate error rate. In [45] and [60], on the other hand, a Quantum attacks resistant protocol, called Commit-Delay-Reveal, was discussed, which allows users to securely move their funds from non-quantum-resistant outputs to those adhering to a quantum-resistant Digital Signature (DS) scheme. In [49], a mechanism to prevent from quantum mining is proposed, while Santha et al. [61] proved a weak variant of the second conjecture presented in [49].

On the contrary of the above discussed works, Jogenfors [62] proposed a revolutionary QC-based cryptocurrency system. The economic aspects of this system are similar to Bitcoin's, yet it deploys QC for mining instead of PCs and ASICs for classical methods. Perhaps such solution is still not applicable as Quantum Computers are not yet commercialized. However, such solution is trivially expected to be the next cryptocurrency module once Quantum computers are affordable and available for public use. Additionally, such solution provides QC-attacks resistance methods by design, a feature that is not currently available in cryptocurrencies. In [63], several quantum payment schemes were introduced, for implementing prudent contracts—a non-trivial subset of the functionality that a network such as Ethereum provides. The proposed schemes utilize the futuristic affordability vision of Quantum computers for novel quantum-resistant cryptocurrencies and smart contracts.

With reference to the major assumptions discussed in Appendix VIII-B, QC utilization violates only the second assumption. That is, QC algorithms do search in the whole space of probable solutions, but use different resources to do so, including the physical superposition phenomena and searching the space reversibly. Specifically, we researched in the literature for answers responding to RQ 2.

***RQ** 2:* When and how can a Quantum Attack be launched on PoW-based BCs?

The "When" part of RQ 2 was recently responded to by Nerem and Gaur [64]. The authors investigated the speed and energy efficiency a quantum computer needs to offer an advantage over classical mining while the third assumption is preserved. To do so, an optimal mining procedure was presented, which was used to show the outperformance of an attacker compared to a classical computer in efficiency (cost per block). A condition, under which this attack can be profitable, is mathematically described[3] by the inequality $Q < Crb$, where Q is the cost of a Grover iteration, C is the

[3]We kindly ask the Reader not to confuse these notations with our current work's notations, as we used similar ones presented in reference [64].

cost of a classical hash, r is the quantum miner's speed in Grover iterations per second, and b is a factor that attains its maximum if the quantum miner uses the proposed optimal mining procedure. The condition provided in [64] presents a threshold benchmark for which quantum mining, and the known security risks associated with it, may arise as a major security threat.

The "How" part of RQ 2 was responded to by several previous works, including specific requirements for each approach. We found that two main types of QC-based attacks can be launched against PoW-based BCs. The first type uses the Grover's Algorithm and it mainly aims at attacking the hash function security by finding collisions. Any type of attacks on SHA-256 (see Subsection II-B) can be launched using this algorithm as follows:

- The number of attacker's Qubits and/or Quantum Computers must be sufficient to rival the probability of finding the next block using classical Brute-force [56]. In other words, the third assumption needs to be violated as well.
- A Quantum-based method for PoW mining can be executed in time $O(\sqrt{2}^{\Omega})$, which is much less than classical Brute-force settings [49]. However, it was stated that the size of the quantum computer needs to be $\approx 10^4$ qubits in order to provide such attack efficiency.

The second type uses Shor's Algorithm for reusing addresses, double spending attacks, or live TX hijacking. Aside from the recommendation of most works to use different addresses and keys for each TX, several quantum safe alternatives, for the ECDSA signature scheme used in Bitcoin, were proposed [65]. The most promising proposed approaches include the Winternitz One-Time Signature Scheme (W-OTS+) [66] and the SPHINCS+ [67] method. Those, two methods, specifically, were detailed in [68] and compared to ECDSA and RSA schemes in [69]. For the second type of QC attacks (i.e. using Shor's algorithm), we found the following measures to be the most relevant responses to RQ 2 (each with its source reference):

- 1500 qubits are required and 6×10^9 one-qubit additions are needed (Each one-qubit addition takes 9 quantum gates) in order to successfully attack PoW-based BCs [70].
- 2330 qubits are needed and 1.26×10^{11} Toffoli gate operations are required in order to successfully attack PoW-based BCs [71].
- for 10GHz clock speed and error rate of 10^{-5}, Bitcoin signatures can be cracked in 30 minutes using 485550 qubits [57], according to which, signatures are expected to be easily broken, in time less than the Bitcoin block time, by the year 2027.
- The number of qubits required for the Bitcoin's case is roughly 1536 qubits [70], [72].

As can be noticed in both types of QC-based attacks, thousands of qubits are required to launch a profitable PoW mining, or to hijack encrypted TXs. As of May 2022, the leading giant in QC; IBM, announced its biggest ever Quantum computer that consists of 433 qubits [73], which still does not violate the third PoW assumption. To summarize, although QC-based attacks violate the second PoW assumption, currently available quantum computers cannot successfully attack PoW-based BCs unless they also violate the third assumption.

VI. SHARDING

A sub-graph of G is any graph $G' = (V', \epsilon', w')$, such that $V' \subseteq V$ and $\epsilon' \subseteq \epsilon$. G' is also undirected and weighted as it inherits the properties of the original graph, yet it is not necessarily connected.

Sharding is a type of database partitioning technique that separates a very large database into much smaller, faster, more easily managed parts called data shards [74]. Technically, sharding is a synonym for horizontal partitioning, which makes a large database more manageable and efficient. Following the notations detailed in Subsection II-A, the key idea of BC sharding is to partition V (which are formed initially as one giant shard) into a set $\Gamma = \{Shard_1, Shard_2, ..Shard_\ell\}$. Each $Shard_i \in \Gamma$ consists of $N_{Shard_i} < N$ nodes until condition 7 is satisfied. Each shard processes a disjoint set of TXs, yet all shards utilize the same CA, leading to increased overall system throughput. As described in [75], ℓ grows linearly with both T and N. Denoting the number of adversary nodes in $Shard_i$ as $f_{Shard_i} <= f$, a sharding protocol outputs a set Γ leading to State 8.

$$\sum_{i=1}^{\ell} N_{Shard_i} = N \tag{7}$$

$$\sum_{i=1}^{\ell} f_{Shard_i} = f \tag{8}$$

Typically, a solution for a problem in a permissionless distributed system, such as sharding PoW-based BCs, should consider three main, strongly-connected factors affecting the optimization of that solution. Those three factors are decentralization, scalability, and security. Although privacy issues/discussions are technically different than security issues/discussions, privacy preservation is usually considered as a subdomain within the security discussion. Accordingly, a suitable sharding protocol for PoW-based BCs can be characterized by answering the following Research Questions:

RQ *3 (Decentralization):* Which system element should be responsible for computing the set Γ?

RQ *4 (Scalability):* What is the suitable approach to compute the set Γ?

