

Feasibility Analysis for Sybil Attacks in Shard-Based Permissionless Blockchains

TAYEBEH RAJABI, Isfahan University of Technology, Iran

ALVI ATAUR KHALIL*, Florida International University, USA

MOHAMMAD HOSSEIN MANSHAEI*, Isfahan University of Technology, Iran

MOHAMMAD ASHIQUR RAHMAN, Florida International University, USA

MOHAMMAD DAKHILALIAN, Isfahan University of Technology, Iran

MAURICE NGOUEN, Florida International University, USA

MURTUZA JADLIWALA, University of Texas at San Antonio, USA

A. SELCUK ULUAGAC, Florida International University, USA

Committee-based permissionless blockchain approaches overcome single leader consensus protocols' scalability issues by partitioning the outstanding transaction set into shards and selecting multiple committees to process these transactions in parallel. However, by design, shard-based blockchain solutions are vulnerable to Sybil attacks. An adversary with enough computational/hash power can easily manipulate the consensus protocol by generating multiple valid node identifiers/IDs (i.e., multiple Sybil committee members). Despite the straightforward nature of these attacks, they have not been systematically investigated. This paper fills this research gap by analyzing Sybil attacks in shard-based consensus of PoW blockchain systems. Specifically, we provide a detailed analysis for Elastico, one of the prominent shard-based blockchain models. We show that the PoW technique used for ID generation in the initial phase of such protocols is vulnerable to Sybil attacks when an adversary (could be a group of colluding nodes) possesses enough hash power. We analytically derive conditions for two different Sybil attacks and perform numerical simulations to validate our theoretical results under various parameters. Further, we utilize the BlockSim simulator to validate our mathematical computation and results confirm the correctness of the analysis.

CCS Concepts: • **Security and privacy** → **Logic and verification**; • **Mathematics of computing** → *Numerical analysis*.

Additional Key Words and Phrases: shard-based blockchain, sybil attack, consensus protocol

ACM Reference Format:

Tayebeh Rajabi, Alvi Ataur Khalil, Mohammad Hossein Manshaei, Mohammad Ashiqur Rahman, Mohammad Dakhilalian, Maurice Nguoen, Murtuza Jadliwala, and A. Selcuk Uluagac. 2023. Feasibility Analysis for Sybil Attacks in Shard-Based

*Corresponding author, co-first author

Authors' addresses: Tayebeh Rajabi, t.rajabi@ec.iut.ac.ir, Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan, Isfahan, Iran; Alvi Ataur Khalil, akhal042@fiu.edu, Florida International University, Miami, Florida, USA; Mohammad Hossein Manshaei, manshaei@iut.ac.ir, Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan, Isfahan, Iran; Mohammad Ashiqur Rahman, marahman@fiu.edu, Florida International University, Miami, Florida, USA; Mohammad Dakhilalian, mdalian@iut.ac.ir, Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan, Isfahan, Iran; Maurice Nguoen, mngou002@fiu.edu, Florida International University, Miami, Florida, USA; Murtuza Jadliwala, murtuza.jadliwala@utsa.edu, University of Texas at San Antonio, San Antonio, Texas, USA; A. Selcuk Uluagac, suluagac@fiu.edu, Florida International University, Miami, Florida, USA.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2023 Association for Computing Machinery.

2769-6472/2023/2-ART000 \$15.00

<https://doi.org/XXXXXXXX.XXXXXXX>

1 INTRODUCTION

A blockchain is an append-only and immutable distributed database or ledger that records a time-sequenced history of transactions. One key aspect of the blockchain protocol is the consensus algorithm which enables agreement among a network of *nodes* or *processors* on the current state of the ledger, assuming that some of them could be malicious or faulty. Blockchain protocols are classified as permissioned or permissionless depending on whether a trusted infrastructure to establish verifiable identities for nodes is present or not. One of the first, and still popular, permissionless blockchain protocols is Bitcoin [1]. Consensus in Bitcoin is achieved by selecting a leader node once every 10 minutes (an epoch) on an average, who then gets the right to append a new block onto the blockchain. The network then implicitly accepts this block by building on top of it in the next epoch or by rejecting it by building on another block. Bitcoin uses a Proof-of-Work (PoW) mechanism to select the leader in each epoch. In Bitcoin's PoW, each node independently attempts to solve a hash puzzle, and the one that succeeds is chosen as a leader who gets the right to propose the next block. As PoW involves significant computation, Bitcoin's protocol includes a reward mechanism for the winning node to incentivize everyone to behave honestly. Ever since the introduction of Bitcoin, the power of permissionless blockchain technology has been harnessed to create systems that can host and execute distributed contracts (i.e., smart contracts), which have found many exciting applications, including cryptocurrencies, secure data sharing, and digital copyright management to name a few [2].

Other than the extreme energy requirements of Bitcoin, which is at least three orders of magnitude higher than the highest-consuming Proof-of-Stake (PoS) systems [3, 4], a significant shortcoming of Bitcoin's leader and PoW competition-based consensus protocol is its low transaction throughput and poor network scalability. For instance, Bitcoin's and Ethereum's transaction rates are only 7 and 20 transactions per second, respectively, which is not sufficient for practical applications [5]. Although there have been several efforts towards improving the Bitcoin protocol itself, for instance, BIP [6] and Bitcoin-NG [7], other works have focused on developing alternate high throughput and scalable permissionless blockchain protocols. One key outcome in this line of research is committee or shard-based protocols [8–10] that operate by periodically partitioning the network of nodes into smaller non-overlapping committees, each of which processes a disjoint set of transactions (also called a shard) in parallel with other committees. As each committee is reasonably small, the transaction throughput and scalability can be significantly improved by running a classical Byzantine consensus protocol such as PBFT [11] within each committee (and in parallel across all committees) rather than the traditional PoW based competition used in Bitcoin. The idea of parallelizing the tasks of transaction processing and consensus by partitioning the processor network into committees is very promising and is already seeing deployment [12]. Recent advancements in Layer 2 (L2) blockchains have also introduced solutions that combine increased throughput with established Layer 1 (L1) trust anchors [13, 14]. For example, L2 rollups like Optimistic Rollups and ZK-Rollups compress L2 transactions onto the L1 chain, leveraging its security for scalability. Sidechains enable independent blockchains to interact with the L1 chain, offloading transactions for higher throughput. State channels facilitate off-chain transactions while anchoring the final state on the L1 chain. These developments in L2 blockchains aim to improve scalability while maintaining the trust and security provided by L1 blockchains.

Committee participation in popular shard-based protocols such as Elastico [8] happen by nodes generating verifiable IDs using some pre-defined PoW puzzle at the beginning of each epoch. Only nodes that can generate valid IDs can participate in the consensus process in that epoch. However, it should be easy to see that the ID generation process quickly lends itself to a Sybil attack. An adversary (often a valid nodes in the network) can leverage its computational or hash-power to generate many Sybil IDs and increase its participation within the shards. Through the generated Sybil IDs, the adversary can compromise the intra-committee consensus

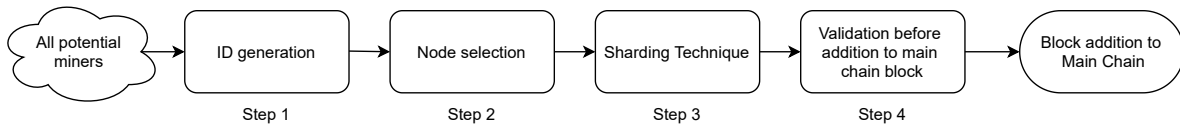


Fig. 1. Basic block diagram of sharding based blockchain protocols.

protocol to either prevent new transaction blocks from being added or to add fake transactions to the blockchain. Existing shard-based protocols assume that all nodes have the same hash-power, and thus the PoW based ID generation mechanism with an appropriate difficulty is not prone to such Sybil attacks. This assumption about the uniformity of hash-power across nodes generally does not hold, making such Sybil attacks feasible. Despite this, there has been a minimal effort within the research community to analyze and combat the threat of Sybil attacks in shard-based blockchains. In this paper, we attempt to address this research gap.

Specifically, we investigate two distinct forms of Sybil attacks within sharded blockchains that leverage the PoW-based ID generation method: (i) *Break Consensus Protocol (or BCP) attack* where the goal of the adversary is to disrupt the consensus process and (ii) *Generate Fake Transaction (or GFT) attack* where the goal of the adversary is to introduce fake or invalid transactions into the blockchain. It is worth mentioning that we consider the “Front-loaded” PoW in this work, where a significant amount of computational work is performed before the blockchain is launched. This precomputed work is stored in the genesis block, making mining faster and easier when the blockchain becomes operational. However, it also presents an opportunity for potential attackers to accumulate a significant portion of the precomputed work and create numerous Sybil IDs. In the rest of the paper, we will refer to “Front-loaded” PoW by just PoW. By assuming a good network and adversary model, we first derive bounds on the success probability of these attacks and then theoretically analyze the necessary conditions to achieve success in these attacks. We further validate our analytical results using numerical simulations for various system and network parameters. These analytical results and simulations shed further light on the computational or hash-power requirement for to compromise a shard-based consensus protocol and the choice of system and network parameters that can significantly reduce the probability of such attacks.

The remainder of this paper is structured as follows. In Section 2, we introduce our system model for a Sybil attack to shard-based blockchain. In section 3, we present our analytical results for the probability of a successful Sybil attack. Section 4 presents the simulation results, following by related work and conclusions in Section 5 and Section 6.

2 TECHNICAL BACKGROUND

In this section, we outline the operational details of shard-based permissionless blockchain systems. Then, we discuss how Sybil attacks can be launched in such blockchain systems.

