Bankrupting Sybil Despite Churn

Diksha Gupta$^1$, Jared Saia$^2$, and Maxwell Young$^3$

$^1$Dept. of Computer Science, University of New Mexico, NM, USA
dgupta@unm.edu

$^2$Dept. of Computer Science, University of New Mexico, NM, USA
saia@cs.unm.edu

$^3$Computer Science and Engineering Dept., Mississippi State University, MS, USA
myoung@cse.msstate.edu

Abstract

A Sybil attack occurs when an adversary pretends to be multiple identities (IDs). Limiting the number of Sybil (bad) IDs to a minority permits the use of well-established tools for tolerating malicious behavior, such as protocols for Byzantine consensus and secure multiparty computation.

A popular technique for enforcing this minority is resource burning; that is, the verifiable consumption of a network resource, such as computational power, bandwidth, or memory. Unfortunately, prior defenses based on resource burning require non-Sybil (good) IDs to consume at least as many resources as the adversary, unless the rate of churn for good IDs is sufficiently low. Since many systems exhibit high churn, this is a significant barrier to deployment.

We present two algorithms that offer useful guarantees against a Sybil adversary under a broadly-applicable model of churn. The first is GoodJEst, which estimates the number of good IDs that join the system over any window of time, despite the adversary injecting bad IDs. GoodJEst is applicable to a broad range of system settings and here we demonstrate its use in our second algorithm, a new resource-burning Sybil defense called ERGO. Even under high churn, ERGO guarantees (1) there is always a minority of bad IDs in the system, and (2) when the system is under attack, the good IDs burn resources at a total rate which is sublinear in the adversary’s consumption.

To evaluate the impact of our theoretical results, we investigate the performance of ERGO alongside prior defenses that employ resource burning. Based on our experiments, we design heuristics that further improve the performance of ERGO by up to four orders of magnitude over these previous Sybil defenses.
1 Introduction

A Sybil attack occurs when a single adversary pretends to be multiple identities (IDs) [18]. One of the oldest defenses against Sybil attacks is resource burning, in which any ID that wishes to use network resources or participate in group decision making must first consume local resources in a verifiable manner [32]. A well-known example of resource burning is proof-of-work (PoW) [19], but several other methods exist (see Section 2).

Resource burning defends against any resource-bounded Sybil adversary. Unfortunately, a significant drawback is that resources must always be consumed, even when the system is not under attack. This non-stop resource burning translates into a substantial energy and, ultimately, a monetary cost; this waste has often been highlighted [15,41,42].

Reducing this waste is challenging, but prior work offers some hope [31]. Informally, a majority of non-Sybil (good) IDs can be maintained in the system, and the good IDs have a resource-burning rate that is asymptotically smaller than the resource-burning rate of the adversary when the system is under attack. In other words, the adversary suffers a larger cost than the defenders for attacking the system; this property both mitigates the magnitude of attacks, while also serving as a deterrent.

Unfortunately, this favorable resource-burning rate fails to hold in settings where the rate at which system membership changes—often referred to as the churn rate—is high. A high churn rate characterizes many systems [20,64,67] where there is little to no admission control, making such systems especially vulnerable to a Sybil attack. Therefore, the inability to tolerate high churn poses a significant challenge to defending those systems which need it most.

A natural question is: Can we design a defense against the Sybil attack while preserving an advantageous spend rate even under a high rate of churn?

1.1 Our Contributions

We demonstrate such a defense, which consists of two algorithms. The first is **GoodJEst**: a general-purpose method for deriving bounds on the rate at which good IDs join the system, $J^G$, despite an adversary who injects bad IDs. Informally, we define possibly unknown parameters $\alpha, \beta \geq 1$ that characterize how $J^G$ changes over time, and the burstiness of good-ID arrivals, respectively (see Section 4 for details). Assuming the number of good IDs is always at least $n_0$, the following results hold with probability $1 - O(1/n_0)$ over a number of ID joins and departures that is polynomial in $n_0$. 
**Theorem 1.** For any interval with good join rate $J^G$, GOODJEST output $\tilde{J}^G$ at the end of previous interval such that:

$$\frac{1}{350\alpha^3\beta^2} J^G \leq \tilde{J}^G \leq 100\alpha^3\beta^4 J^G$$

Let the **good spend rate** be the cost over all good IDs per second, where this cost is due to resource burning. Similarly, let $T$ denote the **adversary’s spend rate**; that is, the cost over all bad IDs per second.

Assume a resource-bounded adversary who controls at most an $\kappa$-fraction of the system resources, for $\kappa > 0$. Our second algorithm, ERGO, leverages GOODJEST to provide the following guarantees.

**Theorem 2.** For $\kappa \leq 1/18$, ERGO ensures that the good spend rate is $O(\alpha^6\beta^4 \left(\sqrt{T(J^G+1)} + J^G\right))$ and guarantees that the fraction of bad IDs in the system is always less than 1/6.

We note that $\kappa = 1/18$ for ease of analysis, and larger values of $\kappa$ can be tolerated.

Finally, we validate our theoretical results by comparing ERGO against prior PoW defenses using real-world data from several networks. We find that ERGO performs up to 2 orders of magnitude better than previous defenses, according to our simulations (Section 6.2). Using insights from these first experiments, we engineer and evaluate several heuristics aimed at further improving the performance of ERGO. Our best heuristic performs up to 3 orders of magnitude better than previous algorithms for large-scale attacks (Section 6.3).

### 2 Our Model

We now describe a general network model that aligns with many permissionless systems, including the work in [31]. The system consists of virtual **identifiers (IDs)**, where each ID is either **good** if it obeys protocol, or **bad** if it is controlled by the Sybil adversary (or just **adversary**).

**Resource-Burning Challenges.** IDs can construct resource-burning challenges of varying hardness, whose solutions cannot be stolen or pre-computed; some examples are discussed in Section 3.3. A **k-hard challenge** for any integer $k \geq 1$ imposes a resource cost of $k$ on the challenge solver. Our results are agnostic to the type of challenges employed, either those discussed above or new resource-burning schemes available for future use.
Coordination. For simplicity of presentation, we assume that there is a single server running our algorithms. However, in Section 7, we illustrate how the server can be replaced with a small committee under some additional assumptions, thus allowing for our algorithm to execute in decentralized network settings.

A round is the amount of time it takes to solve a 1-hard puzzle plus time for communication between the server and corresponding ID for issuing the puzzle and returning a solution. Good IDs are assumed to have clocks that are closely synchronized; techniques for synchronizing on the order of milliseconds are known [44,46] and suffice for our purposes.

Adversary. A single adversary controls all bad IDs. This pessimistically represents perfect collusion and coordination by the bad IDs. Bad IDs may arbitrarily deviate from our protocol, including sending incorrect or spurious messages. The adversary can send messages to any ID at will, and can read the messages diffused by good IDs before sending its own. It knows when good IDs join and depart, but it does not know the private bits of any good ID.

The adversary is resource-bounded: in any single round where all IDs are solving puzzles, the adversary can solve an $\kappa$-fraction of the puzzles; this is common in past PoW literature [2,25,53,65].

Joins and Departures. Every join and departure event is assumed to occur at a unique point in time. In practice, this means that the events must be serialized by, for example, the server or committee.

Whenever the adversary decides to cause a good ID departure event, a good ID is selected independently and uniformly at random to depart from the set of good IDs in the system.

Departing good IDs announce their departure to the network. In practice, each good ID can issue “heartbeat messages” to the server that indicate this ID is still alive; the absence of a heartbeat message is interpreted as a departure by the corresponding ID.

Every joining ID is treated as a new ID. We can ensure that every joining ID is given a new name by concatenating a join-event counter to the name chosen by the ID. Similar to previous works [9,28,34], we assume that every joining ID knows at least one good ID in order to be bootstrapped into the system.

We define $n_0$ to be the minimum number of good IDs in the system at any point. We define the system lifetime to be the duration over which $n_0\gamma$ joins and departures occur, for any fixed constant $\gamma > 0$. 


2.1 Epochs, Smoothness, and High Churn

Our analysis partitions the lifetime of the system into continuous epochs. For any time \( \tau \), let \( G_\tau \) be the set of good IDs at time \( \tau \). Then we have the following definition:

**Definition 3.** For all \( i \geq 1 \), epoch \( i \) begins at time \( t = 0 \) if \( i = 1 \), or at the time \( t \) when epoch \( i - 1 \) ends otherwise. Epoch \( i \) ends at the smallest \( t' \) such that \( |G_{t'} \oplus G_t| \geq (3/4)|G_t| \).

Let \( \hat{\rho}_j \) be the join rate of good IDs (i.e., good join rate) in epoch \( j \); that is, the number of good IDs that join in epoch \( j \) divided by the number of seconds in epoch \( j \).

**Definition 4.** For any \( \alpha \geq 1 \) and \( 1 \leq \beta \leq \sqrt{\frac{5}{56} n_0 - 1} \), we define the following notion of smoothness:

- **\( \alpha \)-smoothness:** \((1/\alpha)\hat{\rho}_{j-1} \leq \hat{\rho}_j \leq \alpha \hat{\rho}_{j-1} \).
- **\( \beta \)-smoothness:** For any duration of \( \ell \) seconds in the epoch, the number of good IDs that join is at least \( \lfloor \ell \hat{\rho}_j / \beta \rfloor \) and at most \( \lceil \beta \ell \hat{\rho}_j \rceil \). Also, the number of good IDs that depart during this duration is at most \( \lceil \beta \ell \hat{\rho}_j \rceil \).