RQ *5 (Security and Privacy):* Must Ψ be maintained in each shard? If Yes, then how?

For demonstration purposes, we will discuss the sharding problem in the following subsections for, specifically, Bitcoin. However, our discussions can be followed for any similar permissionless PoW-based BC.

1) DECENTRALIZATION

The decentralization level, provided in the permissionless public model of Bitcoin, is foundational for gaining trust and reliability [76]. The more entities participating in the protocol, the more trusted the system becomes. To clarify, a sharded version of any network is obviously less decentralized than its non-sharded network. Since ℓ (number of shards) is by definition less than N (number of nodes), the number of entities maintaining the global view of the sharded network's ledger is much less than those which maintain the ledger in the network when it is not sharded.

On the other hand, since each shard processes different TXs than those processed by other shards, the number of validations and confirmations granted per TX is much less than confirmations granted per TX in the non-sharded version of the network. In other words, trust in the finality of a given TX in a sharded network is less than the trust in the finality of the same TX in the same network that is non-sharded.

Furthermore, since a permissionless public BC implies that any node participating in the network might behave dishonestly [1], a foundational question for sharding Bitcoin is RQ 3. We have few answers to choose from:

1) A Trusted Third Party (TTP): which implies the inclusion of a special node(s) that has advanced access permissions to private data of the miners, including their computational capacity, IP-addresses, neighbors, bandwidth, etc. This imposes a critical privacy concern regarding the capability of the TTP to build and misuse global information about the network. Furthermore, a dishonest TTP is assumed possible in Bitcoin, thus was (and still) not used.
2) An Elected Leader: Leader election problem in fully distributed systems was studied in several previous works. Although it is hard to address, once addressed in public permissionless BCs either privacy violations, or very high complexities are usually expected [77]. Even if agreed-on to be used, incentivization mechanisms need to be implemented, and high complexity issues must be solved.
3) Dynamic Selection of Leader Shards: this is a hybrid version of the the above two methods. Instead of electing a single leader to compute set Γ, a whole shard is elected which utilizes a CA to agree on a sharded version. Although this option solves the problems faced by a TTP utilization, it adds one extra layer of complexity to the single leader election problem (i.e. complexity associated with the CA). Furthermore, it is obvious that in such settings, a small group in V controls the logical communications amongst all V elements. For this, the majority of nodes within a shard must be honest or the system security would crash (discussed later).

It is noticeable by now how sharding indeed deviates the Bitcoin network towards centralization. Nevertheless, with adequate tuning of the sharding parameterization, this should not be a big concern knowing that there are tens of thousands of nodes. However, high scalability gained from sharding the network must be convincing for lowering the decentralization level. Assuming that there is a sharding protocol that preserves equivalent level of security with high level of scalability, the leader shard option seems to be the most appropriate.

2) SCALABILITY

A sharding protocol is theoretically expected to enhance a BC-based solution in terms of scalability [78]. That is, the scalability of the BC-based solution is usually benchmarked by the overall throughput of the system while increasing N. Since processing tasks is distributed among shards, new TXs and blocks can be processed in parallel. Accordingly, increasing the number of shards (ℓ) should increase the overall throughput of the system regardless of the approach of nodes distribution among shards. Nevertheless, ℓ configuration must not be too high so that each shard would consist of one node, nor too low so that the scalability is not much enhanced. Thus, the first configuration needed is an adequate number of shards per a given V.

Since the aim of sharding is to process TXs and blocks faster, we do also care about shards' individual throughput level. The more shards with high throughput, the higher the throughput of the whole system. To achieve an optimal throughput in a given shard, the weight or the diameter of the network within needs to be minimized. As in non-sharded networks, the lower the weight or the diameter, the sooner data are propagated and, thus, the faster the processing of data. This, in fact, is a well known optimization problem in Graph Theory, namely Graph Partitioning Problem (GPP) [79], whose objective function is to minimize the average weight within shards.

As all optimization problems, GPP can be formalized using heuristics [80], meta-heuristics [81], or linear programming [79]. Aside from meta-heuristics, other formalization approaches impose high computational complexity to solve GPP. However, meta-heuristics can not provide optimum solutions, nor can be controlled to a certain level of optimization. With considering our previous recommendation regarding the system element to carry on computing the set Γ, we do recommend a meta-heuristic approach to solve the problem since it is fast and requires low computational capacity. However, the optimization criteria is not necessarily accurate. Detailed analysis and comparison between optimization criteria may reveal that minimizing the average weights of shards does not provide optimal scalability. For example, investigating the minimization of average computing capacity of shards might be a good research direction. The multi-objective minimization of average computing capacities together with average weights could be even a better approach (e.g. using a multi-objective ACO [82] or MOHEFT [83]).

As will be discussed later, state-of-the-art solutions do not optimize node distribution as they randomly distribute them among shards. Accordingly, those solutions only partially benefit from sharding the network as TXs are processed in parallel logical threads (i.e. shards) without the optimization of those threads.

3) SECURITY AND PRIVACY

The security-related *agreement* property in a given sharded BC is benchmarked with reference to the previously discussed Ψ benchmark depending on the CA used. In Bitcoin, Ψ is determined with reference to Inequality 4. If the Bitcoin network is to be sharded, each shard must provide a level of agreement that is equivalent to the agreement level of Bitcoin when it is not sharded. Specifically, Condition 9 must always hold.

$$q_{Shard_i} < \frac{T_{Shard_i}}{2} \quad \forall \; Shard_i \in \Gamma \tag{9}$$

The requirement to maintain Ψ within each shard is necessary so that the outputs of the shard, i.e. confirmed TXs, agreed-on chain version, and the set Γ in case of our recommendation, are reliable for the system. Since each shard processes a different set of TXs, an inter-shard protocol merges all distinct chain versions produced by each shard into one global DL. Merging approaches have been discussed in the literature and are out of the scope of our work. However, Bitcoin's sharding protocol should be able to maintain Condition 9 simply because a fraud sub-chain would not be detected as a fraud once it is confirmed by a shard's majority. If Ψ is not guaranteed on the shard level then no TX can be proven valid.

In most sharding protocols, randomized sharding is regularly run in order to maintain a high probability that Condition 9 holds. However, such sharding approach waves away the optimal data propagation within, and among, shards. Specifically, this approach results in relatively low throughput per shard, in exchange for (potential) high level of security, resulting in the throughput of some shards to be nearly equivalent to the non-sharded network's. Nonetheless, randomized sharding can never guarantee that condition 9 always holds since adversary nodes are not detected prior, or during, the run of the sharding protocol. Thus, it is not recommended for such approach to be used in public and permissionless BCs [84], such as Bitcoin.

Some previous works suggested using DSs to sign each TX and block, which may indeed solve the security problem. On the other hand, using DSs would further decrease the throughput, due to the relatively high time complexity for signing and verifying, probably resulting in a total throughput even worse than the original throughput of the non-sharded network. Additionally, the utilized DSs represent ID pointers to miners, which raises concerns regarding linkage attacks possibility in public BCs.