2.1 Abstract Shard-Based Blockchain Model

In general, the operation of any shard-based permissionless blockchain protocol can be fully represented by means of four sequential steps, as outlined in Figure 1. These steps are executed in each time period or epoch. In step 1, nodes participate in the initial pool by a PoW defined mechanism. Then, in step 2, system-defined required number of nodes, that have passed the step 1, are selected with protocol-defined randomness to participate in the third step. Later, in step 3, selected nodes in the initial pool are distributed among the shards following a protocol-specific deterministic way. Finally, in step 4, the transactions are verified and added to blocks. The blocks are then appended to the ledger of the shard, forming the blockchain. The Fork handling in shard-based protocols can also be abstracted since it involves similar techniques for managing conflicts and divergences in the

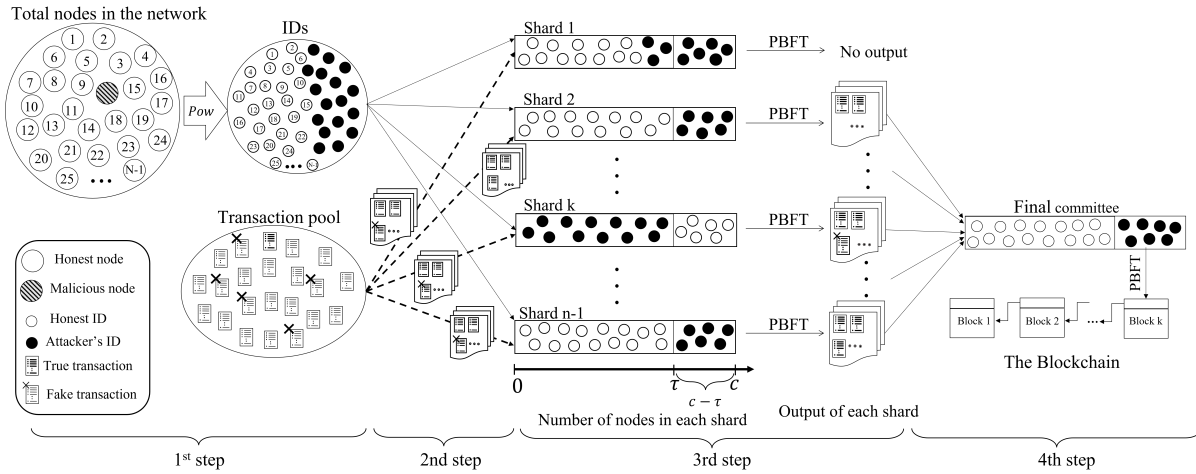


Fig. 2. Elastico system model.

blockchain's history within the multiple shards. To handle forks in shard-based protocols, various approaches are utilized, including deterministic or probabilistic methods to select a single canonical chain among conflicting branches, mechanisms for shard coordination to achieve consistency, and cross-shard communication protocols to reconcile divergent states [15]. In the following sections, we will describe some of the prominent shard-based blockchain protocols, namely Elastico [8], RapidChain [16], Omniledger [9], and Monoxide [17], in details, with respect to the basic steps. We will conclude each of the sections by briefly discussing the protocols' vulnerability to Sybil attack.

2.1.1 Elastico. The basic operation of Elastico is represented in Figure 2, which can be described by four sequential steps. In step 1, all nodes generate their IDs using a PoW defined mechanism. Then, in step 2, IDs and transactions will be distributed among shards, given the generated IDs. There exist 2^s shards, where s is a network-defined parameter. Each shard comprises of up to c IDs. Later, in step 3, each shard runs a PBFT consensus mechanism to make sure the assigned transactions set is validated. At least τ ($= c \times 2/3$) member nodes have to agree on the decision (transaction validation) of each shard. Finally in step 4, the final committee (established based on the consensus from the whole network) merges the agreed transactions of shards and creates a final block. The PoW based node ID generation process of Elastico makes it vulnerable to Sybil attack, as an adversary with enough hashing power can create numerous IDs.

2.1.2 Omniledger. The Omniledger process can be summarized in four steps. In step 1, all participating nodes get valid IDs through an identity blockchain using the proof of Work (PoW) mechanism. Step 2 executes a *Secure Distributed Randomness Generation* called RandHound at the start of an epoch to randomly assign a specified threshold of validators (or consensus group members) to new shards and assign new validators, registered from the identity blockchain in the first epoch. In step 3, clients maintain consistency of their cross-shard transactions using Atomix (protocol for atomically processing transactions); simultaneously, validators ensure consistency of the shards' ledgers via ByzCoinX (the consensus protocol used) in step 4 [9]. If the private key used in the Verifiable Random Function (VRF) of Omniledger is compromised, it can allow an adversary to predict the VRF output and control the leader selection process. This vulnerability can lead to the adversary controlling up to 25% of the network's nodes and potentially escalating into a Sybil attack. With such control, the adversary can disrupt

consensus, manipulate transactions, and compromise the security and integrity of the blockchain. Protecting the private key and implementing additional security measures are crucial to mitigate these risks, such as secure key management practices and employing alternative consensus and validation mechanisms.

2.1.3 Monoxide. Here, the system is divided into s shards (zones) and each zone has c nodes or miners [17]. In step 1, the system is divided into consensus zones or shards. Each zone constitutes a single chain system on its own. The number of zones or shards is a power of 2 (i.e., $c = 2^k$, where k is called the sharding scale). In step 2, the nodes are assigned to the zones. The ID is assigned to a zone using the first k bits of the ID. This is also the hash value of its public key. The mining power of miners is evenly diluted across all the zones in step 3. In step 4, miners validate transactions using the in-zone consensus mechanism. The vulnerability of Monoxide, arises from the possibility of colluding nodes manipulating their public keys to generate hashed IDs that place them within the same consensus zone. This vulnerability makes Monoxide susceptible to Sybil attacks, as the colluding nodes can concentrate their influence and potentially manipulate the transactions and consensus process within that zone, undermining the security and integrity of the protocol.

2.1.4 Rapid Chain. The primary operation of RapidChain can be abstracted by four sequential steps. Here, a peer-to-peer network is considered with n nodes, all of which establish IDs by solving computationally-hard puzzle (PoW) in step 1. All participants are assumed to have equivalent computational resources. A synchronous gossip protocol guarantees a message sent by an honest node (authenticated with the sender's private key) will be delivered to all honest nodes within a known fixed time, but order is not preserved. In the step 2, a reference committee is formed of size $O(\log n)$, by the initially formed root committee of size $O(\sqrt{n})$. Later, in step 3, a randomized committee-election protocol takes place (first epoch) allowing participants to agree on a committee of $m = O(\log n)$ nodes in a constant number of rounds. Less than one-third nodes are assumed to be controlled by adversary.

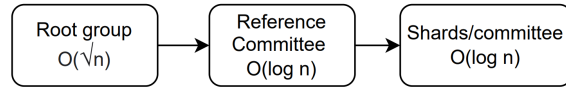


Fig. 3. RapidChain committee formation.

Committee-election protocol samples a committee from all nodes to bound a fraction of corrupt nodes by half with high probability. The referenced committee generates a fresh random string (epoch randomness) at the beginning of every epoch to ensure that it is used to allow new nodes to join the system in every epoch and re-organize the existing committees to prevent adversarial take-over after nodes join and leave the system at the beginning of every epoch. Figure 3 shows the consecutive steps leading up to shard-committees. In step 4, the agreement for transaction validation within committee is achieved by gossiping protocol and synchronous consensus protocol. Every committee member maintains a disjoint transaction ledger (for a shard) as a blockchain. The vulnerability of RapidChain to a Sybil attack is rooted in its PoW-based solution requirement (similar to Elastico), as an adversary with sufficient hashing power can join the protocol with multiple nodes, enabling them to manipulate the consensus process and compromise the security of the network.

2.2 Shard-based Network Model

In this section, we primarily discuss about the Elastico system model, as the network model that we consider in this work and later define the threat model based on it. We leave out the detailed analytical model for the rest of the shard-based blockchain models for our future work.

We assume that the blockchain network comprises N nodes or processors. The hashing powers of the nodes (both honest and attacker nodes) are uniformly distributed within a specific range $[H_x, H_y]$, where $H_y - H_x \leq Q$. Shard-based consensus models in the literature [8, 9] often assume that all nodes have the same hash-power. In such cases, Q will be small, close to zero. Although H_x could be too small to create an ID, the upper bound H_y

extends to an amount of hashing power that enables a node to create multiple IDs. We assume an honest node with maximum capacity will create only one ID. However, a dishonest node can exploit its high computational power to generate multiple IDs (i.e., Sybil IDs) but is limited in its computational capacity. The hash power, which specifies the number of hash computations that can be performed by a node per second, is denoted by h . To elucidate the presentation, Table 1 summarizes the symbols (and their meanings) used throughout the paper.

Like any permissionless system, nodes in *Elastico* do not have any predefined identity assigned by a trusted third party. In the first step of *Elastico*, as shown in Figure 2, each node attempts to generate a verifiable and pseudorandom ID, enabling it to participate in the rest of the steps in that time period or epoch. The nodes use the solution of a Proof-of-Work (PoW) puzzle with a network-determined difficulty to decide if they have arrived at a valid ID. Let $Hash()$ be the hash function employed by a node in the blockchain network and let IP and PK denote a node's network address and public key, respectively. A publicly-known pseudorandom string $epochRandomness$, generated at the end of the previous epoch of the protocol (to avoid puzzle pre-computation), is used as a seed for the PoW puzzle. Each node in the first step of the protocol attempts to solve the PoW puzzle by finding a nonce such that

$Hash(epochRandomness||IP||PK||nonce)$ is smaller than some network-determined *target* value. The target value which determines the difficulty of the PoW puzzle is adapted by the network during each epoch based on the network-wide hash-power. Let's denote the target by L^{t_i} bits, where L is the size of the message digest (in terms of bits) and t_i is the corresponding time epoch. In other words, a valid ID value during epoch t_i must be smaller than $2^{L^{t_i}}$ and a node successful in generating such a valid ID assumes it as its own ID during the later steps of the protocol. It should be noted that all nodes must generate their ID during a given *initialization time* T_I , defined by the protocol. If they cannot solve the puzzle within this time, they will not possess a valid ID to join the rest of the network and participate in the protocol. It should be clear that nodes with a higher hash-power have a higher probability of solving the ID generation PoW puzzle, and thus participating in the protocol, compared to nodes with lower hash-power.

After ID generation, in step 2 the generated node IDs and distributed transactions (which may contain both valid and fake/invalid transactions) are randomly distributed (or partitioned) into different *shards* or committees for validation. In *Elastico*, there exist a total of 2^s shards, where s is a network-defined parameter. Each node will be placed in a shard according to the last s bits of its ID. If the *capacity* of each shard is denoted by c , then it is clear that the minimum number of valid IDs required to execute a single epoch (all steps) of *Elastico* is $N^* = 2^s \times c$. The processors discover IDs of other processors in their shard by communicating with each other.