**High Churn.** We emphasize that Definition 4 captures a compelling notion of high churn. First, the notion of \( \alpha \)-smoothness captures any good join rate between consecutive epochs, since there always exists a parameter \( \alpha \) that satisfies the definition. Therefore, the good join rate may change rapidly under this smoothness definition. For example, suppose that the \( \hat{\rho}_j \) increases by even a small constant factor of, say, \( \alpha = 2 \) over consecutive values \( j \); this can occur if the same number of good IDs join in each epoch, but over half the time of the previous epoch. In this case, the good join rate exhibits exponential increase over these epochs. Similarly, the good join rate may decrease exponentially. Second, while \( \hat{\rho}_j \) is the good join rate over the entire epoch \( j \), \( \beta \)-smoothness allows for deviations within that epoch.

3 Related Work

3.1 Model Assumptions

A common assumption in the related literature is that the number of good IDs is fixed at a sufficiently large value or can vary by at most a constant

\footnote{The bound on \( \beta \) is crucial to the proof of Lemma 10 where we obtain a lower bound on the number of good IDs that join over an interval.}
Several results by Augustine et al. [4–8] address robust distributed computation under a model of churn where the system size is static (or varies by a constant factor), and system membership changes by up to a constant factor. Guerraoui et al. [28] address a challenging setting where the system size can vary polynomially as a function of some initial quantity of good IDs; we address the same challenge here. Unlike our result, these past works do not tune resource costs to the amount of churn observed.

### 3.2 Sybil Attack

There is large body of literature on defending against the Sybil attack [18]; for example, see surveys [35, 50, 57], and additional work documenting real-world Sybil attacks [54, 66, 69].

**Radio-Resource Testing.** In a wireless network with multiple communication channels, Sybil attacks can be mitigated via *radio-resource testing* which relies on the inability of the adversary to listen to many channels simultaneously [26, 27, 51]. However, this approach may fail if the adversary can monitor most or all of the channels. Furthermore, even in the absence of attack, radio-resource testing requires testing at fixed periods of time.

**Social Network Properties.** Several results that leverage social networks for Sybil resistance [49, 68, 71]. However, social-network information may not be available in many settings. Another idea is to use network measurements to verify the uniqueness of IDs [24, 39, 62], but these techniques rely on accurate measurements of latency, signal strength, or round-trip times, and this may not always be possible. Containment strategies are explored in overlays [17, 61], but these results do not ensure a bound on the fraction of bad IDs.

### 3.3 Resource Burning

A number of such resource burning schemes exist. *Computational puzzles* have been used in several prior works for defending against the Sybil attack [2, 38, 53]. *Proof of Exercise (PoE)* is an alternative to hash-based PoW puzzles, and it requires solving computationally-intensive scientific problems [63]. Another example is *Proof of Space-Time*, which involves proving an allocation of storage capacity for a certain amount of time [52].
In the following, $t'$ is the current time and $S_x$ is the set of IDs in the system at time $x$.

$t ←$ time at system initialization.

$\hat{J}^G ← |S_t|$ divided by time required for initialization.

Repeat forever: whenever $|S_{t'} + S_t| \geq \frac{5}{2} |S_{t'}|$, do

1. $\hat{J}^G ← |S_{t'}|/(t' - t)$.
2. $t ← t'$.

Figure 1: Pseudocode for GOODJEst.

3.4 Guaranteed Spend Rate Relative to Attacker

In \cite{29} and \cite{31}, Gupta et al. proposed two algorithms CCOM and GMComm that ensure that the fraction of bad IDs is always small, with respective good spend rates of $O(T + J_G)$ and $O(J^G + \sqrt{T(J^G + 1)})$. Unfortunately, this latter spend rate only holds in the case where (1) GMCom always knows the join rate of good IDs; and (2) there is a fixed constant amount of time that separates all join events by good IDs (i.e., non-bursty arrivals). We note that ERGO does not require these assumptions.

4 GOODJEst

GOODJEst estimates bounds on the join rate of good IDs in any system where at most a small constant fraction of bad IDs; for simplicity, we assume the fraction is less than $1/6$, but larger constants can be tolerated. In permissioned systems, this bound may be guaranteed by an admission-control mechanism. In permissionless systems, this bound can be enforced by a decentralized algorithm and, in Section 5, we present such an approach. However, first we present some preliminary definitions for quantifying churn.

4.1 Description of GOODJEst

GOODJEst starts out with a value for $\hat{J}^G$ equal to the number of IDs at system initialization divided by the total time taken for initialization, where initialization consists of assigning each ID in the system a 1-difficult RB-Challenge to be solved over the next one round. The value $t$ is set to the system start time. Throughout the protocol, $t$ will equal the last time that
\( \tilde{J}^G \) was updated, and \( t' \) will be the current time.

There are two questions that must be addressed in designing GoodJEst. First, at what points in time should \( \tilde{J}^G \) be updated? This occurs whenever the system membership has changed by a constant factor with respect to the current system size. In particular, \( \tilde{J}^G \) is updated when \( |S_{t'} \oplus S_t| \geq \frac{5}{8} |S_{t'}| \) holds true. Since join and departure events are serialized, this is equivalent to the property that \( |S_{t'} \oplus S_t| = \lceil \frac{5}{8} |S_{t'}| \rceil \). We refer to \( (t, t') \) as an interval. The execution of GoodJEst divides time into consecutive, disjoint intervals.

Second, what is entailed in updating \( \tilde{J}^G \)? This is done by setting \( \tilde{J}^G \) to the current system size divided by the amount of time since the last update to \( \tilde{J}^G \); that is, \( \tilde{J}^G \leftarrow |S_{t'}|/(t' - t) \).

### 4.2 Proof of Theorem 1

Recall that GoodJEst imposes a division of time into intervals (Section 4.1). We say that interval intersects an epoch if there is a point in time belonging to both the interval and the epoch. In this section, for any time \( t \), let \( S_t \) and \( G_t \) denote the set of all IDs and set of good IDs, respectively, in the system at time \( t \). The following proofs hold with high probability in \( n_0 \) over all intervals, assuming the Population invariant always holds.

**Lemma 5.** An interval intersects at most two epochs.

*Proof.* Assume that some interval starts at time \( t_0 \) and intersects at least three epochs; we will derive a contradiction. Given this assumption, there must be at least one epoch entirely contained within the interval. Consider the earliest such epoch, and let it start at time \( t_1 \geq t_0 \) and end at time \( t_2 > t_1 \). Observe that:

\[
|S_{t_2} \oplus S_{t_0}| \geq |G_{t_2} \oplus G_{t_1}| \geq \frac{3}{4} |G_{t_2}| \geq \left( \frac{3}{4} \right) \left( \frac{5}{6} \right) |S_{t_2}| = \left( \frac{5}{8} \right) |S_{t_2}|
\]

In the above, step one holds since \( |S_{t_2} \oplus S_{t_0}| \geq |G_{t_2} \oplus G_{t_1}| \); step two holds by the definition of an epoch; and the second last step holds by the Population Invariant.

But the above inequalities show that \( |S_{t_2} \oplus S_{t_0}| \geq \frac{5}{8} |S_{t_2}| \). Therefore, the interval ends by time \( t_2 \), and there can be no third epoch intersecting the interval; this contradiction completes the argument. \( \square \)

The aim of the next set of lemmas is to derive upper and lower bounds on the number of good IDs that can join over an interval as a function of the the system size, and \( \alpha \) and \( \beta \).
Lemma 6. Fix an interval that starts at time \( t \) and ends at time \( t' \). Let \( a \) be the number of good IDs that have joined the system between time \( t \) and \( t' \). Then, \( a \leq 23|S_t| + 4 \).

Proof. First, note that:

\[
\left\lceil \frac{5}{8}|S_{t'}| \right\rceil = |S_{t'} \oplus S_t| \\
= |S_{t'} - S_t| + |S_t - S_{t'}| \\
\geq |S_{t'} - S_t| \\
\geq |G_{t'} - G_t|
\]

where the first step holds by the definition of an interval and the fact that all join and leave events occur at unique times, i.e. events are linearized. The last step holds since sets of good and bad IDs are disjoint. Thus, we have

\[
|G_{t'} - G_t| \leq \left\lceil \frac{5}{8}|S_{t'}| \right\rceil \leq \frac{3}{4}|G_{t'}| + 1
\]

In the above, the second step holds by the population invariant, since \( \frac{|G_{t'}|}{|S_{t'}|} > \frac{5}{6} \) implies that \( |S_{t'}| < \frac{6}{5}|G_{t'}| \). Then, \( \left\lceil \frac{5}{8}|S_{t'}| \right\rceil \leq \frac{6}{5}|S_{t'}| + 1 < \frac{3}{4}|G_{t'}| + 1 \).

This gives our first key inequality:

\[
|G_{t'} - G_t| \leq \frac{3}{4}|G_{t'}| + 1 \tag{1}
\]

Next, let \( d \) be the number of good IDs that have departed in the interval. We note that

\[
d \leq 5|G_t| \tag{2}
\]

To see this, recall that when the departure of a good ID occurs, the ID that departs is selected uniformly at random from the set of all good IDs. Thus, Inequality \( 1 \) implies that a departing good ID will be from the set \( G_t \) with probability at least \( 1/4 \).