4) EXPERIMENTS AND DISCUSSION

The trade-off between scalability and security in sharded public BCs is an open optimization problem [85]. The utilization of a TTP further deviates the system towards centralization, while the utilization of a leader election raises identity privacy concerns [77]. Decentralization and Privacy were among the few core issues that Bitcoin primarily attempted to solve. As long as this Decentralization-Privacy-Security-Scalability (DePriSS) tetralemma[4] is not solved, sharding the Bitcoin network would always lead to the violation of its third major assumption. That is, even if $p > q$, non of the state-of-the-art BC sharding solutions guarantee Condition 9 to always hold.

TABLE 2. Technical details of state-of-the-art sharding solutions.

Sharding Solution	TTP?	Ψ	DS?	Approach
[87]	Yes	$f \leq N/4$	No	Randomized
ELASTICO [88]	Yes	$q \leq 1/4$	Yes	Randomized
Ostraka [86]	Yes	N/A	No	Randomized
SBSCH-GKA [89]	Yes	N/A	Yes	N/A
Cycledger [90]	Yes	$f \leq$ N/3	Yes	Randomized
RepChain [91]	Yes	$f \leq$ N/3	Yes	Randomized
SSHC [92]	Yes	$q \leq 1/3$	Yes	Randomized
RapidChain [93]	No	$f \leq$ N/3	Yes	Cuckoo [94]
Required for Bitcoin	No	$q \leq 1/2$	No	Non-Randomized

In Table 2, we provide brief technical details of recent state-of-the-art works for sharding BCs. The table presents the proposals' requirements for a TTP, or a leader, the initial security assumption, the utilization of DSs, and the GPP sharding approach used. It is obvious from the table that none of the previous works had proposed a suitable sharding solution for Bitcoin, or similar, networks. Even in [86], where the authors claimed that the security level is preserved as is prior to sharding, they had the exception of Bitcoin and left it as a future research direction.

RapidChain [93] is the closest in its configuration to the needed sharding protocol for Bitcoin. Here, allowing the use of DSs as a relaxation of Privacy and Scalability requirements could not provide a protocol that always guarantees the validity of Equation 9 while $q \leq 1/2$. In short, a Bitcoin sharding protocol should fulfill the requirements presented in Table 2 to address the DePriSS tetralemma. Otherwise, it would be much easy for an adversary miner to overpower and control the system output although it occupies a minority of computational power.

Let's assume that the Decentralization and Privacy issues were addressed in a sharding protocol that uses a randomization sharding approach. We want to test how such sharding approach contributes to Scalability and Security issues. To do that, we build a simple Bitcoin network simulator using Python 3.9, in which we deployed Networkx[5] and PyMetis[6][7] libraries. To mimic the connection model in Bitcoin, we use a random graph connection model in our experiments, namely Erdos-Renyi model [95], with each node connected to approximately $V/2$ nodes. The weights of edges, representing the transmission time between each two adjacent nodes, was configured randomly as well. The simulator iteratively builds random networks with $V \in [100, 450]$

[4]Or quadrilemma, defined as a difficult choice between four alternatives.
[5]networkx.org
[6]metis.readthedocs.io/en/latest
[7]Github.com/inducer/pymetis

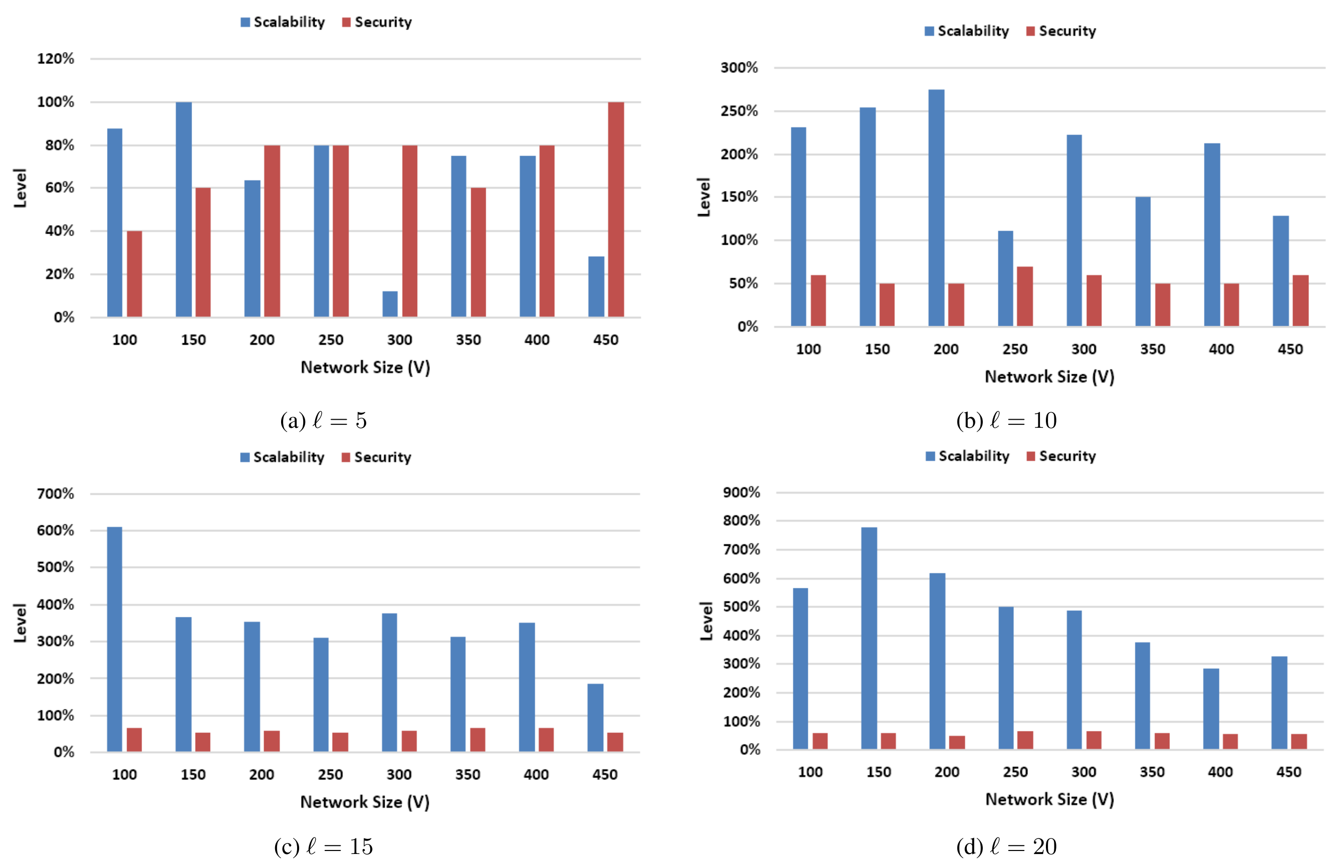


FIGURE 3. **Scalability and Security in randomly sharded networks with $q = 0.4$ and different configurations of number of shards (ℓ).**

and $\ell \in [5, 20]$, then it measures the Scalability and Security levels, of the sharded network versions, using Equations 10 and 11, respectively.