In the third step, processors of each shard simultaneously validate the transaction set assigned to that shard and agree on a consensus transaction set (within that shard) using the PBFT algorithm [11] [18]. Let τ denote the consensus threshold for each shard, i.e., if at least τ processors in a shard agree on a transaction set, then consensus within the shard is successful and the consensus transaction set is added to the shard's output. In other

Table 1. List of Notations

Symbol	Definition
N	Number of nodes in the network
N^*	Total required IDs at each epoch ($N^* < N$)
M	Number of IDs generated by the adversary
n	Number of selected Sybil IDs from the ID pool
t_i	The time of epoch i
L	The length of output of secure hash function (bit)
L^{t_i}	The length of target value (bit) at epoch t_i
c	Capacity of each shard
h	Hash-power of each processor
s	Represent the number of shards (2^s)
T_I	Initialization time needed for ID generations
τ	Consensus threshold
x	Number of chosen adversary's IDs
$Dif(t_i)$	Difficulty of solving PoW puzzle at epoch t_i
$p(t_i)$	The probability of finding a correct ID at epoch t_i

words, the consensus protocol within a shard successfully outputs a valid consensus transaction set even if $c - \tau$ nodes within the shard do not cooperate and/or behave maliciously. In Figure 2, we see this case in Shards 2 and $n - 1$. Post the intra-shard consensus, the leader node within each shard sends the signed value of the consensus transaction set, along with the signatures of the contributing shard nodes, to a final consensus committee (step 4). The final committee of processors, chosen based on their ID, merges the consensus transaction sets from each shard to create a final block which is eventually appended to the blockchain. Each final committee member first validates that the values (consensus transaction sets) received from each shard is signed by at least $c/2 + 1$ members of the shard, and then computes the ordered set union of all inputs. Finally, nodes of the final committee execute PBFT to determine the consensus final block, which is signed and gets appended as the next block to the blockchain.

2.3 Threat Model

We assume that the adversary (\mathcal{A}) in this setup has enough hash-power to generate more than one IDs (during step 1, as outlined above) and attempts to launch a *Sybil attack* to disrupt the operation of the shard-based consensus protocol. Before outlining the specific Sybil attacks that could be carried out by the adversary, let us first characterize the difficulty of an adversary in generating Sybil nodes. As outlined earlier, each node or processor uses the solution of a PoW puzzle as an ID such that in epoch t_i the selected ID must be smaller than some network-agreed target value $2^{L^{t_i}}$. Let *MaxTarget* denote the maximum possible value of the target. As *MaxTarget* is observed in the first epoch, $MaxTarget = 2^{L^{t_1}}$ (e.g., in Bitcoin $MaxTarget = 2^{224}$ [19]). Given *MaxTarget*, we define the *difficulty* of solving a PoW puzzle in epoch t_i as:

$$Dif(t_i) = \frac{MaxTarget}{2^{L^{t_i}}} \quad (1)$$

It is easy to see that, as the target value $2^{L^{t_i}}$ during a particular time epoch t_i reduces, the PoW puzzle for ID computation becomes harder to solve for the nodes or processors, which is indicated by a higher difficulty value $Dif(t_i)$. Now, the probability of finding a valid ID during epoch t_i is given by:

$$p(t_i) = \frac{2^{L^{t_i}}}{2^L} \quad (2)$$

where 2^L denotes the message digest space of the hash function $Hash()$ using the PoW puzzle. Given Equation 1, $p(t_i)$ can be rewritten as:

$$p(t_i) = \frac{2^{L^{t_1}}}{2^L} = \frac{2^{L^{t_1}}}{Dif(t_i) \times 2^L} \quad (3)$$

Equation (3) represents the formal relationship between the difficulty of finding a valid PoW puzzle solution (i.e., ID in this case) and the success probability of solving the puzzle during epoch t_i . Now, let us assume that the adversary \mathcal{A} 's hash-power (or hash computation capability) is $h^{\mathcal{A}}$. In other words, during the initialization period T_I (step 1) of an epoch t_i , \mathcal{A} can generate a maximum of $h^{\mathcal{A}} \times T_I$ potential solutions (or message digests), of which only those that are smaller than $2^{L^{t_i}}$ (target value during t_i) can be used as valid IDs. Let's further assume that \mathcal{A} can find M valid solutions or IDs (i.e., those that satisfy the puzzle or are within the target value) during T_I . Thus, M can be calculated by:

$$M = p(t_i) \times h^{\mathcal{A}} \times T_I \quad (4)$$

Substituting for $p(t_i)$ from Equation 3 we get:

$$M = \frac{2^{L^{t_1}}}{Dif(t_i) \times 2^L} \times h^{\mathcal{A}} \times T_I \quad (5)$$

Table 2. Differentiating the BCP and GFT attacks.

Distinction	BCP Attack	GFT Attack
<i>Procedure</i>	Adversary breaks the intra-shard consensus protocol in a shard.	Adversary aims to control and manipulate the intra-shard consensus process in a shard.
<i>Requirement</i>	Adversary needs to generate more than $c - \tau$ valid IDs in a (target) shard.	Adversary needs to add at least τ valid IDs in a (target) shard.
<i>Goal</i>	Preventing insertion of some transactions into blockchain blocks.	Include fake (including, double spending or invalid) transactions in the blockchain blocks.

Now, a PoW puzzle based identity or ID generation mechanism is said to be *strictly Sybil-resistant* if and only if the value of M in the mechanism can be restricted to less than 2. In other words, for the above PoW puzzle based identity or ID generation mechanism in Elastico to be strictly Sybil-resistant, the solution difficulty $Dif(\cdot)$ and initialization time T_I should be such that for a given adversary hash-power $h^{\mathcal{A}}$, L , and L^t , M is always smaller than 2, i.e., \mathcal{A} should be able to generate at most one ID during T_I . In this paper, we assume that the shard-based blockchain protocol's PoW-based identity generation mechanism is not strictly Sybil-resistant and that the adversary's hash-power is large enough to generate two or more than IDs (i.e., $M \geq 2$) during the initialization phase (step 1) of any epoch. The adversary then employs these numerous valid IDs for placing itself in multiple shards in order to carry out different types Sybil attacks (as described below), the goal of which is to subvert the correct operation of the blockchain protocol. In contrast to the adversary, we assume that each honest node in the network generates only a single valid ID during T_I in each epoch. Now, let's describe in further details the two different types of Sybil attacks that can be carried out by the adversary. The exact distinctions between the two different types of Sybil attacks are presented in Table 2.

1. Break Consensus Protocol (BCP) Attack: In order to accomplish the BCP attack, the goal of which is to disrupt the shard-based consensus process, the adversary will need to generate more than $c - \tau$ valid IDs in a (target) shard. This threshold of valid IDs will enable the adversary to break the intra-shard consensus protocol in that shard, thereby preventing insertion of some transactions (specifically, from the target shard) into the blockchain. An instance of the BCP attack is depicted in Figure 2, where the adversary successfully inserts more than $c - \tau$ Sybil IDs in Shard 1.

2. Generate Fake Transaction (GFT) Attack: In order to accomplish the GFT attack, the goal of which is to include fake (including, double spending or invalid) transactions in the blockchain blocks by taking over the consensus process, the adversary will need to add at least τ valid IDs in a (target) shard. By doing so, the adversary aims to control and manipulate the consensus process (i.e., the PBFT algorithm) using his Sybil IDs so that intra-shard consensus could be arrived on a desired set of fake transactions, which eventually get inserted into the blockchain block after final consensus. Figure 2 illustrates the GFT attack on shard k .

2.4 Cost of Sybil ID Generation

Given the above threat model, we next study the feasibility (from a cost perspective) of an adversary in generating the Sybil IDs required for the above attacks. For the current difficulty of the PoW puzzle, let H_R be the average number of hashes required to solve the PoW-puzzle for one ID generation. As outlined before, $h^{\mathcal{A}}$ represents the hash capacity of the adversary in terms of the number of hashes that can be performed per second. In order to successfully execute the BCP and GFT attacks described above, $h^{\mathcal{A}}$ should be high enough, i.e., the adversary should be able to generate sufficient Sybil IDs within the initialization period (T_I) to successfully carry out the attacks. Higher the hash-power $h^{\mathcal{A}}$, the more Sybil IDs can be generated within T_I and thus greater is the probability of successful BCP and GFT attacks. Similarly, let $h^{\mathcal{H}}$ be a random variable that represents the hash-power or capability of the honest nodes in the network. We assume that $h^{\mathcal{H}}$ follows a Normal distribution

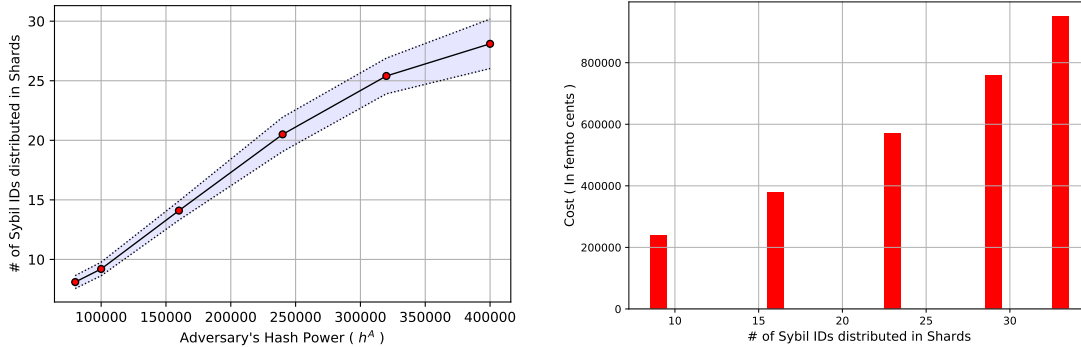


Fig. 4. (a) The average number of Sybil IDs in each shard with confidence interval of 95% for different values of adversary's hash-power, (b) The cost of ID generation.

with mean $\frac{H_R}{T_I}$. In other words, we set the mean value to be the exact amount of hash-power needed to generate a single ID within T_I and the standard deviation is set to be 30% of the mean value. Our goal here is to quantify the hash-power, and the related cost, needed by the adversary to have a certain amount of IDs in the shards. In order to accomplish this goal, we performed the numerical simulations with Python language utilizing Jupyter Notebook from the Anaconda distribution. The simulations were performed on a computer with an 11th Gen Intel(R) Core(TM) i7-1195G7 @2.90GHz processor and 16 GB of memory. In the simulation, we set $c = 5$, $T_I = 5$, and $N = 150$. Figure 4(a) shows the result of our simulation, where we set the adversary hash power (h^A) to 50k, 100k, 160k, 240k, 320k and 400k hashes/sec. For each instance we have ran 10 iterations and average out the results to obtain a more stable and representative result. The results show that the number of distributed IDs is approximately a linear function of the adversary's hash-power.