Let the random variable \( X \) be the number of IDs in \( G_t \) that have departed during the interval. By the above, \( E(X) \geq \frac{1}{4}d \). Additionally, \( X \) stochastically dominates a simpler random variable that counts the number of successes when there are \( d \) independent trials, each succeeding with probability \( \frac{1}{4} \). Hence, by Chernoff bounds, when \( d \geq |G_t| \), \( X \geq \frac{1}{5}d \), with probability of failure that is \( O(e^{-cw}) \), for some constant \( c > 0 \). This probability is at most \( n_0^{-\Lambda - 1} \) for \( n_0 \) sufficiently large. Clearly, \( X \leq |G_t| \). So by the above, we have that, with high probability, \( \frac{1}{5}d \leq |G_t| \), which gives inequality \( 2 \).
Finally, let $a$ be the number of good IDs added during the interval. Since the number of new good IDs in $S_{t'}$ is $a - d$, then $|G_{t'} - G_t| \geq a - d$. Thus, we have:

$$a \leq |G_{t'} - G_t| + d$$

$$\leq \left(\frac{3}{4}|G_{t'}| + 1\right) + 5|G_t|$$

$$\leq \frac{3}{4}(|G_t| + a) + 1 + 5|G_t|$$

$$\leq \frac{23}{4}|G_t| + \frac{3}{4}a + 1$$

In the above, the second step follows by applying inequalities 1 and 2. Finally, the lemma follows by solving the above inequality for $a$, to get $a \leq 23|G_t| + 4 \leq 23|S_t| + 4$.

**Lemma 7.** Fix an interval. Suppose $a$ is the number of good IDs that join and $d$ is the number of good IDs that depart during this interval. Then, $d \leq \beta^2(a + 2) + 2$.

**Proof.** Recall from Lemma 5 that an interval intersects at most two epochs. Suppose $\hat{\rho}, \hat{\rho}'$ is the good join rate over the two epochs that intersect the current interval, and $\hat{\ell}, \hat{\ell}'$ be the lengths of the intersection. Then, from $\beta$-smoothness, we have:

$$a \geq \left\lfloor \frac{\hat{\rho}\hat{\ell}}{\beta} \right\rfloor + \left\lfloor \frac{\hat{\rho}'\hat{\ell}'}{\beta} \right\rfloor \geq \frac{\hat{\rho}\hat{\ell} + \hat{\rho}'\hat{\ell}'}{\beta} - 2 \tag{3}$$

Similarly, we can upper bound the number of departures using the $\beta$-smoothness as:

$$d \leq \left\lceil \beta\hat{\rho}\hat{\ell} \right\rceil + \left\lceil \beta\hat{\rho}'\hat{\ell}' \right\rceil \leq \beta^2(a + 2) + 2$$

where the last step follows from inequality 3.

**Lemma 8.** Fix an interval that starts at time $t$ and ends at time $t'$. Then, $|S_{t'}| \geq \frac{6}{13}|S_t|$.

**Proof.** We know that $|S_{t'}|$ can reduce by at most $|S_{t'} \oplus S_t|$ over the interval. Thus:

$$|S_{t'}| \geq |S_t| - |S_{t'} \oplus S_t|$$

$$\geq |S_t| - \left(\frac{5}{8}|S_{t'}| + 1\right)$$
where the second inequality follows from the definition of an interval. Then, on solving the above inequality for $|S_{t'}|$, we get:

$$|S_{t'}| \geq \frac{8}{13} (|S_t| - 1) \geq \frac{8}{13} \left( \frac{3}{4} |S_t| \right) \geq \frac{6}{13} |S_t|$$

where the second inequality holds for $n_0 \geq 4$.

**Lemma 9.** Fix an interval that starts at time $t$ and ends at time $t'$. Let $a$ be the number of good IDs that have joined the system during the interval. Then,

$$a \geq \left( 1 + \frac{1}{12(1+\beta^2)} \right) |S_t| - 2.$$

**Proof.** Let $d$ be the number of good IDs that depart from the system and $a$ be the number of good IDs that join the system in the current interval. Then, we obtain bounds on $|G_{t'} \oplus G_t|$ as:

$$|G_{t'} \oplus G_t| \leq a + d$$

$$\leq a + \beta^2 (a + 2) + 2$$

$$\leq (1 + \beta^2)(a + 2)$$

where the second to last step follows from Lemma 7.

Let $B_{t'}$ be the set of bad IDs in the system, then we can lower bound $|G_{t'} \oplus G_t|$ as:

$$|G_{t'} \oplus G_t| = |S_{t'} \oplus S_t| - |B_{t'} \oplus B_t|$$

$$\geq \frac{5}{8} |S_{t'}| - |B_{t'} \oplus B_t|$$

$$\geq \frac{5}{8} |S_{t'}| - (|B_{t'} - B_t| + |B_t - B_{t'}|)$$

$$\geq \frac{5}{8} |S_{t'}| - |S_{t'}| - \frac{|S_t|}{6} - \frac{|S_t|}{6}$$

$$\geq \frac{5}{8} |S_{t'}| - |S_{t'}| - \frac{13}{6} \left( \frac{|S_t|}{6} \right)$$

$$\geq \left( \frac{5}{8} - \frac{1}{6} - \frac{13}{36} \right) |S_{t'}|$$

$$\geq \frac{|S_{t'}|}{12}$$

The first step holds since the sets $G$ and $B$ are disjoint, and the second step follows from the definition of an interval and the fact that $|B_{t'} \oplus B_t| = |B_{t'} - B_t| + |B_t - B_{t'}|$. The fourth inequality follows from the population invariant: $|B_{t'} - B_t| \leq \frac{|S_{t'}}{6}$ and $|B_t - B_{t'}| \leq \frac{|S_t|}{6}$. The fifth step follows from Lemma 8.
Finally, combining inequalities 4 and 5 we get:

\[(1 + \beta^2)(a + 2) \geq \frac{|S_{t'}|}{12}\]

On solving the above for \(a\), we get the result.

**Lemma 10.** Fix an interval that starts at time \(t\) and ends at time \(t'\). Let \(a\) be the number of good IDs that have joined the system during the interval. Then, \(a \geq 8\).

**Proof.** Let \(a\) be the number of good IDs that join and \(d\) be the number of good IDs that depart over the interval. We begin by obtaining a lower bound on \(a\), which is used later in the proof.

Recall from the definition of an interval that:

\[
a \geq \frac{5}{8} |S_{t'}| - d
\]

\[
\geq \frac{5}{8} n_0 - (\beta^2(a + 2) + 2)
\]

\[
\geq \frac{5}{8} n_0 - \left(\frac{5}{80} n_0 - 1\right)(a + 2) - 2
\]

where the second step follows from Lemma 7 and the third step follows from the fact that \(\beta \leq \sqrt{\frac{5}{80} n_0 - 1}\).

Then, on solving the above inequality for \(a\), we get the result.

**Lemma 11.** Fix an interval. Let \(J^G\) be the join rate of good IDs over this interval. Then, at the end of the interval:

\[
\frac{1}{104} J^G \leq \tilde{J}^G \leq 30\beta^2 J^G
\]

**Proof.** Let \(a\) be the number of good IDs that join and \(d\) be the number of good IDs that depart over the interval.

**Lower bound:** Note that at the end of the interval, we set the estimate of good join rate as:

\[
J = \frac{|S_{t'}|}{t' - t} \geq \frac{(6/13)|S_t|}{t' - t} \geq \frac{6}{13} \left(\frac{a - 4}{23}\right) \geq \frac{1}{52} \left(\frac{a - 4}{t' - t}\right) \geq \frac{1}{52} \left(\frac{a - a/2}{t' - t}\right) \geq \frac{J^G}{104}
\]

The second inequality follows from Lemma 8. The third inequality follows from Lemma 6 and second last step follows from Lemma 10.
Upper bound: Similarly:

\[ \tilde{j} = \frac{|S_{t'}|}{t' - t} \leq \frac{12(1 + \beta^2)(a + 2)}{(t' - t)} \leq 12(1 + \beta^2) \left( \frac{a}{t' - t} + \frac{2}{t' - t} \right) \leq 12(1 + \beta^2) \left( \frac{a}{t' - t} + \frac{a}{4(t' - t)} \right) \leq 12(1 + \beta^2) \left( \frac{5a}{4(t' - t)} \right) \leq 12(1 + \beta^2) \left( \frac{5J^G}{4} \right) \leq 15(1 + \beta^2)J^G \leq 30\beta^2J^G \]

where the second step follows from Lemma 9, the fourth step follows from Lemma 10, and the last inequality holds since \( \beta \geq 1 \) from the \( \beta \)-smoothness property.

**Lemma 12.** Fix an interval and consider an epoch that intersects this epoch. Suppose \( J^G \) is the join rate of good IDs over this interval, and \( \hat{\rho} \) is the join rate over this entire epoch. Then:

\[ \frac{4}{5\alpha \beta} \hat{\rho} \leq J^G \leq \frac{4}{3}(1 + \alpha)\beta \hat{\rho} \]

**Proof.** Let \( \ell \) denote the length of the interval. We know that interval intersects at most two epochs. Let \( \hat{\rho} \) and \( \hat{\rho}' \) be the join rate of good IDs over the two epochs intersecting the interval over say lengths \( \hat{\ell} \) and \( \hat{\ell}' \), respectively. Then, by \( \beta \)-smoothness property, we have:

\[
J^G \geq \frac{1}{\ell} \left( \left\lfloor \hat{\rho} \hat{\ell} \beta \right\rfloor + \left\lfloor \hat{\rho}' \hat{\ell}' \beta \right\rfloor \right) \\
\geq \frac{1}{\beta} \left( \hat{\rho} \left( \frac{\hat{\ell}}{\ell} \right) + \frac{\hat{\rho}'}{\alpha} \left( \frac{\hat{\ell}'}{\ell} \right) - \frac{2}{\ell} \right) \\
\geq \frac{\hat{\rho}}{\beta \alpha} \left( \frac{\hat{\ell} + \hat{\ell}'}{\ell} \right) - \frac{(J^G \ell/4)}{\ell} \\
\geq \frac{\rho}{\alpha \beta} - \frac{J^G}{4}
\]
where the second step follows from the $\alpha$-smoothness property, and the third inequality follows from Lemma 10. Then, on solving the above inequality, we get the lower bound.