$$Scalability = (1 - \frac{Avg\_Shard\_Diameter}{Network\_Diameter}) \times \ell \quad (10)$$

$$Security\_Level = \frac{Number\_Of\_Secure\_Shards}{\ell} \quad (11)$$

In both of those equations, the higher the result the better. Specifically, the first part of Equation 10 provides the expected throughput outperformance of each individual shard, compared to the non-sharded network (i.e. on average, with reference to the average diameter of shards and of the network). The second part of Equation 10 multiplies this enhancement by the number of shards, resulting in the total expected throughput enhancement of the network due to sharding. Equation 11, on the other hand, tests the security of each shard, using Equation 9, and computes the ratio between the number of secure shards to the total number of shards. A sharding protocol that fulfils the Security requirement should always provide a security level equal to 1.

Our initial parameterization for all of the non-sharded networks complies with the third assumption (i.e. $p > q$). We ran the described experiments on Google Cloud Platform using a VM of type c2-standard-4 (4 Intel Cascade Lake vCPUs clocked at 3.8GHz with 16GB of RAM), running Ubuntu 18.04 LTS. Our code is publicly available at Github.[8] The experiments aim to highlight that even if randomly sharding the Bitcoin network would provide higher scalability, it can never guarantee the security unless either DSs were used (thus negatively affects the scalability enhancement), or a TTP (or elected leader) thus further deviates the network towards centralization and less privacy.

The results of our experiments are depicted in Figure 3. As can be noted, increasing the number of shards does indeed significantly increase the scalability even though nodes were distributed randomly among shards. However, Equation 3 was violated in almost all of the cases, which proves that such sharding approach violates Bitcoin's third assumption on the shard level.

VII. OTHER APPROACHES

We also found other mining approaches in the literature that violate the first assumption of PoW-based BCs. As a result, those approaches either increase mining profits in spite of the utilization of similar resources to an honest majority, or allow to overpower the system despite controlling a minority of the total computational power.

[8] github.com/HamzaBaniata/Sharding

A. PARTIAL PRE-IMAGE ATTACK

Heusser [96], [97] proposed encoding the nonce search as a decision problem solved by a SAT solver [98] in such a way that a satisfiable nonce is obtained. The key ingredients in the algorithm are a non-deterministic nonce, and the ability to take advantage of the known structure of a valid hash 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. Consequently, dishonest miners that use this attack approach are expected to eventually overpower the network despite controlling a minority.

We have recently presented our formal analysis of this attack in [99]. Our analysis predicted an exact critical difficulty value ($\hat{r} = 56$). Once this puzzle difficulty is reached, an attacker can (with storage limitations) compromise the system regardless of the proportion controlled ($q > 0$).

B. SELFISH MINING

Eyal and Sirer [100] showed that the mining protocol in PoW-based BCs is not incentive compatible. That is, miners in public permissionless BCs typically organize themselves into mining pools [101]. All members of a pool work together to mine each block, and share their revenues when one of them successfully mines a block. The research presented a selfish mining attack model, with which colluding miners' revenue is larger than their fair share. This attack was reported to have significant consequences since rational miners would prefer to join the attackers, and the colluding group will increase in size until it, eventually, becomes a majority. To address this, the authors proposed practical modifications on the PoW protocol that may secure the system in the general case. However, it was evident that a coalition commanding a minority of $q = 25\%$ or more of the resources can launch the presented attack.

C. HYBRID DOUBLE SPEND AND SYBIL ATTACK

Zhang and Lee [102] proposed an attack model that combines a double-spend attack with a Sybil attack was proposed. Specifically, a Sybil attack is proposed to be run, in order to delay the propagation of newly mined blocks. Meanwhile, the attacker runs a double-spend attack to refund payments confirmed earlier. The attack was formalized and analytically evaluated, and it was proven that $q = 32\%$ is sufficient to successfully rewrite the BC history, which is far lower than the claimed $q = 50\%$ proposed by Nakamoto. However, two approaches to mitigate such attack were proposed in [103], namely charging an identity fee for miners and setting a deadline time to receive block confirmation. That is, attacked party can refuse to hand over the commodity to the attacker if z block confirmations were not received within a stipulated time.

D. ACCELERATED BLOCK RATE

Kiayias and Panagiotakos [104] presented a theoretical attack against PoW-based BCs that works when BC rate is highly accelerated. The attack was validated in simulation, presenting a natural upper bound in the context of the speed-security trade-off. In a nutshell, the presented analysis showed that the security of PoW-based BCs is bound by about $q = 49\%$ which is, although satisfactory compared to other attacks presented in this paper, less than the claimed $q = 50\%$.

VIII. CONCLUSION AND FUTURE RESEARCH DIRECTIONS

Controlling a minority of the network's computational power, by a dishonest party, is claimed to be tolerated in Nakamoto's model. In this paper, we investigated this claim, and we reviewed several approaches that can be used to undermine/overpower PoW-based BCs. Specifically, we technically discussed how a dishonest miner can take over the network using improved Brute-forcing, AI-assisted mining, Quantum Computing, Sharding, Partial Pre-imaging, Selfish mining, among other approaches. As a result, we found that several practical approaches are available in the literature to undermine PoW-based BCs under some circumstances, even with minority.

A. SUMMARY OF FINDINGS

Our findings and observations can be summarized as follows:

1) A function that generates random values, whose randomness level matches the randomness level of SHA-256, can be used to enhance PoW mining. Theory on the mining outperformance, compared to classical mining, and experiments can then be conducted and evaluated.
2) Machine Learning can be used to conditionally enhance PoW mining. Machine Learning can enhance PoW mining, in terms of block finality, up to 26.6%. An attacker that controls 35.4% or more of the network can unfairly mine PoW-based Blockchains using Machine Learning.
3) Current advancements in Quantum Computers are still not sufficient to attack or mine PoW-based Blockchains.
4) There is no available sharding protocol for permissionless PoW-based Blockchains that guarantees the maintenance of system security. All current works require/solve for one, or more, of the following:
    - Trusted Third Party inclusion (lower decentralization),
    - Lower security threshold,
    - Utilization of Digital Signatures (lower privacy and higher processing time),
    - Non-optimized propagation time (lower scalability).
5) SAT solvers can be used to attack or mine PoW-based Blockchains, once a predefined puzzle difficulty of 56 leading zeros is reached.