Now for the computational power cost calculation let's assume that it takes one chip and one unit of electricity to produce one unit of computational power. Thus, the per-block cost (c) of one unit of computational power, including both variable costs such as electricity and a rental cost for capital equipment, can be computed as $c = rC + e$, where C is the cost of a chip, r is the per-block cost of capital (including depreciation) and e is the per-block cost of one unit of electricity [20]. For our simulations, we consider a popular, specialized mining hardware or chip (called ASIC), namely, "Antminer S9". The hash-power of "Antminer S9" is 14.0 TH/s and it costs approximately \$3,000 [21]. The estimated maximum lifetime for this chip is 2 years [22]. So, if it hashes continuously, the maximum number of hashes performed by "Antminer S9" will be $883,008 \times 10^{15}$ hashes in lifetime. The estimated lifetime electricity use of this chip is 24,037 kWh [22], so the electricity usage per hash is 2.7×10^{-17} kWh and cost of it will be 1.36×10^{-16} cents. From the price and lifetime hash number of the chip, we computed the depreciation per hash, which is 1.13×10^{-19} % and the depreciation cost per hash is 3.39×10^{-16} cents. So, the cost of one unit of computational power (from the above equation) will be 4.75×10^{-16} cents/hash. Figure 4(b) shows the results of our simulation for the cost of Sybil ID generation, given the above parameters. We keep the adversary hash power (h^A) similar to the previous simulation (50k, 100k, 160k, 240k, 320k and 400k hashes/sec). These results show that the Sybil attack is plausible given the derived costs of Sybil ID generation.

Our overarching goal is to determine bounds on the probabilities of successfully carrying out the BCP and GFT Sybil attacks in different network and protocol parameters. In the following section, we present theoretical analysis outlining the computation of these probability bounds.

3 ANALYTICAL RESULTS

The remaining operations (step 2 onward) of the shard-based blockchain protocol are initiated in each epoch only after receiving N^* IDs from the ID pool generated during the initialization or identifier generation phase (step 1), as discussed earlier in Section 2. Now, given the M Sybil IDs (generated by the adversary) in the ID pool comprising of a total of $M + N - 1$ IDs, our first task is to analyze the process of ID selection.

Let x denote a random variable representing the number of Sybil IDs (generated by the adversary) chosen from the $M + N - 1$ IDs generated during the initialization step (step 1). In other words, x IDs belonging to the adversary while $N^* - x$ IDs belonging to the honest nodes are distributed among the various shards after the initialization period T_I (step 1). The success probability of the various attacks described above is thus a function of the number of Sybil IDs that the adversary is able to generate and get distributed among the different shards in each epoch. From the discussion in Section 2.3, it should be clear that if the number of Sybil IDs generated by an adversary is smaller than or equal to $c - \tau$, then the adversary cannot successfully execute the BCP or GFT attacks. Let n denote the actual number of Sybil IDs chosen to be distributed among the various shards, i.e., $n \in \{1, 2, \dots, M\}$. Thus, we first need to calculate the probability $P\{x = n\}$ of selecting/choosing n Sybil IDs or nodes from the entire pool of $M + N - 1$ IDs in an epoch. This is given by the following Lemma.

LEMMA 1. *If \mathcal{A} can generate M Sybil IDs during T_I , then the probability of selecting n Sybil IDs from the ID pool is:*

$$P\{x = n\} = \frac{\binom{M}{n} \binom{N-1}{N^*-n}}{\binom{M+N-1}{N^*}} \quad (6)$$

The proof of this lemma follows trivially from the fact that x follows a hypergeometric distribution [23]. The following subsections will be devoted to the computation of the adversary's success probability in executing BCP and GFT Sybil attacks. Table 3 summarizes the definition of main probability bounds.

3.1 Probability of Successful BCP Attack

Recall that for successfully executing the BCP attack, \mathcal{A} must have more than $c - \tau$ (Sybil) IDs in at least one shard. Hence, if $M \leq c - \tau$, \mathcal{A} cannot launch BCP attacks, i.e., the probability of a successful BCP attack would be zero. Thus, to calculate the successful probability of a BCP attack, we first need to calculate the probability of having at least $c - \tau + 1$ Sybil IDs in one shard, when $x = n$ Sybil IDs (generated by the adversary \mathcal{A}) have been chosen from the overall ID pool (at the end of the initialization step). The following lemma captures this probability.

LEMMA 2. *In a shard-based blockchain protocol, when n Sybil IDs (generated by the adversary \mathcal{A}) have been chosen from the ID pool after the initialization step, the probability of having at least $c - \tau + 1$ Sybil IDs in one shard*

Table 3. List of Analytical Variables.

Symbol	Definition
$P\{x = n\}$	The probability of selecting n adversary's ID
$P_{c-\tau+1}$	The probability of having at least $c - \tau + 1$ adversary's IDs in one shard, if $n \in (c - \tau, c]$
$P'_{c-\tau+1}$	The probability of having at least $c - \tau + 1$ adversary's IDs in one shard, if $n \in (c, 2^s(c - \tau)]$
$P''_{c-\tau+1}$	The probability of having at least $c - \tau + 1$ adversary's IDs in one shard, if $n > 2^s(c - \tau)$
P_τ	The probability of having at least τ adversary's IDs in one shard, if $n \in [\tau, c]$
P'_τ	The probability of having at least τ adversary's IDs in one shard, if $n \in (c, 2^s(\tau - 1)]$
P''_τ	The probability of having at least τ adversary's IDs in one shard, if $n > 2^s(\tau - 1)$
P_B	The probability of successful BCP attack
P_G	The probability of successful GFT attack

is :

$$P_{c-\tau+1} = \frac{2^s \sum_{m=c-\tau+1}^n \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{N^*}{c}} \quad (7)$$

where $c - \tau + 1 \leq n$.

Given the number of shards (i.e., 2^s), the capacity of each shard (i.e., c), and the total number of selected IDs (i.e., N^*) the sample space and the space of the desirable event (i.e., having at least $c - \tau + 1$ Sybil IDs in one shard) will be:

$$n(S) = \binom{N^*}{c} \binom{N^*-c}{c} \dots \binom{c}{c} = \frac{N^*!}{c!c!\dots c!} = \frac{N^*!}{c!^{2^s}}. \quad (8)$$

$$\begin{aligned} n(E) = & 2^s \binom{n}{c-\tau+1} \binom{N^*-n}{\tau-1} \binom{N^*-c}{c} \dots \binom{c}{c} \\ & + 2^s \binom{n}{c-\tau+2} \binom{N^*-n}{\tau-2} \binom{N^*-c}{c} \dots \binom{c}{c} \\ & + \dots + 2^s \binom{n}{c-n} \binom{N^*-n}{c-n} \binom{N^*-c}{c} \dots \binom{c}{c}. \end{aligned} \quad (9)$$

The probability $P_{c-\tau+1}$ can then be computed by:

$$P_{c-\tau+1} = \frac{n(E)}{n(S)} = \frac{2^s \sum_{m=c-\tau+1}^n \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{N^*}{c}}.$$

Please note that the above closed-form solution is correct if $n \leq c$. But for the values of n bigger than c we cannot employ the same calculation and deriving a closed-form solution was not possible. In the following analysis we employ numerical analysis to obtain these values, and we leave this derivation for our future work. The following theorem calculates the successful probability of BCP attack, when the number of Sybil IDs is smaller than or equal to c .

THEOREM 1. *In a shard-based blockchain protocol, when the number of Sybil IDs (generated by the adversary \mathcal{A}) is smaller than or equal to c and greater than $c - \tau$, the probability of a successful BCP attack in at least one shard is :*

$$P_B = \frac{2^s \sum_{n=c-\tau+1}^M \sum_{m=c-\tau+1}^n \binom{M}{n} \binom{N-1}{N^*-n} \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{M+N-1}{N^*} \binom{N^*}{c}} \quad (10)$$

The probability of a successful BCP attack in at least one shard is:

$$\begin{aligned} P_B &= P\{x = c - \tau + 1\}P_{c-\tau+1} + P\{x = c - \tau + 2\}P_{c-\tau+1} \\ &+ \dots + P\{x = M\}P_{c-\tau+1} \\ &= \left(\sum_{n=c-\tau+1}^M P\{x = n\} \right) P_{c-\tau+1}. \end{aligned} \quad (11)$$

Given Lemma 1 and Lemma 2, we can rewrite P_B by:

$$P_B = \left(\frac{\sum_{n=c-\tau+1}^M \binom{M}{n} \binom{N-1}{N^*-n}}{\binom{M+N-1}{N^*}} \right) \left(\frac{2^s \sum_{m=c-\tau+1}^n \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{N^*}{c}} \right)$$

This can be rewritten as Theorem 1 (Equation 10).

Having computed this probability, we now attempt to determine the impact of different shard-based blockchain system design parameters on the robustness against such Sybil attacks. In Theorems 2 and 3, we present similar success probability estimations for higher values of M .

THEOREM 2. *In a shard-based blockchain protocol, when the number of Sybil IDs (generated by the adversary) is smaller than or equal to $2^s(c - \tau)$ and greater than c , the probability of a successful BCP attack in at least one shard is :*

$$P_B = \frac{2^s \sum_{n=c-\tau+1}^c \sum_{m=c-\tau+1}^n \binom{M}{n} \binom{N-1}{N^*-n} \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{M+N-1}{N^*} \binom{N^*}{c}} + \frac{\sum_{n=c+1}^M \binom{M}{n} \binom{N-1}{N^*-n}}{\binom{M+N-1}{N^*}} P'_{c-\tau+1} \quad (12)$$

Similar to the proof of Theorem 1, this probability can be calculated by:

$$P_B = \left(\sum_{n=c-\tau+1}^c P\{x = n\} \right) P_{c-\tau+1} + \left(\sum_{n=c+1}^M P\{x = n\} \right) P'_{c-\tau+1}.$$

Here, by replacing $P_{c-\tau+1}$ (using Lemma 2), we find Theorem 2 (Equation 12); the probability of successful BFT attack.