Next, we prove the upper bound. Again, using the $\beta$-smoothness property, we have:

\[
J^G \leq \frac{1}{\ell} \left( \left\lceil \beta \hat{\rho} \hat{\ell} \right\rceil + \left\lceil \beta \hat{\rho}' \hat{\ell}' \right\rceil \right)
\]
\[
\leq \beta \left( \hat{\rho} \left( \frac{\hat{\ell}}{\ell} \right) + \alpha \hat{\rho} \left( \frac{\hat{\ell}'}{\ell} \right) \right) + \frac{2}{\ell}
\]
\[
\leq \beta \left( \hat{\rho} \left( \frac{\hat{\ell}}{\ell} \right) + \alpha \hat{\rho} \left( \frac{\hat{\ell}'}{\ell} \right) \right) + \frac{\left( J^G \ell / 4 \right)}{\ell}
\]
\[
\leq (1 + \alpha) \beta \hat{\rho} + \frac{J^G}{4}
\]

where the second inequality follows from the $\alpha$-smoothness property, and the third inequality follows from Lemma 10. Then, on solving the above, we obtain the upper bound.

Finally, we restate and prove the main theorem for our GOODJEST function:

**Theorem 1.** For any interval with good join rate $J^G$, GOODJEST output $\tilde{J}^G$ at the end of previous interval such that:

\[
1 / \left( 350 \alpha^3 \beta^2 \right) J^G \leq \tilde{J}^G \leq 100 \alpha^3 \beta^4 J^G
\]

**Proof.** Fix an interval $i$. We set $\tilde{J}^G$ for interval $i$ at the end of interval $i - 1$. Let $J^G_{i-1}$ be the join rate of good IDs in interval $i - 1$, $J^G_i$ be the join rate of good IDs in interval $i$ and $\ell_i$ denote the length of interval $i$. By Lemma 5, intervals $i$ and $i - 1$ will intersect at most 4 epochs.

Let $\hat{\rho}_1$ and $\hat{\rho}_2$ be the join rate of good IDs over the two epochs intersecting interval $i - 1$ over say lengths $\hat{\ell}_1$ and $\hat{\ell}_2$, respectively, and $\hat{\rho}_3$ and $\hat{\rho}_4$ be the join rate of good IDs over the two epochs intersecting interval $i$ over say lengths $\hat{\ell}_3$ and $\hat{\ell}_4$, respectively.

**Upper Bound.** By Lemma 12 the good-ID join rate in interval $i - 1$ is given by:

\[
J^G_{i-1} \geq \frac{4}{5 \alpha \beta} \hat{\rho}^2
\]
and that for interval $i$ is:

$$J_i^G \leq \frac{4}{3} (1 + \alpha) \beta \hat{\rho}_3$$

$$\leq \frac{4}{3} (1 + \alpha) \beta \alpha \hat{\rho}_2$$

$$\leq \frac{4}{3} (1 + \alpha) \beta \left( \frac{5 \alpha \beta}{4} J_{i-1}^G \right)$$

$$= \frac{5}{3} (1 + \alpha) \alpha^2 \beta^2 J_{i-1}^G$$

$$\leq \frac{10}{3} \alpha^3 \beta^2 J_{i-1}^G$$

where the second step follow from $\alpha$-smoothness property, the third step follows from inequality [6] and the final inequality holds since $\alpha \geq 1$ from the $\alpha$-smoothness property. Finally, using the upper bound from Lemma [11] we obtain our upper bound.

**Lower Bound.** Similarly, by Lemma [12] the good-ID join rate in interval $i - 1$ is given by:

$$J_{i-1}^G \leq \frac{4}{3} (1 + \alpha) \beta \hat{\rho}_2 \quad (7)$$

and for interval $i$ is:

$$J_i^G \geq \frac{4}{5 \alpha \beta} \hat{\rho}_3$$

$$\geq \frac{4}{5 \beta \alpha^2} \hat{\rho}_2$$

$$\geq \frac{4}{5 \beta \alpha^2} \left( \frac{3}{4(1 + \alpha) \beta} J_{i-1}^G \right)$$

$$\geq \frac{3}{5(1 + \alpha) \alpha^2 \beta^2 J_{i-1}^G}$$

$$\geq \frac{3}{10 \alpha^3 \beta^2 J_{i-1}^G}$$

where the second step follows from the $\alpha$-smoothness property, the third step follows from inequality [7] and the final inequality holds since $\alpha \geq 1$ from the $\alpha$-smoothness property. Finally, using the lower bound from Lemma [11] we obtain our lower bound. 

\[\square\]
5 ERGO - A Sybil Defense using GOODJEST

In this section, we describe an asymmetric Sybil defense algorithm - ERGO, for permissionless system with churn. We begin with the algorithm overview in Section 5.1. Then, we develop intuition for why this method yields the asymmetric property, and how it relies on a robust estimate of the good join rate obtained using GOODJEST in Section 5.2.

5.1 Overview of ERGO

We describe ERGO while referencing its pseudocode in Figure 2. Execution occurs over disjoint periods of time called iterations, and each iteration $i \geq 1$ consists of Steps 1 and 2.

The server initializes the system with an honest majority by issuing every ID a 1-difficult RB-challenge.

In Step 1, each joining ID must solve an RB-challenge of difficulty 1 plus the number of IDs that join within the last $1/\tilde{J}_G$ seconds, where $\tilde{J}_G$ is the current estimate of the join rate of good IDs obtained using GOODJEST.

This RB-Challenge difficulty approximates the ratio of the total join rate over the good join rate, which motivates the name ENTIRE BY RATE OF GOOD.

Step 1 lasts until the earliest point in time when the number of IDs that

---

**Entire by Rate of Good (ERGO)**

The server runs the following code:

$S_0 \leftarrow$ set of IDs that returned a valid solution to 1-difficult RB-challenge.

$\tilde{J}_G$ is maintained by running GOODJEST in parallel to the following code.

For each iteration $i = 1, \ldots$, do:

1. Each joining ID is assigned a RB-challenge of difficulty equal to 1 plus the number of IDs that have joined in the last $1/\tilde{j}_G$ seconds of the current iteration.

2. When number of joining and departing IDs in this iteration exceeds $|S_{i-1}|/11$, Purge:
   (a) Issue all IDs a RB-challenge of difficulty 1.
   (b) $S_i \leftarrow$ set of IDs solving this challenge within 1 round.

---

Figure 2: Pseudocode for ERGO.
Figure 3: Illustration relating epoch, interval and iteration. The notion of an *epoch* derives from our model of churn (Section 2). An interval is defined by GoodJEst, specifically the times at which it sets the variable $\tilde{J}_G^i$; An iteration is the defined by EnGO, specifically the duration between purges.

join and depart in iteration $i$ is at least $(1/11)|S_{i-1}|$. When Step 1 ends, the server performs a *purge* by issuing an RB-Challenge of difficulty 1 to all the IDs in the system, in Step 2(a). In Step 2(b), each ID must respond with a valid solution within 1 round. The server removes unresponsive or late-responding IDs from its whitelist.

5.2 Developing Intuition for the Asymmetric Property

Initially, the asymmetric result may be surprising, and so we offer intuition for this. Consider iteration $i$. In the absence of an attack, the entrance cost should be proportional to the good join rate. This is indeed the case since the puzzle difficulty is $O(1)$ corresponding to the number of (good) IDs that join within the last $1/\tilde{J}_G^i ≈ 1/J_G^i$ seconds.

In contrast, if there is a large attack, then the entrance-cost function imposes a significant cost on the adversary. Consider the case where a batch of many bad IDs is rapidly injected into the system. This drives up the entrance cost since the number of IDs joining within $1/\tilde{J}_G^i$ seconds increases.

More precisely, assume the adversary’s spending rate is $T = \xi J_{i}^{\text{all}}$, where $\xi$ is the entrance cost, and $J_{i}^{\text{all}}$ is the join rate for all IDs. For the good IDs, the spending rate due to the entrance cost is $\xi J_{i}^{G}$, and the spending rate due to the purge cost is $J_{i}^{\text{all}}$. Setting these to be equal, and solving for $\xi$, we get $\xi = J_{i}^{\text{all}} / J_{i}^{G}$; in other words, the number of IDs that have joined over the last $1/J_{i}^{G}$ seconds. This is the entrance cost function that best balances entrance and purge costs.

Spending rate of good IDs due to the entrance costs and purge costs is:

$$\xi J_{i}^{G} + J_{i}^{\text{all}} \leq 2J_{i}^{\text{all}} = 2\sqrt{(J_{i}^{\text{all}})^2} = 2\sqrt{J_{i}^{\text{all}}\xi J_{i}^{G}} = 2\sqrt{J_{i}^{G}T},$$

where the first inequality holds by our setting of $\xi$, the third step since $J_{i}^{\text{all}} = \xi J_{i}^{G}$, and the final step since $T = \xi J_{i}^{\text{all}}$. This informal analysis shows how knowledge of the good join rate can be used to reduce the good spend rate.
5.3 Proof of Theorem 2

We begin we proving that ERGO always maintains an honest majority in the system.

For any iteration $i$, we let $B_i$ and $G_i$ respectively denote the number of bad and good IDs in the system at the end of iteration $i$, and we let $N_i = B_i + G_i$.

Lemma 13. For all iterations $i \geq 0$, $B_i < N_i/3$.