6) A mining pool that controls a minority of 25% or more of the network can unfairly mine PoW-based Blockchains using Selfish Mining attacks.
7) An attacker that controls a minority of 32% or more of the network can unfairly mine PoW-based Blockchains by combining Double-Spend and Sybil attacks.

B. FUTURE RESEARCH DIRECTIONS

Our future research directions include the investigation of answers to Research Question 1. We also plan to deepen our knowledge regarding suitable ML models that can be used similar to the models presented. Accordingly, we hope to increase the potential of AI utilization for more optimized AI-based PoW mining.

Furthermore, our future research will investigate technical details of Quantum Computing, including detailed tutorials on selected Quantum Computing tools and simulators, leading to experimental results of Quantum Computing utilization for attacking, or for alternative mining in, PoW-based BCs.

APPENDIX A SIMPLIFIED ASSUMPTIONS AND CALCULATIONS

To facilitate our discussions in the this paper, we list the main notations and abbreviations used in Table 3.

TABLE 3. 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 BC
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 PoW puzzle
Q	Probability $k_j \in K$ solves PoW puzzle
Ω	Non-Zero digits allowed for a correct PoW
r	Zero digits required for a correct PoW
Abbreviation	**Description**
BC	Blockchain
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

In the original paper, only one attack model was analyzed, where an attacker controls q. Assuming Equation 1 holds, Equation 12 was adopted.

$$p + q = 1 \tag{12}$$

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-fraud block. In this model, the two portions of the network, i.e. attacker and honest portions, are racing towards finding the valid block.

The Bitcoin proposal was mainly built upon the following three major assumptions:

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 mechanism requires only computational power, and
3) **Honest majority**: At any given moment, $p > q$ is valid. Accordingly, honest miners are expected to have higher probability to find the next valid block.

Following these assumptions, it was 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 the next valid block is found by any given miner $\in V$, and $\overline{\Phi}$ be the probability the next valid block is NOT found by that miner, then:

$$\Phi + \overline{\Phi} = 1 \tag{13}$$

The probability a miner $\in M$ finds the next block, denoted as E, should then be calculated using Equation 14, while the probability a miner $\in K$ finds the next block, denoted as Q, should be calculated using Equation 15.

$$E = p \times \Phi \tag{14}$$
$$Q = q \times \Phi \tag{15}$$

Although it was not mentioned clearly, Bitcoin calculations assumed that $\overline{\Phi}$ is always equal to 0, which resulted in Equation 16 (by summing Equations 14 and 15, with reference to Equation 12, 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 17, to compute the attacker's potential progress λ for a given block at depth z. Several other inaccuracies of the formalization and validation in the original Bitcoin proposal were discussed in [105], [106], and [107].

$$E + Q = 1 \tag{16}$$
$$\lambda = z\frac{q}{p} \tag{17}$$

By the time of writing the original paper, there was no evident algorithm proving an efficient method to run Preimage attacks on SHA-256. However, several researchers [24], [108], [109], [110], [111], [112] attempted, and still attempting, to run such attacks. As long as a preimage 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 to forge an already confirmed 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, 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 followed this argument, it is easily discoverable whether the third assumption does not hold anymore (i.e. using tools such as [113]). Even with an attacker that follows the rules, the system would lose its credibility against user trust, once it was found that $q > p$.

REFERENCES

[1] G. Praveen, M. Anand, P. K. Singh, and P. Ranjan, "An overview of blockchain consensus and vulnerability," in *Proc. Int. Conf. Inf. Commun. Technol. Intell. Syst.* Cham, Switzerland: Springer, 2020, pp. 459–468.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Decentralized Bus. Rev., White Paper, Oct. 2008, p. 21260.

[3] M. Jakobsson and A. Juels, "Proofs of work and bread pudding protocols," in *Secure Information Networks*. Boston, MA, USA: Springer, 1999, pp. 258–272.

[4] *Secure Hash Standard (SHS)*, FIPS PUB, Standard 180-4, 2012.

[5] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Proc. Conf. Theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 1987, pp. 369–378.

[6] H. Massias, X. S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal trust requirements," in *Proc. 20th Symp. Inf. Theory Benelux*, May 1999.

[7] N. Kshetri and J. Voas, "Blockchain-enabled E-voting," *IEEE Softw.*, vol. 35, no. 4, pp. 95–99, Jul./Aug. 2018.

[8] Z. Liao and S. Cheng, "RVC: A reputation and voting based blockchain consensus mechanism for edge computing-enabled IoT systems," *J. Netw. Comput. Appl.*, vol. 209, Jan. 2023, Art. no. 103510.

[9] G. Alandjani, "Blockchain based auditable medical transaction scheme for organ transplant services," *3C Tecnología_Glosas Innovación Aplicadas Pyme*, vol. 10, no. 4, pp. 41–63, Nov. 2019.

[10] A. Hasselgren, K. Kralevska, D. Gligoroski, S. A. Pedersen, and A. Faxvaag, "Blockchain in healthcare and health sciences—A scoping review," *Int. J. Med. Informat.*, vol. 134, Jan. 2020, Art. no. 104040.

[11] X. Xiang, J. Cao, and W. Fan, "Decentralized authentication and access control protocol for blockchain-based e-health systems," *J. Netw. Comput. Appl.*, vol. 207, Nov. 2022, Art. no. 103512.

[12] M. Samaniego, U. Jamsrandorj, and R. Deters, "Blockchain as a service for IoT," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Dec. 2016, pp. 433–436.

[13] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F.-Y. Wang, "Blockchain-enabled smart contracts: Architecture, applications, and future trends," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 11, pp. 2266–2277, Nov. 2019.

[14] T. G. Jiang, H. Fang, and H. Wang, "Blockchain-based Internet of Vehicles: Distributed network architecture and performance analysis," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4640–4649, Jun. 2019.

[15] L. Gyongyosi and S. Imre, "A survey on quantum computing technology," *Comput. Sci. Rev.*, vol. 31, pp. 51–71, Feb. 2019.

[16] G. Yu, X. Wang, K. Yu, W. Ni, J. A. Zhang, and R. P. Liu, "Survey: Sharding in blockchains," *IEEE Access*, vol. 8, pp. 14155–14181, 2020.

[17] M. Swan, *BlockChain: Blueprint for a New Economy*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2015.

[18] A. Kertesz and H. Baniata, "Consistency analysis of distributed ledgers in fog-enhanced blockchains," in *Proc. Int. Eur. Conf. Parallel Distrib. Comput.*, vol. 27, 2021, pp. 393–404.

[19] X. Yang, Y. Chen, and X. Chen, "Effective scheme against 51% attack on proof-of-work blockchain with history weighted information," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, Jul. 2019, pp. 261–265.

[20] Q. Wang, R. Li, S. Chen, and Y. Xiang, "Formal security analysis on dBFT protocol of NEO," 2021, *arXiv:2105.07459*.