THEOREM 3. *In a shard-based blockchain protocol, when the number of Sybil IDs (generated by the adversary \mathcal{A}) is greater than $2^s(c - \tau)$, the probability of a successful BCP attack in at least one shard is :*

$$P_B = \frac{2^s \sum_{n=c-\tau+1}^c \sum_{m=c-\tau+1}^n \binom{M}{n} \binom{N-1}{N^*-n} \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{M+N-1}{N^*} \binom{N^*}{c}} + \frac{\sum_{n=c+1}^{2^s(c-\tau)} \binom{M}{n} \binom{N-1}{N^*-n}}{\binom{M+N-1}{N^*}} P'_{c-\tau+1} + \frac{\sum_{n=2^s(c-\tau)+1}^{\min(M, N^*)} \binom{M}{n} \binom{N-1}{N^*-n}}{\binom{M+N-1}{N^*}} \quad (13)$$

Similar to earlier proofs, we can compute this probability by:

$$P_B = \left(\sum_{n=c-\tau+1}^c P\{x = n\} \right) P_{c-\tau+1} + \left(\sum_{n=c+1}^{2^s(c-\tau)} P\{x = n\} \right) P'_{c-\tau+1} + \left(\sum_{n=2^s(c-\tau)+1}^{\min(M, N^*)} P\{x = n\} \right) P''_{c-\tau+1}. \quad (14)$$

Since $n > 2^s(c - \tau)$, $c - \tau$ Sybil IDs (generated by the adversary) will definitely reside in each shard. In other words, given the total number of Sybil IDs generated by the adversary, at least $c - \tau + 1$ of these IDs will be placed in at least one shard. When this happens, \mathcal{A} can compromise the consensus protocol of this shard with probability 1. So $P''_{c-\tau+1} = 1$. Moreover based on Lemma 1 and Lemma 2, Equation 14 can be rewritten as Theorem 3 (Equation 13).

In the following, we will similarly analyze the success probability of an adversary in executing the GFT attack.

3.2 Probability of a Successful GFT Attack

In GFT attack, \mathcal{A} would like to change the output of at least one shard, and insert his favorite transactions in the blockchain. The calculation of successful probability of this attack is similar to the previous section (i.e., BCP attack), but the number of required Sybil IDs is different. In this attack, \mathcal{A} must have more than τ Sybil IDs in at least one shard. Hence, if $M < \tau$, the adversary cannot successfully execute the GFT attacks (i.e., probability of success is zero). In order to estimate the success probability of an adversary in carrying out GFT attacks, we first need to calculate the probability of having at least τ Sybil IDs in one shard, when a total of $x = n$ Sybil IDs (generated by the adversary) have been chosen from the pool of all generated IDs (during the initialization step). The following lemma calculates this probability.

LEMMA 3. *In a shard-based blockchain protocol, when n Sybil IDs (generated by the adversary \mathcal{A}) have been chosen or selected from the ID pool after the initialization step, the probability of having at least τ Sybil IDs in one shard is :*

$$P_{\tau} = \frac{2^s \sum_{m=\tau}^n \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{N^*}{c}}$$

where $\tau \leq n$.

According to the Lemma 2, the sample space is equal to Equation 8. Similarly, by considering all possible combinations of Sybil ID distributions within the shards, we can compute the space of the desirable event (i.e., having at least τ Sybil IDs in one shard) by:

$$\begin{aligned} n(E) = & 2^s \binom{n}{\tau} \binom{N^*-n}{c-\tau} \binom{N^*-c}{c} \dots \binom{c}{c} \\ & + 2^s \binom{n}{\tau+1} \binom{N^*-n}{c-\tau-1} \binom{N^*-c}{c} \dots \binom{c}{c} \\ & + \dots + 2^s \binom{n}{n} \binom{N^*-n}{c-n} \binom{N^*-c}{c} \dots \binom{c}{c} \end{aligned}$$

The probability P_{τ} can then be computed by taking the ratio of the event space to the sample space as:

$$P_{\tau} = \frac{n(E)}{n(S)} = \frac{2^s \sum_{m=\tau}^n \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{N^*}{c}}. \quad (15)$$

Similar to BCP attack, the above closed-form solution is correct if $n \leq c$. Finding a closed-form solution for the case where $n > c$ is a part of our future challenges. The following theorem calculates the successful probability of GFT attack, when the number of Sybil IDs is smaller than or equal to c .

THEOREM 4. *In a shard-based blockchain protocol, when the number of Sybil IDs (generated by the adversary \mathcal{A}) is smaller than or equal to c and greater than $\tau - 1$, the probability of a successful GFT attack in at least one shard is :*

$$P_G = \frac{2^s \sum_{n=\tau}^M \sum_{m=\tau}^n \binom{M}{n} \binom{N-1}{N^*-n} \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{M+N-1}{N^*} \binom{N^*}{c}} \quad (16)$$

This probability can be computed similar to the proof of Theorem 1 as:

$$P_G = \left(\sum_{n=\tau}^M P\{x = n\} \right) P_{\tau}. \quad (17)$$

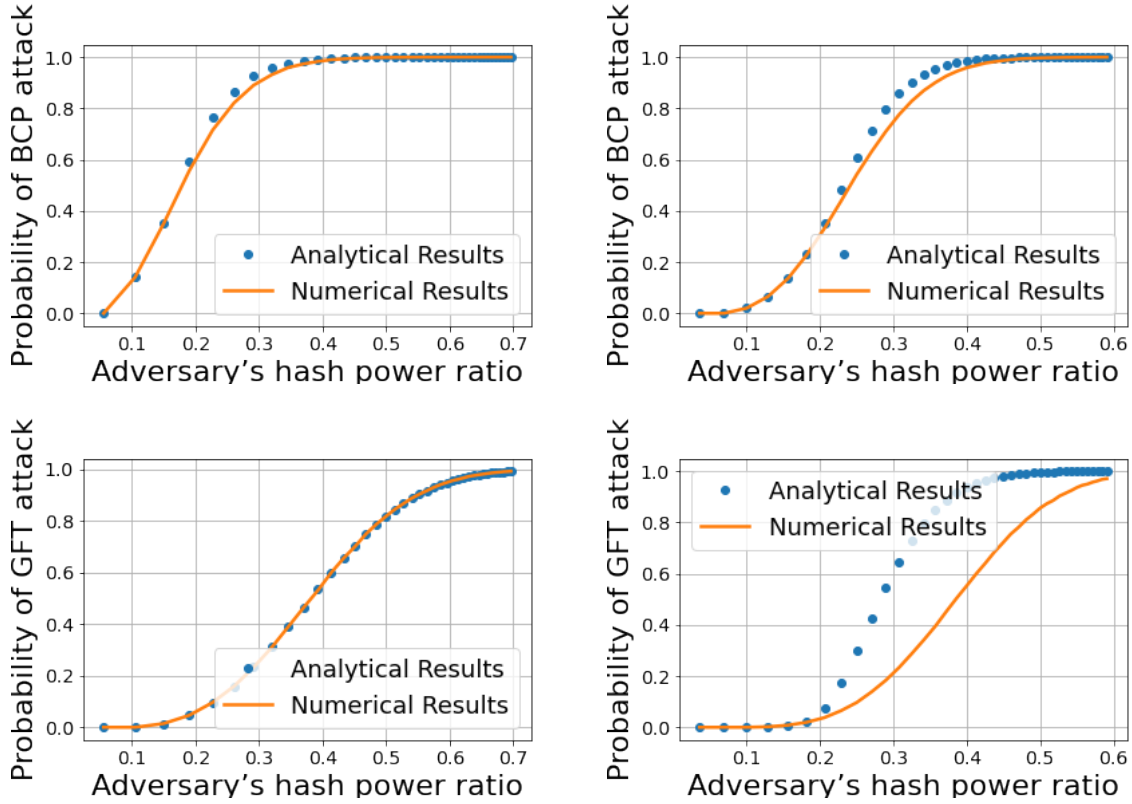


Fig. 5. Our model verification. (a) BCP attack simulations for $N = 18$ and $c = 4$ with 4 shards. (b) BCP attack simulation with $N = 28$ and $c = 6$ with 4 shards. (c) GFT attack simulations for $N = 18$ and $c = 4$ with 4 shards. (d) GFT attack simulation with $N = 28$ and $c = 6$ with 4 shards.

Based on Lemma 1 and Lemma 3, Equation 17 can be rewritten as:

$$P_G = \left(\frac{\sum_{n=\tau}^M \binom{M}{n} \binom{N-1}{N^*-n}}{\binom{M+N-1}{N^*}} \right) \left(\frac{2^s \sum_{m=\tau}^n \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{N^*}{c}} \right)$$

This can be rewritten as Theorem 4 (Equation 16).

Having computed this probability, we now attempt to determine the impact of different shard-based blockchain system design parameters on the robustness against such Sybil-based GFT attacks. In the following two theorems, we present similar success probability estimations for higher values of M .

THEOREM 5. *In a shard-based blockchain protocol, when the number of Sybil IDs (generated by the adversary \mathcal{A}) is smaller than or equal to $2^s(\tau - 1)$ and greater than c , the probability of a successful GFT attack in at least one*