Proof. Recall that the CA issues all IDs a 1–difficult RB-challenge prior to the first iteration. This reduces the fraction of bad IDs in the system to at most an $\alpha$–fraction, which is strictly less than a third of the total resources in the system. Thus, our lemma statement holds for $i = 0$.

For $i > 0$, note that the adversary has total resources less than $G_i/2$. Thus, after Step 2 that ends iteration $i$, we have $B_i < G_i/2$. Adding $B_i/2$ to both sides of this inequality, we get:

$$B_i < \frac{2}{3} \left( \frac{G_i}{2} + \frac{B_i}{2} \right) < \frac{N_i}{3}$$

where the last inequality holds since $N_i = G_i + B_i$. \hfill $\square$

Let $n^a_i, g^a_i, b^a_i$ denote the total, good, and bad IDs that arrive over iteration $i$. Similarly, let $n^d_i, g^d_i, b^d_i$ denote the total, good, and bad IDs that depart over iteration $i$.

Note that the server will always have an accurate value for all of these variables except for possibly $b^d_i$ since the bad IDs do not need to give notification when they depart — and, consequently, $n^d_i$, in these two cases, the CA may hold values which are underestimates of the true values.

Lemma 14. The fraction of bad IDs is always at most $1/2$.

Proof. Fix some iteration $i > 0$. Recall that an iteration ends when $n^a_i + n^d_i \geq |S|/3$ where $|S| = N_{i-1}$. Therefore, we have:

$$b^a_i + g^d_i \leq n^a_i + n^d_i \leq N_{i-1}/3$$

We are interested in the maximum value of the ratio of bad IDs to the total IDs at any point during the iteration. Thus, we pessimistically assume all additions of bad IDs and removals of good IDs come first in the iteration. We are then interested in the maximum value of the ratio:

$$\frac{B_{i-1} + b^a_i}{N_{i-1} + b^a_i - g^d_i} \leq \frac{N_{i-1}/3 + b^a_i}{N_{i-1} + b^a_i - g^d_i}$$

18
where the above inequality follows from Lemma 13 subject to the constraint that $b_i^a + g_i^d \leq N_{i-1}/3$. This ratio is maximized when the constraint achieves equality, that is when $g_i^d = N_{i-1}/3 - b_i^a$. Plugging this back into the ratio, we get:

\[
\frac{N_{i-1}/3 + b_i^a}{N_{i-1} + b_i^a - g_i^d} \leq \frac{N_{i-1}/3 + b_i^a}{2N_{i-1}/3 + 2b_i^a} = \frac{1}{2}
\]

Therefore, the maximum fraction of bad IDs is $1/2$ at any point during any arbitrary iteration $i > 0$.

Finally, we note that this argument is valid even though $\mathcal{S}_{cur}$ may not account for bad IDs that have departed without notifying the server. Intuitively, this is not a problem since such departures can only lower the fraction of bad IDs in the system; formally, the critical equation in the above argument is $b_i^a + g_i^d \leq N_{i-1}/3$, and this does not depend on $b_i^d$.

Over the remainder of this section we prove our resource costs.

Fix an iteration. For ease of analysis, we divide the iteration into sub-intervals of length $1/\tilde{J}$, indexed by $j \in \{1, ..., t\}$, where $\tilde{J}$ is the current estimate of join rate. Then:

**Lemma 15.** For any sub-interval $j \geq 1$ in an iteration, let $T_j$ be the total spending of the adversary in sub-interval $j$. Then, the number of bad IDs that join in this sub-interval, $b_j \leq \sqrt{2T_j}$.

**Proof.** Fix a sub-interval $j$ in some iteration. Then, assuming all bad IDs join before any good IDs in a sub-interval, and using the entrance difficulty setting from Step 1 of our algorithm, we get:

\[
T_j \geq \sum_{k=1}^{b_j} k \geq \frac{b_j^2}{2}
\]

On solving the above, we obtain the result.

**Lemma 16.** Fix an iteration. Suppose $\ell$ is the length of this iteration, and $J^G$ is the join rate of good IDs in this iteration. Then, the number of sub-intervals is at most $50\alpha^3\beta^6J^G\ell$. 

19
Proof. Suppose the iteration overlaps with $k$ distinct intervals. Let $\ell_j$ denote the length of the $j^{th}$ overlapping interval, and $\rho_j$ be the join rate of good IDs over the $j^{th}$ overlapping interval.

Next, recall from Lemma[5] that an interval intersects at most two epochs. So, let $\hat{\rho}_j$ be the join rate of good IDs over the two epochs that intersect with interval $j$, and let $\hat{\ell}_j$ be the lengths of their intersection. Then, from the $\beta-$smoothness property, we have:

$$J^G\ell \geq \sum_{j=1}^{k} \left( \hat{\rho}_j \hat{\ell}_j \right) / \beta$$  \hfill (8)

Finally, from the definition of a sub-interval, the number of sub-intervals in iteration $i$ is:

$$\sum_{j=1}^{k} \ell_j J^G \leq 50\alpha^3\beta^4 \sum_{j=1}^{k} \ell_j \rho_j \leq 50\alpha^3\beta^5 \sum_{j=1}^{k} \beta \left( \hat{\rho}_j \hat{\ell}_j \right) \leq 50\alpha^3\beta^5 (J^G \ell)$$

The first inequality follows from Theorem[1], the second step follows from $\beta-$smoothness property, and the third step from inequality[8].

**Lemma 17.** Fix a sub-interval in an iteration. Then, the number of good IDs that join over this sub-interval is at most $260\alpha^4\beta^5$.

**Proof.** Suppose $J^G$ is the estimate of good join rate in the current interval and $\rho$ is the join rate of good IDs over this interval. Assume that the sub-interval lies in the epoch with $\hat{\rho}$ join rate. Then, from the definition of a sub-interval and using the $\beta-$smoothness property, we bound the number of good IDs that join over the sub-interval as:

$$\beta \hat{\rho} \left( \frac{1}{J^G} \right) \leq \alpha \beta^2 \rho \left( \frac{1}{J^G} \right) \leq \alpha \beta^2 \left( \frac{260\alpha^3\beta^5}{\rho} \right) \rho \leq 260\alpha^4\beta^4$$

where the first step follows from Lemma[12] and the second step from from Theorem[1].

In the remainder of the proofs, we make use of the following algebraic fact that follows from the Cauchy-Schwartz inequality:

**Lemma 18.** Suppose that $u$ and $v$ are $x$-dimensional vectors in Euclidean space. For all $x \geq 1$:

$$\sum_{j=1}^{x} \sqrt{u_j v_j} \leq \sqrt{\sum_{j=1}^{x} u_j} \sqrt{\sum_{j=1}^{x} v_j}$$

20
Proof. Using the Cauchy-Schwarz inequality, we have:

\[
\left( \sum_{j=1}^{n} \sqrt{u_j v_j} \right)^2 \leq \sum_{j=1}^{n} u_j \sum_{j=1}^{n} v_j
\]

Taking the square-root of both sides, we obtain the desired result. \(\square\)

Lemma 19. Fix an iteration. Suppose \(\ell\) is the length of this iteration, \(J^G\) is the join rate of good IDs in this iteration, and \(T\) is the total resource cost to the adversary. Then, the total entrance cost to good IDs:

\[
f_1(\alpha, \beta) \left( \sqrt{f_2(\alpha, \beta) J^G \ell T} + f_1(\alpha, \beta) f_2(\alpha, \beta) J^G \ell \right)
\]

where \(f_1(\alpha, \beta) = 260\alpha^4\beta^4\) and \(f_2(\alpha, \beta) = 50\alpha^3\beta^6\).

Proof. Fix a sub-interval \(j\) in some iteration \(i\). Let \(g_j\) be the number of good IDs that join in sub-interval \(j\). Then, assuming all good IDs enter at the end of a sub-interval, we obtain the total resource cost to good IDs in sub-interval \(j\) to be at most:

\[
\sum_{k=1}^{g_j} (b_j + k) \\
\leq g_j \left( \sqrt{2T_j} + g_j \right) \\
\leq 260\alpha^4\beta^4 \left( \sqrt{2T_j} + 260\alpha^4\beta^4 \right)
\]

The second step follows from Lemma 15 and the last inequality follows from Lemma 17.

Suppose the iteration consists of \(t\) sub-intervals. Then, the total entrance spending of the good IDs in iteration \(i\) is:

\[
\sum_{j=1}^{t} \left( 260\alpha^4\beta^4 \left( \sqrt{2T_j} + 260\alpha^4\beta^4 \right) \right) \\
\leq 260\alpha^4\beta^4 \left( \sqrt{2t} \sum_{j=1}^{t} T_j + 260\alpha^4\beta^4 \ell \right) \\
\leq 260\alpha^4\beta^4 \left( \sqrt{50\alpha^3\beta^6 J^G \ell T} + 13000\alpha^7\beta^{10} J^G \ell \right)
\]

The second inequality follows from Lemma 18 and the last step follows from Lemma 16 and by substituting \(\sum_{j=1}^{t} T_j = T\). \(\square\)
Lemma 20. Fix an iteration. Suppose $\ell$ is the length of the iteration, $D$ is the rate of departure over this iteration, $J^G$ is the join rate of good IDs over this iteration, and $T$ is the total resource cost to the adversary in this iteration. Then, the total spending of good IDs in this iteration is:

$$A \leq 11D\ell + (f_1^2(\alpha, \beta)f_2(\alpha, \beta) + 11) J^G\ell + (f_1(\alpha, \beta) + 11) \sqrt{f_2(\alpha, \beta)J^G\ell T}$$

where $f_1(\alpha, \beta) = 260\alpha^4\beta^4$ and $f_2(\alpha, \beta) = 50\alpha^3\beta^6$.