[21] C. T. Porter *How to Cheat at VoIP Security*. Oxford, U.K.: Syngress, 2011.

[22] L. Johnson, *Security Controls Evaluation, Testing, and Assessment Handbook*. New York, NY, USA: Academic, 2019.

[23] M. M. J. Stevens, *Attacks on Hash Functions and Applications*. Leiden, The Netherlands: Leiden Univ., 2012.

[24] Y. Sasaki, L. Wang, and K. Aoki, "Preimage attacks on 41-step SHA-256 and 46-step SHA-512," Cryptol. ePrint Arch., White Paper, Sep. 2009, pp. 1–15.

[25] T. Isobe and K. Shibutani, "Preimage attacks on reduced tiger and SHA-2," in *Proc. Int. Workshop Fast Softw. Encryption*. Cham, Switzerland: Springer, 2009, pp. 139–155.

[26] J. Kelsey and B. Schneier, "Second preimages on n-bit hash functions for much less than 2^n work," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 2005, pp. 474–490.

[27] D. M. A. Cortez, A. M. Sison, and R. P. Medina, "Cryptographic randomness test of the modified hashing function of SHA256 to address length extension attack," in *Proc. 8th Int. Conf. Commun. Broadband Netw.*, Apr. 2020, pp. 24–28.

[28] L. Bosnjak, J. Sres, and B. Brumen, "Brute-force and dictionary attack on hashed real-world passwords," in *Proc. 41st Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, May 2018, pp. 1161–1166.

[29] H. Guo and X. Yu, "A survey on blockchain technology and its security," *Blockchain, Res. Appl.*, vol. 3, no. 2, Jun. 2022, Art. no. 100067.

[30] H. Gilbert and H. Handschuh, "Security analysis of SHA-256 and sisters," in *Proc. Int. Workshop Sel. Areas Cryptogr.* Berlin, Germany: Springer, 2003, pp. 175–193.

[31] M. Juliato and C. Gebotys, "SEU-resistant SHA-256 design for security in satellites," in *Proc. 10th Int. Workshop Signal Process. Space Commun.*, Oct. 2008, pp. 1–7.

[32] H. Handschub and H. Gilbert, "Evaluation report security level of cryptography—SHA-256," Issy-les-Moulineaux, France, Tech. Rep., 2002, pp. 1–21.

[33] K. Jonathan and A. K. Sari, "Security issues and vulnerabilities on a blockchain system: A review," in *Proc. Int. Seminar Res. Inf. Technol. Intell. Syst. (ISRITI)*, 2019, pp. 228–232.

[34] R. Benkoczi, D. Gaur, N. Nagy, M. Nagy, and S. Hossain, "Quantum Bitcoin mining," *Entropy*, vol. 24, no. 3, p. 323, Feb. 2022.

[35] H. Guo, W. Tang, Y. Liu, and W. Wei, "Truly random number generation based on measurement of phase noise of a laser," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 81, no. 5, pp. 051137-1–051137-4, May 2010.

[36] M. Iqbal and R. Matulevicius, "Exploring Sybil and double-spending risks in blockchain systems," *IEEE Access*, vol. 9, pp. 76153–76177, 2021.

[37] J. C. Lagarias, "The $3\times + 1$ problem and its generalizations," *Amer. Math. Monthly*, vol. 92, no. 1, pp. 3–23, 1985.

[38] J. A. T. Machado, A. Galhano, and D. C. Labora, "A clustering perspective of the Collatz conjecture," *Mathematics*, vol. 9, no. 4, p. 314, Feb. 2021.

[39] D. Xu and D. E. Tamir, "Pseudo-random number generators based on the Collatz conjecture," *Int. J. Inf. Technol.*, vol. 11, no. 3, pp. 453–459, Sep. 2019.

[40] A. Kaminsky, "Testing the randomness of cryptographic function mappings," Cryptol. ePrint Arch., White Paper, Jan. 2019, pp. 1–25.

[41] K. Bhattacharjee and S. Das, "A search for good pseudo-random number generators: Survey and empirical studies," *Comput. Sci. Rev.*, vol. 45, Aug. 2022, Art. no. 100471.

[42] N. Jan, R. Maqsood, A. Nasir, M. S. Alhilal, A. Alabrah, and N. Al-Aidroos, "A new approach to model machine learning by using complex bipolar intuitionistic fuzzy information," *J. Function Spaces*, vol. 2022, pp. 1–17, Jan. 2022.

[43] Javatpoint. *Regression vs. Classification in Machine Learning*. Accessed: Sep. 2022. [Online]. Available: http://www.javatpoint.com/regression-vs-classification-in-machine-learning