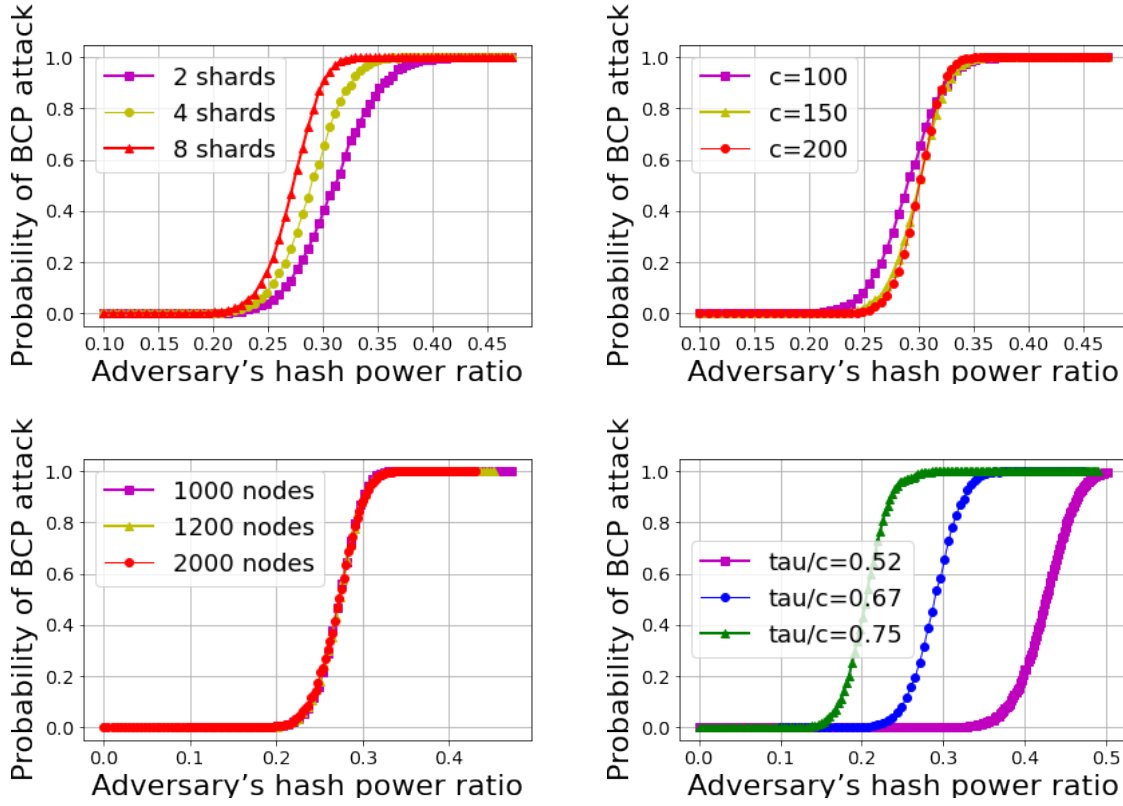


Fig. 6. The effect of (a) the number of shards, (b) the capacity of shard, (c) the number of nodes, and (d) τ on successful BCP attack.

shard is:

$$P_G = \frac{2^s \sum_{n=\tau}^c \sum_{m=\tau}^n \binom{M}{n} \binom{N-1}{N^*-n} \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{M+N-1}{N^*} \binom{N^*}{c}} + \frac{\sum_{n=c+1}^M \binom{M}{n} \binom{N-1}{N^*-n}}{\binom{M+N-1}{N^*}} P'_\tau \quad (18)$$

Similar to the proof of Theorem 4, this probability can be calculated by:

$$P_G = \left(\sum_{n=\tau}^c P\{x=n\} \right) P_\tau + \left(\sum_{n=c+1}^M P\{x=n\} \right) P'_\tau.$$

It is then easy to show that P_G is derived by Equation 18.

THEOREM 6. *In a shard-based blockchain protocol, when the number of Sybil IDs (generated by the adversary \mathcal{A}) is greater than $2^s(\tau - 1)$, the probability of a successful GFT attack in at least one shard is :*

$$P_G = \frac{2^s \sum_{n=\tau}^c \sum_{m=\tau}^n \binom{M}{n} \binom{N-1}{N^*-n} \binom{n}{m} \binom{N^*-n}{c-m}}{\binom{M+N-1}{N^*} \binom{N^*}{c}} + \frac{\sum_{n=c+1}^{2^s(\tau-1)} \binom{M}{n} \binom{N-1}{N^*-n} P'_\tau}{\binom{M+N-1}{N^*}} + \frac{\sum_{n=2^s(\tau-1)+1}^{\min(M, N^*)} \binom{M}{n} \binom{N-1}{N^*-n}}{\binom{M+N-1}{N^*}} \quad (19)$$

Similarly, the probability of a successful GFT attack can be computed as:

$$P_G = \left(\sum_{n=\tau}^c P\{x = n\} \right) P_\tau + \left(\sum_{n=c+1}^{2^s(\tau-1)} P\{x = n\} \right) P'_\tau + \left(\sum_{n=2^s(\tau-1)+1}^{\min(M, N^*)} P\{x = n\} \right) P''_\tau. \quad (20)$$

Since $n > 2^s(\tau - 1)$, $\tau - 1$ Sybil IDs (generated by the adversary) will definitely reside in each shard. In other words, given the total number of Sybil IDs generated by the adversary, at least τ of these IDs will be placed in at least one shard. When this happens, \mathcal{A} can compromise the PBFT consensus protocol to change the output produced by this shard with probability 1. So $P''_\tau = 1$ and we can rewrite Equation 20 as shown by Equation 19. In the following section, we will verify our closed-form solutions with the help of numerical simulations. .

4 MODEL VALIDATION WITH NUMERICAL RESULTS AND DISCUSSION

We validate the correctness of our analytical results discussed earlier by conducting extensive numerical simulations as outlined next. We implement a Python-based simulator for Elastico's consensus technique and simulate the BCP and GFT Sybil attacks by considering an adversary with different hash-powers. Success probabilities of these BCP and GFT attacks during simulations are computed against different hash power ratio of adversary and compared with our previously determined analytical results. The adversary's hash power ratio is defined as follows:

$$\text{Hash power ratio} = \frac{\# \text{ of IDs generated by the adversary}}{\# \text{ of total IDs}}$$

We investigate the effect of various system parameters on the success probability of these attacks, including the number of shards (i.e., 2^s), the capacity of each shard (i.e., c), the total number of participating nodes in the network (i.e., N) and the threshold for consensus (i.e., τ). Finally, we leverage a BlockChain simulator to further validate our analytical model.

4.1 Validation of Analytical Results

We start by considering a small network of 18 participating nodes (i.e., $N = 18$) and 4 shards (i.e., $s = 2$), each with a capacity of 4 nodes (i.e., $c = 4$) undergoing a BCP attack by assuming different adversarial hash-rates. Figure 5(a), which represents the probability of a successful BCP attack under this simulated scenario, shows that results from our simulations align very closely to those obtained from our analytical results (i.e., Theorems 1, 2 and 3). Here, the hash-rate (shown on the x-axis) is computed as the ratio of the adversary's hash-power (\mathcal{A}) to the average hash-power of the entire network. Figure 5(b) shows similar results for a larger network with parameters $N = 28$, $c = 6$, and $s = 2$. Even in this case, it can be observed that our analytical results are in line with the simulated behavior of the Elastico network. We similarly verify the validity of our analytical results

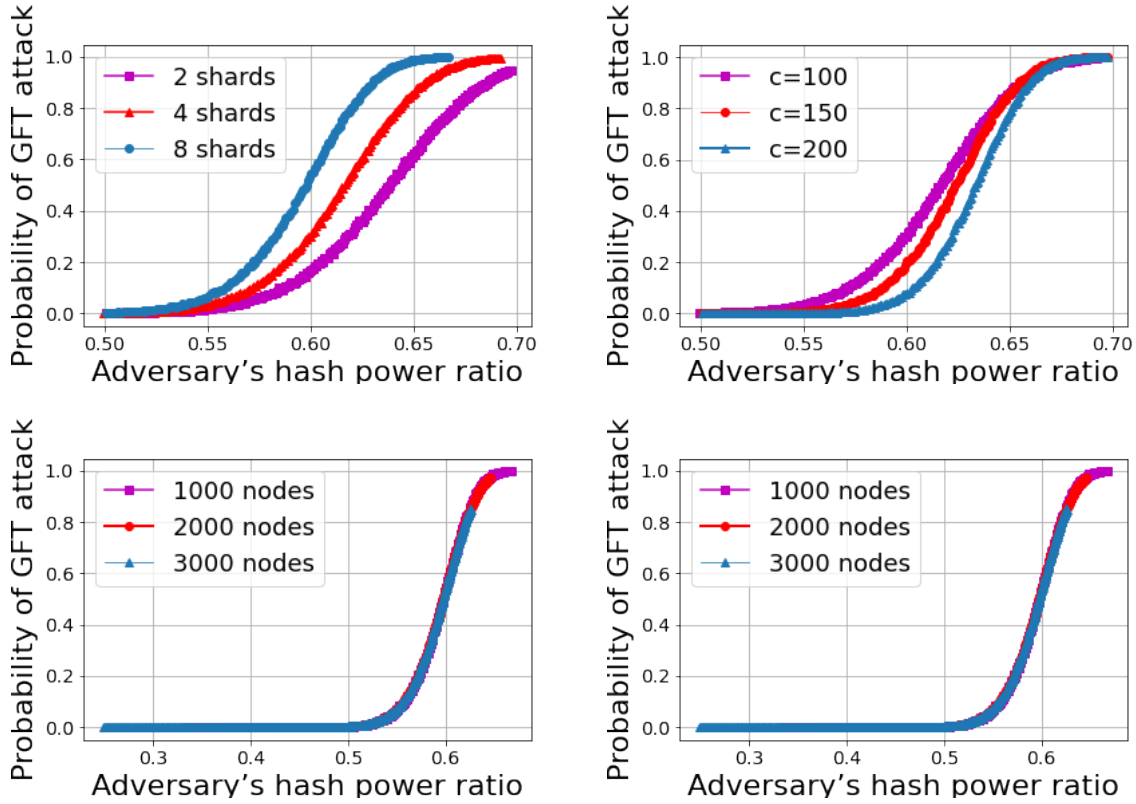


Fig. 7. The effect of (a) number of shards, (b) shard capacity, (c) number of nodes, and (d) τ on successful GFT attack.

(Theorems 4, 5 and 6) for the GFT attacks. Figure 5(c) shows the success probability of the GFT attacks in the Elastico network, with parameters $N = 18$, $c = 4$, and $s = 2$. For a larger network with parameters $N = 28$, $c = 6$, and $s = 2$ in Figure 5(c), the analytical results are also very closely in line with the simulated behavior for GFT attack. These results confirm that our analytical computation of these probabilities was correct. Next, for each attack scenario, we will employ simulations to demonstrate the impact of different system parameters on the success probability of these attacks.

4.2 Numerical Analysis Results

Figure 6 shows the success probability of BCP attacks for different hash-powers of the adversary. Figure 6(a) shows that the BCP attack probability increases when we increase the number of shards. In this experiment, the capacity (c) of each shard is set to 100 and the value of τ/c is set to $\frac{2}{3}$. Our results show that an adversary who has 25% of the network hash-power can compromise (and manipulate) the consensus algorithm employed by the shard-based protocol (e.g., PBFT). Figure 6(b) shows that the probability of successful BCP attacks decreases when the capacity (c) of each shard increases. We also evaluated the effect of the number of active/participating nodes on the probability of successful BCP attacks. We execute the simulations by setting the shard capacity (c) to 100, number of shards (s) to 4 and the threshold value τ/c to $\frac{2}{3}$. Results from these simulations show that the

adversary (\mathcal{A}) needs to have 33% of the hash-power of whole network to launch a successful BCP attack. But, it is indeed more difficult to accumulate 33% of the network's hash-power, if we increase the number of total nodes.