Proof. Suppose $S$ is the set of IDs in the system at end of the iteration. Let $t$ denote the number of sub-intervals in this iteration, $g(b)$ be the number of good(bad) IDs that join in this iteration, $d$ be the number of departure in this iteration, and $T_j$ be resource cost to the adversary in sub-interval $j$. Since each good IDs solves a RB-challenge of difficulty 1 at the end of an iteration, hence the resource cost due to purge is at most:

$$|S| = 11(d + b + g)$$

$$\leq 11 \left(D\ell + \sum_{j=1}^{t} \sqrt{2T_j + J^G\ell} \right)$$

$$\leq 11 \left(D\ell + \sqrt{2lt \sum_{j=1}^{t} T_j + J^G\ell} \right)$$

$$\leq 11 \left(D\ell + \sqrt{f_2(\alpha, \beta)J^G\ell T + J^G\ell} \right)$$

where the second step follows from the purge condition of our algorithm, the third inequality follows from Lemma 15 and substituting $g = J^G\ell, b = D\ell$, the fourth inequality from Lemma 18, and the last inequality follows from Lemma 16 where $f_2(\alpha, \beta) = 400\alpha^3\beta^6$ and substituting $\sum_{j=1}^{t} T_j = T_t$.

Finally, using Lemma 19 and the purge cost calculated above, we compute the total spending of the good IDs in the iteration as:

$$A \leq f_1(\alpha, \beta) \left( \sqrt{f_2(\alpha, \beta)J^G\ell T} + f_1(\alpha, \beta)f_2(\alpha, \beta)J^G\ell \right)$$

$$+ 11 \left(D\ell + \sqrt{f_2(\alpha, \beta)J^G\ell T + J^G\ell} \right)$$

$$\leq 11D\ell + (f_1^2(\alpha, \beta)f_2(\alpha, \beta) + 11) J^G\ell + (f_1(\alpha, \beta) + 11) \sqrt{f_2(\alpha, \beta)J^G\ell T}$$

$\square$
Let \( I \) be any subset of iterations that for integers \( x \) and \( y \), contains every iteration with index between \( x \) and \( y \) inclusive. Let \( \delta(I) \) be \( |S_x - S_y| \); and let \( \Delta(I) \) be \( \delta(I) \) divided by the length of \( I \). Let \( \mathcal{A}_I \) and \( \mathcal{T}_I \) be the algorithmic and adversarial spend rates over \( I \); and let \( J^G_I \) be the good join rate over all of \( I \). Then we have the following lemma.

**Lemma 21.** For any subset of contiguous iterations, \( I \), which starts after iteration 1, the algorithmic spending rate over \( I \) is:

\[
O \left( \Delta(I) + f_1^2(\alpha, \beta) f_2(\alpha, \beta) J^G_I + f_1(\alpha, \beta) \sqrt{T_2(\alpha, \beta)} \sqrt{T_I J^G_I} \right)
\]

where \( f_1(\alpha, \beta) = 260 \alpha^4 \beta^4 \) and \( f_2(\alpha, \beta) = 50 \alpha^3 \beta^6 \).

**Proof.** Fix a subset of iteration \( I \). For all \( i \in I \), let \( \ell_i \) be the length of iteration \( i \), \( J^G_i \) be the join rate of good IDs, \( D_i \) be the departure rate of good IDs, and \( T_i \) be resource cost to the adversary in iteration \( i \). Then, from Lemma 20, we have the total spending of the good IDs over all iterations in \( I \), \( \mathcal{A}_I \) to be at most:

\[
\sum_{i \in I} \left( 11D_i \ell_i + (f_1^2(\alpha, \beta) f_2(\alpha, \beta) + 11) J^G_i \ell_i + (f_1(\alpha, \beta) + 11) \sqrt{f_2(\alpha, \beta)} J^G_i T_i \right)
\]

\[
\leq 11 \sum_{i \in I} D_i \ell_i + (f_1^2(\alpha, \beta) f_2(\alpha, \beta) + 11) \sum_{i \in I} J^G_i \ell_i
\]

\[
+ (f_1(\alpha, \beta) + 11) \sqrt{f_2(\alpha, \beta)} \frac{1}{\sum_{i \in I} \ell_i} \sum_{i \in I} J^G_i \ell_i T_i
\]

where the above inequality follows from Lemma 18. Dividing both sides of the above inequality by \( \sum_{i \in I} \ell_i \), we get:

\[
\mathcal{A}_I \leq 11 \frac{\sum_{i \in I} D_i \ell_i}{\sum_{i \in I} \ell_i} + (f_1^2(\alpha, \beta) f_2(\alpha, \beta) + 11) \frac{\sum_{i \in I} J^G_i \ell_i}{\sum_{i \in I} \ell_i}
\]

\[
+ \sqrt{f_2(\alpha, \beta)} (f_1(\alpha, \beta) + 11) \sqrt{\frac{\sum_{i \in I} T_i}{\sum_{i \in I} \ell_i} \frac{\sum_{i \in I} J^G_i \ell_i}{\sum_{i \in I} \ell_i}}
\]

\[
= O \left( \Delta(I) + f_1^2(\alpha, \beta) f_2(\alpha, \beta) J^G_I + f_1(\alpha, \beta) \sqrt{f_2(\alpha, \beta)} \sqrt{T_I J^G_I} \right)
\]

\[\square\]

**Proof of Theorem 2.** The resource cost follows from Lemma 21 by noting that \( \Delta(I) = 0 \) when \( I \) is all iterations, since the system is initially empty. Then, combining this result with Lemma 14 completes our proof.  \[\square\]
We state an interesting corollary implied by Theorem 2.

**Corollary 22.** For \( \alpha \leq 1/18 \), with error probability polynomially small in \( n_0 \) over the system lifetime, and for any subset of contiguous iterations \( \mathcal{I} \), \( \text{ERGO} \) has an algorithmic spending rate of
\[
O\left( \Delta(\mathcal{I}) + J_{\mathcal{I}}^G + \sqrt{T_{\mathcal{I}} J_{\mathcal{I}}^G} \right).
\]

This shows that the spending rate for the algorithm remains small, even when focusing on just a subset of iterations. To understand why this is important, consider a long-lived system which suffers a single, significant attack for a small number of iterations, after which there are no more attacks. The cost of any defense may be small when amortized over the lifetime of the system, but this does not give a useful guarantee on performance during the time of attack.

### 6 Experiments

We now report on several empirical results. First, in Section 6.1, we evaluate the performance of the GOODJEST algorithm by measuring the approximation factor for the join rate of good IDs. Then, in Section 6.2, we measure the resource burning cost, for ERGO, as a function of the adversarial cost, and compare it against prior resource burning based algorithms. Finally, in Section 6.3, we propose and implement several heuristics to improve the performance of ERGO. All our experiments were written in MATLAB.

**Data Sets.** Our experiments use data from the following networks:

1. **Bitcoin:** This dataset records the join and departure events of IDs in the Bitcoin network, timestamped to the second, over roughly 7 days [55].

2. **BitTorrent RedHat:** This dataset simulates the join and departure events for the BitTorrent network to obtain a RedHat ISO image. We use the Weibull distribution with shape and scale parameters of 0.59 and 41.0, respectively, from [64].

3. **Ethereum:** This dataset simulates join and departure events of IDs for the Ethereum network. Based on a study in [37], we use the Weibull distribution with shape parameter of 0.52 and scale parameter of 9.8.

4. **Gnutella:** This dataset simulates join and departure events for the Gnutella network. Based on a study in [59], we use an exponential distribution with mean of 2.3 hours for session time, and Poisson distribution with mean of 1 ID per second for the arrival rate.
6.1 Evaluating the performance of ESTIMATE

For the Bitcoin network, the system initially consists of 9212 IDs, and the join and departure events are based off the dataset from [56]. For the remaining networks, we initialize them with 10000 IDs each, and simulate the join and departure events over 100,000 timesteps.

Along with the churn based on the data sets for the various systems, we add bad IDs at a constant join rate to simulate two specific scenarios, where we vary the resource budget of the adversary per unit time, $T = 0$ and 100, and vary the fraction of bad IDs in the system at any time to be $\in \{1/1500, 1/375, 1/94, 1/24, 1/6\}$. Then, for each combination of $T$ and the fraction of bad IDs, we measure the $\tilde{J}_i^{\rho_i}$ for all epoch $i \geq 1$ over the entire simulation. Finally, we plot error bars for each of the six systems.

We report our results in Figure 4. These plots demonstrate the robustness

![Figure 4: Ratio of $\tilde{J}_i^{\rho_i}$ to $\hat{\rho}_{i-1}$ versus fraction of bad IDs in the system for GOODJEST.](image)

25
of our estimate to adversarial budget. In the absence of an attack i.e., \( T = 0 \), our estimate is within a constant factor in the range \((0.08, 1.2)\). This variation can be attributed the \( \alpha, \beta \)—smoothness of churn. Moreover, even when the adversarial budget per unit time is quite high, our estimate is off by a constant factor in the range \((0.08, 4)\) over all the systems at any time.

### 6.2 Evaluating Computational Cost without Heuristics

We now measure the spend rate for ERGO, focusing solely on the cost of solving RB-challenges. Throughout, we assume a cost of \( k \) for solving a RB-challenge of difficulty \( k \). We compare the performance of ERGO against four resource burning based Sybil defense algorithms: **GMCom** [31], **CCom** [30], **SybilControl** [38], and **REMP** (a name that uses the authors’ initials) [58], summarized below.