[44] H. Baniata, R. Prodan, and A. Kertesz, "Machine learning for alternative mining in PoW-based blockchains: Theory, implications and applications," White Paper, Oct. 2022.
[45] I. Stewart, D. Ilie, A. Zamyatin, S. Werner, M. F. Torshizi, and W. J. Knottenbelt, "Committing to quantum resistance: A slow defence for Bitcoin against a fast quantum computing attack," *Roy. Soc. Open Sci.*, vol. 5, no. 6, Jun. 2018, Art. no. 180410.
[46] T. Hey, "Quantum computing: An introduction," *Comput. Control Eng. J.*, vol. 10, no. 3, pp. 105–112, Jun.1999.
[47] D. Ilie, W. Knottenbelt, and K. Leung, "Making Bitcoin quantum resistant," MSc. thesis, Imperial College, London, U.K., Jun. 2018, pp. 1–94.
[48] A. Cojocaru, J. Garay, A. Kiayias, F. Song, and P. Wallden, "The Bitcoin backbone protocol against quantum adversaries," Cryptol. ePrint Arch., White Paper, Feb. 2020, pp. 1–45.
[49] O. Sattath, "On the insecurity of quantum Bitcoin mining," *Int. J. Inf. Secur.*, vol. 19, no. 3, pp. 291–302, Jun. 2020.
[50] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th Annu. ACM Symp. Theory Comput.*, 1996, pp. 212–219.
[51] N. Anhao, "Bitcoin post-quantum," bitcoinpq.org, Tech. Rep., 2018. [Online]. Available: https://bitcoinpq.org/download/bitcoinpq-whitepaper-english.pdf
[52] *IBM Q Experience*. Accessed: Jun. 14, 2022. [Online]. Available: http://www.quantum-computing.ibm.com
[53] *Intel Quantum Computing Service*. Accessed: Jun. 14, 2022. [Online]. Available: http://www.intel.com/content/www/us/en /research/quantum-computing.html
[54] *Microsoft Quantum Computing Service*. Accessed: Jun. 14, 2022. [Online]. Available: http://www.microsoft.com/en-us/research/research-area/quantum-computing
[55] M. Mohseni, P. Read, H. Neven, S. Boixo, V. Denchev, R. Babbush, A. Fowler, V. Smelyanskiy, and J. Martinis, "Commercialize quantum technologies in five years," *Nature*, vol. 543, no. 7644, pp. 171–174, 2017.
[56] L. Tessler and T. Byrnes, "Bitcoin and quantum computing," 2017, *arXiv:1711.04235*.
[57] D. Aggarwal, G. K. Brennen, T. Lee, M. Santha, and M. Tomamichel, "Quantum attacks on Bitcoin, and how to protect against them," 2017, *arXiv:1710.10377*.
[58] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Rev.*, vol. 41, no. 2, pp. 303–332, 1999.
[59] A. Parent, M. Roetteler, and K. M. Svore, "Reversible circuit compilation with space constraints," 2015, *arXiv:1510.00377*.
[60] D. I. Ilie, W. J. Knottenbelt, and I. D. Stewart, "Committing to quantum resistance, better: A speed-and-risk-configurable defence for Bitcoin against a fast quantum computing attack," in *Mathematical Research for Blockchain Economy*. Cham, Switzerland: Springer, 2020, pp. 117–132.
[61] M. Santha, T. Lee, and M. Ray, "Strategies for quantum races," in *Proc. 10th Innov. Theor. Comput. Sci. Conf. (ITCS)*, Seattle, WA, USA, 2019.
[62] J. Jogenfors, "Quantum Bitcoin: An anonymous, distributed, and secure currency secured by the no-cloning theorem of quantum mechanics," in *Proc. IEEE Int. Conf. Blockchain Cryptocurrency (ICBC)*, May 2019, pp. 245–252.
[63] O. Sattath, "Quantum prudent contracts with applications to Bitcoin," 2022, *arXiv:2204.12806*.
[64] R. R. Nerem and D. R. Gaur, "Conditions for advantageous quantum Bitcoin mining," 2021, *arXiv:2110.00878*.
[65] S. Holmes and L. Chen, "Assessment of quantum threat to Bitcoin and derived cryptocurrencies," Cryptol. ePrint Arch., White Paper, Jul. 2021, pp. 1–10.
[66] A. Hülsing, "W-OTS+—Shorter signatures for hash-based signature schemes," in *Proc. Int. Conf. Cryptol. Afr.* Berlin, Germany: Springer, 2013, pp. 173–188.
[67] D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, and P. Schwabe, "The SPHINCS^{+} signature framework," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 2129–2146.
[68] T. V. Trinh, "Quantum-safe Bitcoin," M.S. thesis, Dept. Inform., University of Oslo, Oslo, Norway, 2020.
[69] M. D. Noel, O. V. Waziri, M. S. Abdulhamid, A. J. Ojeniyi, and M. U. Okoro, "Comparative analysis of classical and post-quantum digital signature algorithms used in Bitcoin transactions," in *Proc. 2nd Int. Conf. Comput. Inf. Sci. (ICCIS)*, Oct. 2020, pp. 1–6.
[70] J. Proos and C. Zalka, "Shor's discrete logarithm quantum algorithm for elliptic curves," 2003, *arXiv:quant-ph/0301141*.
[71] M. Roetteler, M. Naehrig, K. M. Svore, and K. Lauter, "Quantum resource estimates for computing elliptic curve discrete logarithms," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Cham, Switzerland: Springer, 2017, pp. 241–270.
[72] N. Samtani, "How would quantum computing impact the security of Bitcoin by enhancing our ability to solve the elliptic curve discrete logarithm problem?" SSRN, White Paper, Aug. 2018, pp. 1–16.
[73] *IBM Quantum Computer*. Accessed: Jun. 14, 2022. [Online]. Available: http://www.ibm.com/quantum
[74] Y. Liu, Y. Wang, and Y. Jin, "Research on the improvement of MongoDB auto-sharding in cloud environment," in *Proc. 7th Int. Conf. Comput. Sci. Educ. (ICCSE)*, Jul. 2012, pp. 851–854.
[75] G. Wang, Z. J. Shi, M. Nixon, and S. Han, "SoK: Sharding on blockchain," in *Proc. 1st ACM Conf. Adv. Financial Technol.*, Oct. 2019, pp. 41–61.
[76] A. Arooj, M. S. Farooq, and T. Umer, "Unfolding the blockchain era: Timeline, evolution, types and real-world applications," *J. Netw. Comput. Appl.*, vol. 207, Nov. 2022, Art. no. 103511.
[77] H. Baniata, A. Anaqreh, and A. Kertesz, "DONS: Dynamic optimized neighbor selection for smart blockchain networks," *Future Gener. Comput. Syst.*, vol. 130, pp. 75–90, May 2022.
[78] A. I. Sanka and R. C. C. Cheung, "A systematic review of blockchain scalability: Issues, solutions, analysis and future research," *J. Netw. Comput. Appl.*, vol. 195, Dec. 2021, Art. no. 103232.
[79] M. Cordero, A. Miniguano-Trujillo, D. Recalde, R. Torres, and P. Vaca, "Graph partitioning in connected components with minimum size constraints via mixed integer programming," 2022, *arXiv:2202.11254*.
[80] F. Rothlauf, *Design of Modern Heuristics: Principles and Application*, vol. 8. Cham, Switzerland: Springer, 2011.
[81] I. Boussaïd, J. Lepagnot, and P. Siarry, "A survey on optimization metaheuristics," *Inf. Sci.*, vol. 237, pp. 82–117, Jul. 2013.
[82] H. Baniata, A. Anaqreh, and A. Kertesz, "PF-BTS: A privacy-aware fog-enhanced blockchain-assisted task scheduling," *Inf. Process. Manage.*, vol. 58, no. 1, Jan. 2021, Art. no. 102393.
[83] J. J. Durillo, H. M. Fard, and R. Prodan, "MOHEFT: A multi-objective list-based method for workflow scheduling," in *Proc. 4th IEEE Int. Conf. Cloud Comput. Technol. Sci.*, Dec. 2012, pp. 185–192.
[84] A. Singh, R. M. Parizi, M. Han, A. Dehghantanha, H. Karimipour, and K.-K. R. Choo, "Public blockchains scalability: An examination of sharding and segregated witness," in *Blockchain Cybersecurity, Trust and Privacy*. Cham, Switzerland: Springer, 2020, pp. 203–232.
[85] G. D. Monte, D. Pennino, and M. Pizzonia, "Scaling blockchains without giving up decentralization and security: A solution to the blockchain scalability trilemma," in *Proc. 3rd Workshop Cryptocurrencies Blockchains Distrib. Syst.*, Sep. 2020, pp. 71–76.
[86] A. Manuskin, M. Mirkin, and I. Eyal, "Ostraka: Secure blockchain scaling by node sharding," in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops*, Sep. 2020, pp. 397–406.
[87] S. Cai, N. Yang, and Z. Ming, "A decentralized sharding service network framework with scalability," in *Proc. Int. Conf. Web Services*. Cham, Switzerland: Springer, 2018, pp. 151–165.
[88] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 17–30.
[89] V. S. Naresh, V. V. L. D. Allavarpu, S. Reddi, P. S. R. Murty, N. V. S. L. Raju, and R. N. V. J. Mohan, "A provably secure sharding based blockchain smart contract centric hierarchical group key agreement for large wireless ad-hoc networks," *Concurrency Comput., Pract. Exper.*, vol. 34, no. 3, p. e6553, Feb. 2022.
[90] M. Zhang, J. Li, Z. Chen, H. Chen, and X. Deng, "CycLedger: A scalable and secure parallel protocol for distributed ledger via sharding," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS)*, May 2020, pp. 358–367.
[91] C. Huang, Z. Wang, H. Chen, Q. Hu, Q. Zhang, W. Wang, and X. Guan, "RepChain: A reputation-based secure, fast, and high incentive blockchain system via sharding," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4291–4304, Mar. 2021.
[92] Y. Liu, J. Liu, Q. Wu, H. Yu, Y. Hei, and Z. Zhou, "SSHC: A secure and scalable hybrid consensus protocol for sharding blockchains with a formal security framework," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 3, pp. 2070–2088, May 2022.