Figure 7 shows the success probability of GFT attacks for different values of the adversary's hash-power. As shown in Figure 7(a), if the adversary's hash-power is less than 50% of the network hash-power, it cannot execute a successful GFT attack. We also observe that the GFT attack probability increases when the number of shards increases. Moreover, Figure 7(b) shows that as the shard capacity (c) increases, the GFT attack success probability decreases. It should be noted that this trend is observable only when the adversary's hash-power is less than 65% of the total network hash-power. If the adversary has more than 65% of the total network hash-power, increasing the capacity (c) has no significant impact on P_G . Figure 7(c) shows the effect of the total number of participating nodes in the network on the success probability of the GFT attack. Our simulation results show that if the adversary holds 65% of the network's hash-power, it can successfully perform the GFT attack when the total number of participating nodes is 2000 or less. However, as the total number of participating nodes increases (e.g., 3000), the success of the GFT attack is not guaranteed.

Finally, we investigate the effect of the consensus threshold (τ/c) on the success probability of BCP and GFT attacks. The results are shown in Figure 6(d) and 7(d). We vary the value of τ/c from 0.52 and 0.75. Our results show that if the value of τ/c changes from τ_1/c to τ_2/c , the adversary needs about $1 - (\tau_2/c - \tau_1/c)$ hash-power of the previous hash-power, in order to achieve the same attack success probability for τ_1/c . As shown in Figure 6(d), if the BCP attack success probability of an adversary with hash-power HP was P_B with $\tau/c = \frac{2}{3}$, then the adversary would need to accumulate a hash-power of $0.91 \times HP$ in order to achieve the same success probability (P_B) with $\tau/c = 0.75$. Figure 7(d) shows the impact of τ/c on the success of GFT attacks. Here we see a similar trend as in the case of BCP attacks, where with higher values of τ/c the adversary needs more hash-power to conduct a successful GFT attack. A notable observation from Figure 6(a) and Figure 7(a) is that increasing the number of shards in a system enhances performance by increasing throughput but also renders it more susceptible to both BCP and GFT attacks. This reveals a trade-off between performance and security, where the benefits of improved performance must be weighed against the heightened vulnerability to Sybil attacks as a consequence of having more shards.

Table 4 summarizes the results of a series of Sybil attack simulations with different parameters. These results show that we can avoid successful GFT and BCP attack with an optimal selection of number of shards and their capacity even if the adversary's hash-power is about 25% of the average hash-power of the network. But if the adversary's hash-power is more than 33%, it can successfully deploy BCP attack. However, with a maximum of 16 shards each with a capacity of 600 nodes, we can decrease the probability of successful GFT to less than 0.001. We believe that these results and parameter choices can help designers prevent such DoS attacks in their distributed shard-based blockchain protocols.

Table 4. Numerical Evaluation of Sybil attacks.

$h^{\mathcal{A}}$	τ/c	#shards	c	P_B	P_G
25%	2/3	At most 16	At least 600	$\leq 10^{-4}$	0
[33%, 53%]	2/3	-	-	≥ 0.8	≤ 0.005
56%	2/3	At most 16	At least 600	1	≤ 0.001
Above 66%	2/3	-	-	1	≥ 0.75

4.3 Experimental Results with Blockchain Simulator

For experimental purposes, we utilize an extensible simulation tool for blockchain systems, named BlockSim by Maher et al. [24]. We perform the experiments by abstracting the shard-based blockchain protocols into two classes, similar to Hafid et al. [25]: (1) class A, where the total resiliency is 1/4 and committee/shard resiliency is 1/3 (includes Elastico, Omniledger, and Monoxide etc.) and (2) class B, where the total resiliency is 1/3 and committee/shard resiliency is 1/2 (includes RapidChain). Figure 8 presents the comparison of the simulation results against the numerical results with respect to probability of successful BCP and GFT attacks for both

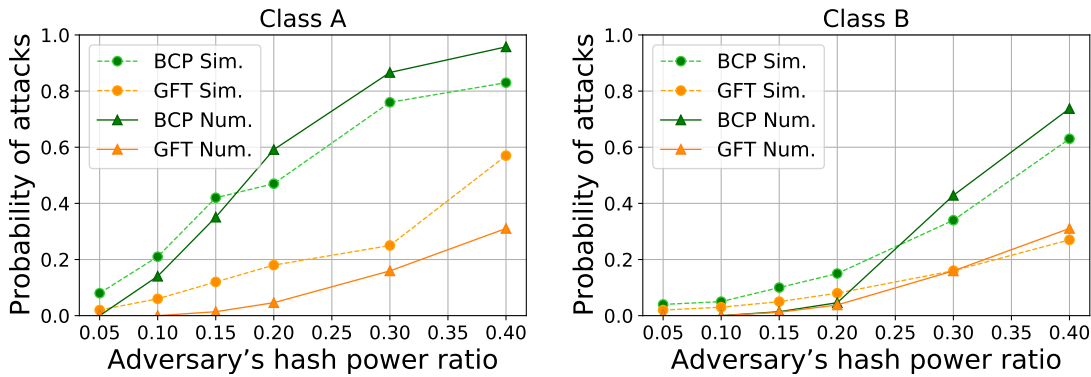


Fig. 8. Probability of BCP and GFT attack with different adversary hash rate: (a) class A and (b) class B.

the classes. Here, we consider network parameters similar to prior analysis of this section ($N = 18$, $c = 4$ and 4 shards) and adversary's hash power ratio is set to 5%, 10%, 15%, 20%, 30%, and 40%. From Figure 8(a), it can be observed that the numerical results are aligned closely to the simulation results for class A. The BCP attack's success probability approximation is more accurate than the GFT attack's success probability approximation. In Figure 8(b), we observe lower probability of attack for both the attacks, primarily due to higher network and shard resiliency. In this case however, the GFT attack's success probability approximation by the numerical analysis is more accurate than the BCP attack's success probability approximation.

4.4 Discussion and Future Work

In this section, we discuss the probable design initiatives toward defending against the Sybil attack and the research objectives of our forthcoming work.

4.4.1 Potential Mitigation Techniques. By conducting a comprehensive analysis, this work takes the crucial first step toward the development of resilient shard-based consensus protocols. Although our analysis in this work does not provide a complete mitigation technique for Sybil attacks, it identifies the initial design objectives for future endeavors that involve the implementation of sophisticated countermeasures. The insights and findings will enable designers to make informed decisions regarding the optimal number of shards and nodes for each shard during the design process. These techniques could include the utilization of randomized allocation of nodes within shards, resource-based allocation methods where nodes are assigned based on their available resources, or reputation-based allocation methods that consider the trustworthiness or reputation of nodes when assigning them to shards. This contributes to the overall goal of creating robust and secure shard-based consensus protocols that can withstand various adversarial scenarios, including Sybil attacks.

4.4.2 Future Work Objectives. Due to the varying susceptibility of shard-based protocols to Sybil attacks based on their specific implementations, we have deferred a comprehensive analysis to our forthcoming research attempts. However, we anticipate that the conclusions drawn from our current work can be readily extended to other shard-based protocols, particularly considering the introduction of new techniques in those systems. Our upcoming research will focus on exploring the operational procedures and assessing the vulnerability to Sybil attacks in prominent shard-based blockchain protocols such as RapidChain, Omniledger, and Monoxide.

By examining these protocols, we aim to contribute further insights and advancements in understanding and mitigating Sybil attacks within the context of sharded systems.

5 RELATED WORK

To put our current research effort in perspective, we now briefly outline some other efforts in the literature towards improving the scalability and security of permissionless blockchains. Bitcoin-NG [7] was the first attempt to improve the transaction throughput of Bitcoin’s consensus protocol [1] by employing the concept of *microblocks*. Due to the significant drawbacks of leader-based consensus protocols, such as Bitcoin and Bitcoin-NG, the research community’s focus shifted to employing committee-based consensus algorithms [26] for permissionless blockchain systems. For instance, Decker et al. [27] proposed one of the first committee-based consensus protocols, named *PeerCensus*, followed by several other proposals [28–30] in a similar direction. The poor scalability of single committee consensus solutions motivated the design of *multiple committee*-based blockchain consensus protocols, where the main idea is to split the pending transaction set among multiple shards and then process these shards in parallel. *RSCoin* [31] was proposed as a shard-based blockchain for centrally-banked cryptocurrencies, while *Elastico* [8] was the first fully distributed shard-based consensus protocol for public blockchains. Later proposed shard-based protocols, such as *Omniledger* [9] and *Rapidchain* [16] attempted to improve upon the scalability and security guarantees of *Elastico*, while *PolyShard* [32] employed techniques from the *coded computing* paradigm [33] to simultaneously improve transaction throughput, storage efficiency, and security. Recently, several novel approaches for shard-based consensus protocols for blockchains have also been proposed [10, 34], and a good review of various shard-based blockchain protocols can be found in [35].

The configuration process in shard-based protocols is a significant challenge. In this part, we will discuss this process in protocols like *Elastico*[8], *Omniledger* [9], *Rapidchain* [16], *Monoxide* [17], *Repchain* [36], *Ethereum II* [37], *SG-PBFT* [38], and *shardable multilayer PBFT* [39]. Consensus methods, especially from the Byzantine consensus family [8, 9, 16, 36, 37], impact node relationships and peer-to-peer communication. Dynamic configuration is preferred, even with other consensus methods like PoW [17]. In most of these protocols, node assignments are handled dynamically in repeated intervals called epochs to reduce attacks [8, 9, 16, 36, 37].

In the protocols [8, 9, 16, 36, 37], consensus process is postponed until configuration completion, while *Monoxide* uses zones with independent PoW. Reliable and unbiased random number generation is crucial for reducing the likelihood of possible vulnerabilities and attacks; however, it can be time-consuming [9, 16]. *Repchain* [36] assumes the availability of reliable random numbers for all nodes. In *Ethereum II* [37], a smart contract called *RANDAO*, performed by a special fraction of nodes, generates random numbers. Hash functions, used in [17], are popular for fast random number generation; however, their reliability and vulnerability are scrutinized in professional assessments. It is worth noting that, despite the extensive research conducted on shard-based blockchains, none of the works reviewed in the analysis adequately addressed the knowledge gap concerning the vulnerability of shard-based blockchains to Sybil attacks. Given the complexity and distributed nature of shard-based blockchains, it is crucial to explore the specific vulnerabilities and investigate potential countermeasures to safeguard against Sybil attacks.