**GMCom.** GMCom is like ERGO, except for two differences. First, the entrance cost is the maximum of 1, and the measured join rate in the current iteration divided by an estimate of the good join rate. Second, the estimate of the good join rate is computed via a different incorrect heuristic [31].

**CCom.** CCom is the same as ERGO except the difficulty of RB-challenge assigned to joining IDs is always 1.

**SybilControl.** Each ID solves a RB-challenge to join. Additionally, each ID tests its neighbors with a RB-challenge every 5 seconds, removing from its list of neighbors those IDs that fail to provide a solution within a fixed time period. These tests are not coordinated between IDs.

**REMP.** Each ID solves a RB-challenge to join. Additionally, each ID must solve RB-challenges every \( W \) seconds. We use Equation (4) from [58] to compute the value of spending rate per ID as:

\[
\frac{L}{W} = \frac{n}{N_{\text{attacker}}} = \frac{T_{\text{max}}}{\kappa N} \tag{9}
\]

where \( L \) is the cost to an ID per \( W \) seconds, \( n \) is the number of IDs that the adversary can add to the system and \( N_{\text{attacker}} \) is the total number of attackers in the system. Suppose \( N \) is the system size, then \( N_{\text{attacker}} \) is \( \kappa N \) since the adversary has an \( \kappa \) fraction of resources of the network in our model. Suppose \( T_{\text{max}} \) is the maximum number of attackers (bad IDs) that can join in \( W \) seconds, then to guarantee that the fraction of bad IDs is less than half, \( n = T_{\text{max}} \). Substituting these values in Equation (9) we can compute total algorithmic spending rate as:

\[
A_{\text{REMP}} = (1 - \kappa)N \times \frac{L}{W} = \frac{(1 - \kappa)T_{\text{max}}}{\kappa} \tag{10}
\]
Setup. We set $\kappa = 1/18$, and let $T$ range over $[2^0, 2^{20}]$, where for each value of $T$, the system is simulated for 10,000 seconds. We also simulate the case $T = 0$. We assume that the adversary only solves RB-challenges to add IDs to the system. For REMP, we consider two values of $T_{\text{max}}$, $10^4$ and $10^7$. Setting $T_{\text{max}} = 10^7$ ensures correctness for all values of $T$ considered, and $T_{\text{max}} = 10^4$ ensures correctness for $T \leq 10^4$.

Results. Figure 5 illustrates our results; we omit error bars since they are negligible. The x-axis is the adversarial spending rate, $T$; and the y-axis is the algorithmic spending rate, $A$. We cut off the plots for REMP-$10^4$ and SYBILCONTROL, when they can no longer ensure that the fraction of bad IDs is less than $1/2$. We also note that REMP-$10^7$ only ensures a minority of bad IDs for up to $T = 10^7$.

ERGO always has spend rate as low as the other algorithms for $T \geq 100$, and significantly less than the other algorithms for large $T$, with improvements that grow to about 2 orders of magnitude. In Section 6.3, our heuristics close this gap, allowing ERGO to outperform all algorithms for all $T \geq 0$. The spend rate for ERGO is linear in $\sqrt{T}$, agreeing with our analytical results. We emphasize that the benefits of ERGO are consistent over four disparate networks. These results illustrate the value of GOODJEST.

Finally, we note that ERGO guarantees a fraction of bad IDs no more than $1/6$ for all values of $T$. In contrast, SYBILCONTROL and REMP guarantee a fraction of bad IDs less than $1/2$ for the values of $T$ plotted.

For $T \geq 100$, GMCOM and CCOM perform almost identically. This occurs because of an error in the estimation heuristic of [31], which causes GMCOM to incorrectly estimate $J^G$, when $T$ is much larger than $J^G$ ($J^G \approx 10$ in these plots). When the estimate is incorrect, by the specification in [31], the entrance-puzzle hardness is set to 1, and so GMCOM reverts to CCOM. Our simulations in [31] assumed knowledge of the good join rate, and thus did not reveal the flaw in GMCOM’s estimation method.

6.3 Heuristics

Next, we present heuristics to improve the performance of ERGO. To determine effective heuristics, we focus on two separate costs to good IDs: cost to join the system and purge cost. In studying these costs for the Bitcoin network in the absence of an attack, we find that the purge cost dominates and, therefore, we focus on reducing the purge frequency.

Heuristic 1: We align the computation of $J^G$ with the end of the iteration. Specifically, at the end of an interval, we wait for the end of the current
Figure 5: Algorithmic cost versus adversarial cost for ERGO, GMCOM, CCom, SybilControl and REMP.

...iteration to end and then obtain the estimate. This reduces the fraction of bad IDs in $|S_{\text{cur}} \oplus S_{i-1}|$ to at most $\kappa$-fraction.

**Heuristic 2:** We use the symmetric difference to determine when to do a purge. Specifically, for iteration $i$, if $|S_{\text{cur}} \oplus S_{i-1}| \geq |S_{i-1}|/11$, then a purge is executed. This ensures that the fraction of bad IDs can increase by no more than in our original specification. Also, it decreases the purge frequency: for example, in the case when some ID joins and departs repeatedly.

**Heuristic 3:** We use the estimated good-ID join rate, obtained via GOOD-JEst, to bound the maximum number of bad IDs that can have joined during an iteration. This allows us to upper-bound the fraction of bad IDs in the system and to purge only when the population invariant is at risk.

**Heuristic 4:** Recent works have explored the possibility of identifying bad IDs based on the network topology [22, 47]. In our experiments, we focus on SybilFuse [22], which has the probability of correctly classifying an ID as either good or bad as 0.92 and 0.98 based on the empirical results from [22],
Section IV-B, last paragraph. We assume these values hold and use SybilFuse to diagnose whether a joining ID is good or bad; in the latter case, the ID is refused entry.

We evaluate the performance of these heuristics against ERGO. The experimental setup is the same as Section 6.2. We define ERGO-H1 using both Heuristic 1 and Heuristic 2, and ERGO-H2 using Heuristic 1, Heuristic 2 and Heuristic 3. We define ERGO-SF(92) and ERGO-SF(98) to be ERGO using Heuristics 1, Heuristic 2, Heuristic 3 and also Heuristic 4, with the accuracy parameter of Heuristic 4 as 0.98 and 0.92, respectively.

We simulated Heuristic 1 in conjunction with ERGO but it did not result in improvements. So, we omit it from our final plots. But this does advocate the independence of GOODJEST from the underlying Sybil defense mechanism, which indicates its wider applicability.

Figure 6 illustrates our results. Note that ERGO-SF(92) and ERGO-SF(98) reduce costs significantly during adversarial attack, with improve-
ments of up to three orders of magnitude during the most significant attack tested. Again, these improvements are consistent across 4 different types of data sets.

7 Decentralizing our Algorithm via a Committee

We adapt our algorithms for systems where there is no server. To this end, we establish and maintain a subset of IDs called a committee, which always consists of an honest majority. The committee takes over the responsibilities of the server. This gives rise to a number of challenges:

1. There is no longer a central authority to generate and issue RB-challenges.

2. In the absence of a server, the committee members need to agree on the system state, sequence of joins and departures as well as the value of $\tilde{J}^{G}$.

3. Finally, the committee members need to run GOODJEST in a distributed manner.

We address all the above challenges below, but before we first discuss some modifications to our model to account for the decentralized nature of our system.

7.1 Modifications to the model

All communication among good IDs uses a broadcast primitive, DIFFUSE, which allows a good ID to send a value to all other good IDs within a known and bounded amount of time, despite an adversary. Such a primitive is a standard assumption in PoW schemes [14, 23, 25, 40]; see [43] for empirical justification. The committee uses diffuse to create and issue RB-challenges, and remainder of IDs in the network use diffuse to communicate solutions to the rest of the network. All IDs are assumed to be synchronized, and have clocks that are synchronized.

In any round, up to a constant fraction of the good IDs can join or depart, and the timing of these events can be arbitrary. Therefore, the number of good IDs may greatly increase or decrease over consecutive rounds, modeling a system that is subject to significant churn.

\footnote{Communication latency in Bitcoin is 12 seconds; puzzle solving time is 10 minutes [16]}
7.2 Committee and System Initialization

To initialize our system, we make use of an algorithm created by Andrychowicz and Dziembowski [2], which we call GENID. This algorithm will be used at the time of system initialization in ERGO to create an initial set of IDs, which contains no more than an $\kappa$-fraction of bad IDs. Additionally, the GENID algorithm also selects a committee of logarithmic size that has a majority of good IDs. A number of other existing works provide similar solutions such as [2,33,36], but are restricted to computational resource burning. So, the design of a generic resource burning based GENID algorithm remains an open problem.

7.3 Maintaining the Committee

Our committee uses State Machine Replication by Abraham et al. [1] to agree on the system state as well as the events occurring in our system to facilitate running GOODJEST and ERGO in parallel. This protocol enables the committee members to commit a log of the requests made by the IDs, such that the following properties hold: 1) honest committee members do not commit to different values at the same log position, and 2) each ID request is eventually committed by the honest committee members.

In order to correctly run state machine replication, we require our committee to always consist of a majority of good IDs. Thus, we define the following goal for our system:

**Committee Invariant:** There always exists a committee known to all good IDs of size $\Theta(\log n_0)$, which always consists of a majority of good IDs.

To maintain this invariant, we modify ERGO to elects a committee of size $C \log |S_i|$ at the end of iteration $i$. The committee generates a random string, $r$ of length $C \log^2 |S_i|$ bits using Awerbuch and Scheideler’s random number generator protocol from [11], where $C > 1$ is some constant. The $i^{th}$ member of the committee is then the ID selected using the $i^{th}$ substring of $r$.