[93] M. Zamani, M. Movahedi, and M. Raykova, "RapidChain: Scaling blockchain via full sharding," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2018, pp. 931–948.
[94] S. Sen and M. J. Freedman, "Commensal cuckoo: Secure group partitioning for large-scale services," *ACM SIGOPS Operating Syst. Rev.*, vol. 46, no. 1, pp. 33–39, Feb. 2012.
[95] P. Erdös and A. Rényi, "On random graphs," *Pub. Math., Debrecen*, vol. 6, pp. 290–297, Jan. 1959.
[96] J. Heusser. (2013). *Sat Solving—An Alternative to Brute Force Bitcoin Mining*. [Online]. Available: https://jheusser.github.io/2013/02/03/satcoin.html
[97] N. Manthey and J. Heusser, "SATcoin–Bitcoin mining via SAT," in *Proc. SAT Competition*, 2018, p. 67.
[98] W. Gong and X. Zhou, "A survey of SAT solver," in *Proc. AIP Conf.*, 2017, Art. no. 020059.
[99] H. Baniata and A. Kertesz, "Bitcoin revisited: Formalization, benchmarking, and open security issues," White Paper, Aug. 2022.
[100] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," *Commun. ACM*, vol. 61, no. 7, pp. 95–102, 2018.
[101] P. R. Nair and D. R. Dorai, "Evaluation of performance and security of proof of work and proof of stake using blockchain," in *Proc. 3rd Int. Conf. Intell. Commun. Technol. Virtual Mobile Netw. (ICICV)*, Feb. 2021, pp. 279–283.
[102] S. Zhang and J.-H. Lee, "Double-spending with a Sybil attack in the Bitcoin decentralized network," *IEEE Trans. Ind. Informat.*, vol. 15, no. 10, pp. 5715–5722, Oct. 2019.
[103] S. Zhang and J.-H. Lee, "Mitigations on Sybil-based double-spend attacks in Bitcoin," *IEEE Consum. Electron. Mag.*, vol. 10, no. 5, pp. 23–28, Sep. 2021.
[104] A. Kiayias and G. Panagiotakos, "Speed-security tradeoffs in blockchain protocols," Cryptol. ePrint Arch., White Paper, Oct. 2016, pp. 1–19.
[105] A. P. Ozisik and B. N. Levine, "An explanation of Nakamoto's analysis of double-spend attacks," 2017, *arXiv:1701.03977*.
[106] C. Grunspan and R. Pérez-Marco, "Double spend races," *Int. J. Theor. Appl. Finance*, vol. 21, no. 8, Dec. 2018, Art. no. 1850053.
[107] M. Rosenfeld, "Analysis of hashrate-based double spending," 2014, *arXiv:1402.2009*.
[108] J. Zhong and X. Lai, "Preimage attacks on reduced DHA-256," Cryptol. ePrint Arch., White Paper, Nov. 2009, pp. 1–25.
[109] H. N. Bhonge, M. K. Ambat, and B. R. Chandavarkar, "An experimental evaluation of SHA-512 for different modes of operation," in *Proc. 11th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Jul. 2020, pp. 1–6.
[110] G. Rowe. (2019). *Has SHA256 Been Broken by Treadwell Stanton Dupont?*. Accessed: May 13, 2022. [Online]. Available: https://crypto.stackexchange.com/questions/74251/hassha256-been-broken-by-treadwell-stanton-dupont
[111] J. J. Kearney and C. A. Perez-Delgado, "Vulnerability of blockchain technologies to quantum attacks," *Array*, vol. 10, Jul. 2021, Art. no. 100065.
[112] R. Preston, "Applying Grover's algorithm to hash functions: A software perspective," 2022, *arXiv:2202.10982*.
[113] *List of Mining Pools*. Accessed: May 13, 2022. [Online]. Available: http://www.miningpoolstats.stream

HAMZA BANIATA received the B.Sc. degree in computer and military science from Mutah University, Jordan, in 2010, and the M.Sc. degree (excellence) in computer science from the University of Jordan, Jordan, in 2018. He is currently pursuing the Ph.D. degree with the Department of Software Engineering, University of Szeged, Hungary. He is also a Research Fellow at the Department of Software Engineering, University of Szeged. He has authored several high-quality publications in the domains of blockchain, cloud computing, fog computing, Internet of Things, and security. He is also a member of the IoT-Cloud Research Group at the university, contributing actively to several (inter) national projects, including the Fog-Block4Trust Sub-Grant of the TruBlo EU H2020 Project, the CERCIRAS EU Cost Action, and the OTKA FK 131793 Project. His work experience includes different roles in the domains of ICT and security.

ATTILA KERTESZ is currently an Associate Professor with the Department of Software Engineering, University of Szeged, Hungary, leading the IoT-Cloud Research Group at the Department. He was the Leader of the FogBlock4Trust Sub-Grant Project of the TruBlo EU H2020 Project and the Work Package Leader of the GINOP IoLT Project financed by the Hungarian Government and the European Regional Development Fund. He is also the Leader of the National Project OTKA FK 131793 financed by the Hungarian Scientific Research Fund. He has published over 140 scientific papers having more than 1600 citations. His research interests include the federative management of IoT, blockchain, fog and cloud systems, and data management issues of distributed systems in general. He is also a Management Committee Member of the CERCIRAS and INDAIRPOLLNET EU COST actions, while he has participated in several successful European projects, including ENTICE EU H2020, COST IC1304, COST IC0805, SHIWA, S-Cube EU FP7, and the CoreGRID EU FP6 Network of Excellence Projects. He has been a member of numerous program committees for European conferences and workshops.