In the direction of security-related analysis and securing shard-based blockchain protocols, Jusik Yun et al. [40] observed that the number of validators per shard is generally smaller than the number of validators in traditional single leader-based protocols, which makes shard-based protocols more vulnerable to 51% attacks. To solve this problem, they then presented a *trust-based shard distribution scheme* that distributes the assignment of potential malicious nodes in the network into shards. Platt et al. [41] analyzed Sybil attack resistance in different blockchain systems and categorized them based on their resistance characteristics, leader selection methodologies, and incentive schemes. They found that mechanisms with strong resistance often use PoW or PoS, while those with limited resistance rely on reputation systems or physical world linking. Few paradigms effectively resist Sybil

attacks in permissionless settings, but there are innovative mechanisms for scenarios with smaller attack surfaces. Hafid et al. [25] was the first to mathematically analyze the security of three shard-based blockchain protocols (i.e., [8], [9], and [16]). They computed the upper bound for the probability of increasing the number of malicious nodes for one committee (and so for each epoch) using tail inequalities for the sum of bounded hypergeometric and binomial distributions. However, they did not validate their analytical results using simulation/implementation.

The viability of Sybil attacks on blockchain consensus protocols is discussed in [42, 43]. In [44], Kedziora et al. studied how susceptible blockchain networks to Sybil attacks, focusing *eclipse attacks* [45] (by isolating honest nodes) as the means. They conduct a series of simulations using an implemented environment to analyze to the extent Sybil attacks in uniformly and randomly distributed computing power between the nodes. While this work discuss potentiality of Sybil attacks through *eclipse attacks*, we focus on Sybil attacks where an adversary relies on creating enough valid IDs (Sybil IDs) to compromise a shard leading to two kinds of attacks on consensus.

6 CONCLUSION

In this paper, we presented an analytical model to calculate the probability of a successful Sybil attack to shard-based permissionless PoW blockchains. We have modeled *Elastico* as the reference shard-based blockchain protocol and defined two types of Sybil attacks: one kind breaks the consensus protocol and the other generates fake transactions. We have shown that we can calculate the probability of successful attacks given different system parameters, such as the number of shards, their capacity, and the number of nodes in the blockchain network. The results have been verified with numerical simulations. We further validate our analytical results using *BlockSim* simulator. We have shown that by carefully designing our blockchain network, we can avoid Sybil attacks. Our results showed that shard-based blockchain protocols are not robust against a Sybil attack against an adversary (a single attacker or a colluding group) that has 25% hash-power of the network. In this case, the adversary can break the consensus protocol in at least a shard with a 20% probability.

REFERENCES

- [1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.
- [2] A. Kiayias, E. Koutsoupias, M. Kyropoulou, and Y. Tselekounis, "Blockchain mining games," in *the ACM Proceedings of the Conference on Economics and Computation*, 2016, pp. 365–382.
- [3] M. Platt, J. Sedlmeir, D. Platt, J. Xu, P. Tasca, N. Vadgama, and J. I. Ibañez, "The energy footprint of blockchain consensus mechanisms beyond proof-of-work," in *2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C)*. IEEE, 2021, pp. 1135–1144.
- [4] A. A. Khalil, J. Franco, I. Parvez, S. Uluagac, H. Shahriar, and M. A. Rahman, "A literature review on blockchain-enabled security and operation of cyber-physical systems," in *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 2022, pp. 1774–1779.
- [5] "Scalability - Bitcoin Wiki," <https://en.bitcoin.it/wiki/Scalability/>, July 2018.
- [6] J. Garzik, "Bitcoin Improvement Proposal 102," <https://github.com/bitcoin/bips/blob/master/bip-0102.mediawiki>, 2015.
- [7] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-ng: A scalable blockchain protocol," in *13th USENIX Symposium on Networked Systems Design and Implementation*, 2016, pp. 45–59.
- [8] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *ACM Proceedings of SIGSAC*, 2016, pp. 17–30.
- [9] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, "Omniledger: A secure, scale-out, decentralized ledger via sharding," in *IEEE Symposium on Security and Privacy (S&P)*, 2018.
- [10] A. Secure, "The zilliqa project: A secure, scalable blockchain platform," 2018.
- [11] M. Castro, B. Liskov et al., "Practical byzantine fault tolerance," in *OSDI*, vol. 99, 1999, pp. 173–186.
- [12] "Ethereum to combine casper and sharding upgrades," <https://cointelegraph.com/news/ethereum-to-combine-casper-and-sharding-upgrades>, June 2018.
- [13] A. Gangwal, H. R. Gangavalli, and A. Thirupathi, "A survey of layer-two blockchain protocols," *Journal of Network and Computer Applications*, vol. 209, p. 103539, 2023.
- [14] A. A. Khalil, M. A. Rahman, and H. A. Kholidy, "Fakey: Fake hashed key attack on payment channel networks," in *2023 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2023.

- [15] R. Nourmohammadi and K. Zhang, "Sharding and its impact on fork probability," in *2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETBlockchain)*. IEEE, 2022, pp. 1–6.
- [16] M. Zamani, M. Movahedi, and M. Raykova, "Rapidchain: Scaling blockchain via full sharding," in *ACM Proceedings of the SIGSAC Conference on Computer and Communications Security*, 2018.
- [17] J. Wang and H. Wang, "Monoxide: Scale out blockchains with asynchronous consensus zones," in *16th USENIX Symposium on Networked Systems Design and Implementation*, 2019, pp. 95–112.
- [18] E. Buchman, "Tendermint: Byzantine fault tolerance in the age of blockchains," Ph.D. dissertation, University of Guelph, 2016.
- [19] K. J. O'Dwyer and D. Malone, "Bitcoin mining and its energy footprint," *Proceedings of the 25th ISSC*, 2014.
- [20] E. Budish, "The economic limits of bitcoin and the blockchain," National Bureau of Economic Research, Tech. Rep., 2018.
- [21] "Bitcoin mining hardware comparison," <https://www.buybitcoinworldwide.com/mining/hardware/>, July 2019.
- [22] A. De Vries, "Bitcoin's growing energy problem," *Joule*, vol. 2, no. 5, pp. 801–805, 2018.
- [23] S. Ross, *A first course in probability*. Pearson, 2014.
- [24] M. Alharby and A. van Moorsel, "Blocksim: An extensible simulation tool for blockchain systems," *Frontiers in Blockchain*, vol. 3, p. 28, 2020.
- [25] A. Hafid, A. S. Hafid, and M. Samih, "New mathematical model to analyze security of sharding-based blockchain protocols," *IEEE Access*, vol. 7, pp. 185 447–185 457, 2019.
- [26] G. Bracha, "An $O(\log n)$ expected rounds randomized byzantine generals protocol," *Journal of the ACM (JACM)*, vol. 34, no. 4, 1987.
- [27] C. Decker, J. Seidel, and R. Wattenhofer, "Bitcoin meets strong consistency," in *ACM Proceedings of the 17th International Conference on Distributed Computing and Networking*, 2016, p. 13.
- [28] R. Pass and E. Shi, "Hybrid consensus: Efficient consensus in the permissionless model," in *LIPICs-Leibniz International Proceedings in Informatics*, 2017.
- [29] E. K. Kogias, P. Jovanovic, N. Gailly, I. Khoffi, L. Gasser, and B. Ford, "Enhancing bitcoin security and performance with strong consistency via collective signing," in *USENIX Security Symposium*, 2016.
- [30] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *Proceedings of the 26th Symposium on Operating Systems Principles*, 2017.
- [31] G. Danezis and S. Meiklejohn, "Centrally banked cryptocurrencies," *CoRR*, vol. abs/1505.06895, 2015. [Online]. Available: <http://arxiv.org/abs/1505.06895>
- [32] S. Li, M. Yu, S. Avestimehr, S. Kannan, and P. Viswanath, "Polyshard: Coded sharding achieves linearly scaling efficiency and security simultaneously," *arXiv preprint arXiv:1809.10361*, 2018.
- [33] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and S. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security and privacy," *arXiv preprint arXiv:1806.00939*, 2018.
- [34] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi, "Towards scaling blockchain systems via sharding," *arXiv preprint arXiv:1804.00399*, 2018.
- [35] G. Wang, Z. J. Shi, M. Nixon, and S. Han, "Sok: Sharding on blockchain," in *ACM Proceedings of the 1st Conference on Advances in Financial Technologies*, 2019, pp. 41–61.
- [36] 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 of Things Journal*, vol. 8, no. 6, pp. 4291–4304, 2020.
- [37] V. Buterine, "Vitalik's annotated ethereum 2.0 spec," <https://notes.ethereum.org/@vbuterin/SkeyEI3xv>, 2022.
- [38] G. Xu, H. Bai, J. Xing, T. Luo, N. N. Xiong, X. Cheng, S. Liu, and X. Zheng, "Sg-pbft: A secure and highly efficient distributed blockchain pbft consensus algorithm for intelligent internet of vehicles," *Journal of Parallel and Distributed Computing*, vol. 164, pp. 1–11, 2022.
- [39] W. Li, C. Feng, L. Zhang, H. Xu, B. Cao, and M. A. Imran, "A scalable multi-layer pbft consensus for blockchain," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 5, pp. 1146–1160, 2020.
- [40] J. Yun, Y. Goh, and J.-M. Chung, "Trust-based shard distribution scheme for fault-tolerant shard blockchain networks," *IEEE Access*, vol. 7, pp. 135 164–135 175, 2019.
- [41] M. Platt and P. McBurney, "Sybil in the haystack: a comprehensive review of blockchain consensus mechanisms in search of strong sybil attack resistance," *Algorithms*, vol. 16, no. 1, p. 34, 2023.
- [42] L. S. Sankar, M. Sindhu, and M. Sethumadhavan, "Survey of consensus protocols on blockchain applications," in *4th IEEE International Conference on Advanced Computing and Communication Systems*, 2017.
- [43] M. Conti, E. S. Kumar, C. Lal, and S. Ruj, "A survey on security and privacy issues of bitcoin," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3416–3452, 2018.
- [44] M. Kedziora, P. Kozlowski, and P. Jozwiak, "Security of blockchain distributed ledger consensus mechanism in context of the sybil attack," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer, 2020, pp. 407–418.
- [45] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoin's peer-to-peer network," in *Proceedings of the 24th USENIX Conference on Security Symposium*, 2015, p. 129–144.