With the modifications outlined above to accommodate for a committee, ERGO gives the following guarantees:

**Theorem 23.** For $\kappa \leq 1/18$, ERGO ensures that the good spend rate is $O \left( \alpha^6 \beta^4 \left( \sqrt{T(J^G + 1)} + J^G \right) \right)$ and maintains the invariants:

- **Population Invariant:** The fraction of bad IDs in the system is always less than $1/6$. 

31
• **Committee Invariant:** There is always a committee, of size logarithmic in the current system size, known to all good IDs, that contains less than a $1/2$-fraction of bad IDs.

### 7.4 Proof of Theorem 23

In the next lemma, we prove the committee invariant.

**Lemma 24.** For $C \geq 90(\gamma + 1)$ and $0 < \epsilon < 1$, ERGO maintains a committee of size at least $\frac{C}{2} \log n_t$, which consists of an honest majority with probability at least $1 - O\left(n_0^{-(\gamma+1)}\right)$.

**Proof.** For epoch $i = 0$, committee goal holds true from the use of GenID to initialize the system (recall from Section 5.1); for details, see Lemma 6 of [3].

Fix an epoch $i > 0$. Recall that a new committee is elected by the existing committee at the end of the current epoch by selecting $c \log |S_i|$ IDs independently and uniformly at random from the set $S_i$, and for a sufficiently large constant $c > 0$ which we define concretely later on in this argument. Then, let $X_G$ be a random variable which denotes the number of good IDs elected to the new committee in epoch $i$. Then:

$$E[X_G] = \frac{|G_i|}{|S_i|} c \log |S_i| = (1 - \alpha)c \log |S_i|$$

where the last inequality follows from the fact that the computational power with the adversary is at most $\alpha$. Next, we bound the number of good IDs in the committee using Chernoff Bound [48] as:

$$Pr (X_G < (1 - \delta)(1 - \alpha)c \log |S_i|) \leq \exp \left\{ -\frac{\delta^2(1 - \alpha)c \log |S_i|}{2} \right\} = \frac{1}{|S_i|\left[(1 - \alpha)c\delta^2/2\right]} = O\left(n_0^{-(\gamma+1)}\right)$$

where the first step holds for any constant $0 < \delta < 1$, the second step follows from Equation 11, and finally, the last step holds for all $c \geq \frac{28}{13} \frac{(\gamma + 1)}{\delta^2}$. For $\delta = 1/100$, we obtain can bound the number of good IDs in the committee to be at least $9/10c \log |S_i|$ with probability at least $1 - O(n_0^{-(\gamma+1)})$. 

32
Let $Y_g$ be a random variable which denotes the number of good IDs that depart from the committee when the number of departures of good IDs from the system is at most $|S_{i-1}|/3$. Then, since the departures of good IDs occurs independently and uniformly at random from the system (See Section 2), we obtain:

$$E[Y_g] \leq \frac{|S_i|}{3} \left( \frac{c \log |S_i|}{|S_i|} \right) = \frac{c}{3} \log |S_i|$$ (12)

Next, we obtain upper bound on the number of departures of good IDs from the committee using Chernoff Bound as:

$$\Pr\left( Y_g > (1 + \delta') \frac{c \log |S_i|}{3} \right) \leq \exp \left\{ -\frac{\delta'^2 c}{6} \log |S_i| \right\}$$

$$= \frac{1}{|S_i|^{\delta'^2 c/6}}$$

$$= O\left( n_0^{-(\gamma+1)} \right)$$

where the first step holds for any constant $0 < \delta' < 1$ and the last step holds for all $c \geq \frac{6(\gamma+1)}{\delta'^2}$.

Thus, for $\delta' = 1/5$, with probability $1 - O(n_0^{-(\gamma+1)})$, the minimum number of good IDs in the committee is greater than $(9/10)c \log |S_i| - (4/10)c \log |S_i|$. Next, use a union bound over $n_0$ epochs to show that the committee goal invariant is maintained throughout the lifetime of the system.

The proof of Theorem 23 follows from Lemma 24 and Lemma 21.

8 Conclusion

We have designed and analyzed two algorithms. The first is GOODJEST, which estimates the good join rate in general systems despite adversarially-scheduled injections/departures of bad IDs, so long as the fraction of bad IDs in the system is sufficiently limited. The second is ERGO, which leverages resource-burning to gain an advantage over a Sybil adversary under churn that cannot be tolerated by previous approaches. Our proposed heuristics yield improved performance, with experiments illustrating that ERGO decreases computational cost to the good IDs compared to other PoW-based Sybil defenses.
References

[1] Ittai Abraham, Dahlia Malkhi, Kartik Nayak, Ling Ren, and Maofan Yin. Sync hotstuff: simple and practical synchronous state machine replication. *IACR Cryptology ePrint Archive*, 2019:270, 2019.

[2] Marcin Andrychowicz and Stefan Dziembowski. Pow-based distributed cryptography with no trusted setup. In *Proceedings of the Annual Cryptology Conference*, pages 379–399. Springer, 2015.

[3] Marcin Andrychowicz, Stefan Dziembowski, Daniel Malinowski, and Łukasz Mazurek. Modeling bitcoin contracts by timed automata. In *International Conference on Formal Modeling and Analysis of Timed Systems*, pages 7–22. Springer, 2014.

[4] John Augustine, Anisur Rahaman Molla, Ehab Morsy, Gopal Pandurangan, Peter Robinson, and Eli Upfal. Storage and Search in Dynamic Peer-to-peer Networks. In *Proceedings of the Twenty-fifth Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 53–62, 2013.

[5] John Augustine, Gopal Pandurangan, and Peter Robinson. Fast Byzantine Agreement in Dynamic Networks. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 74–83, 2013.

[6] John Augustine, Gopal Pandurangan, and Peter Robinson. Fast Byzantine Leader Election in Dynamic Networks. In *International Symposium on Distributed Computing (DISC)*, pages 276–291. Springer, 2015.

[7] John Augustine, Gopal Pandurangan, Peter Robinson, Scott Roche, and Eli Upfal. Enabling Robust and Efficient Distributed Computation in Dynamic Peer-to-Peer Networks. In *Proceedings of the IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 350–369, 2015.

[8] John Augustine, Gopal Pandurangan, Peter Robinson, and Eli Upfal. Towards Robust and Efficient Computation in Dynamic Peer-to-peer Networks. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 551–569, 2012.

[9] Baruch Awerbuch and Christian Scheideler. Chord++: Low-Congestion Routing in Chord. [www.cs.jhu.edu/~scheideler/papers/chord4.ps.gz](http://www.cs.jhu.edu/~scheideler/papers/chord4.ps.gz), 2004.
[10] Baruch Awerbuch and Christian Scheideler. Group Spreading: A Protocol for Provably Secure Distributed Name Service. In 31st International Colloquium on Automata, Languages, and Programming (ICALP), pages 183–195, 2004.

[11] Baruch Awerbuch and Christian Scheideler. Robust Random Number Generation for Peer-to-peer Systems. In 10th International Conference On Principles of Distributed Systems (OPODIS), pages 275–289, 2006.

[12] Baruch Awerbuch and Christian Scheideler. Towards a Scalable and Robust DHT. In Proceedings of the 18th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pages 318–327, 2006.

[13] Baruch Awerbuch and Christian Scheideler. Towards Scalable and Robust Overlay Networks. In Proceedings of the 6th International Workshop on Peer-to-Peer Systems (IPTPS), page n. pag., 2007.

[14] BitcoinWiki. BitcoinWiki Network. https://en.bitcoin.it/wiki/Network#Standard_relaying.

[15] CoinDesk. Think Tank Reignites Debate Over Bitcoin Mining’s Environmental Effects. http://www.coindesk.com/think-tank-debate-bitcoin-mining-environment/, 2015.

[16] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gün Sirer, et al. On scaling decentralized blockchains. In International Conference on Financial Cryptography and Data Security, pages 106–125. Springer, 2016.

[17] George Danezis, Chris Lesniewski-laas, M. Frans Kaashoek, and Ross Anderson. Sybil-Resistant DHT Routing. In Proceedings of the 10th European Symposium On Research In Computer Security (ESORICS), pages 305–318, 2005.

[18] John Douceur. The Sybil Attack. In Proceedings of the Second International Peer-to-Peer Symposium (IPTPS), pages 251–260, 2002.

[19] Cynthia Dwork and Moni Naor. Pricing via Processing or Combatting Junk Mail. In Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology, pages 139–147, 1993.
[20] Jarret Falkner, Michael Piatek, John P. John, Arvind Krishnamurthy, and Thomas Anderson. Profiling a Million User DHT. In 7th ACM SIGCOMM Conference on Internet Measurement, pages 129–134, 2007.

[21] Amos Fiat, Jared Saia, and Maxwell Young. Making Chord Robust to Byzantine Attacks. In Proceedings of the 13th European Symposium on Algorithms (ESA), pages 803–814, 2005.

[22] Peng Gao, Binghui Wang, Neil Zhenqiang Gong, Sanjeev R Kulkarni, Kurt Thomas, and Prateek Mittal. Sybilfuse: Combining local attributes with global structure to perform robust sybil detection. In 2018 IEEE Conference on Communications and Network Security (CNS), pages 1–9, 2018. https://www.princeton.edu/~pmittal/publications/sybilfuse-cns18.pdf.

[23] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. The Bitcoin Backbone Protocol: Analysis and Applications. In Proceedings of 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), pages 281–310, 2015.

[24] Stephanie Gil, Swarun Kumar, Mark Mazumder, Dina Katabi, and Daniela Rus. Guaranteeing Spoof-Resilient Multi-Robot Networks. In Proceedings of Robotics: Science and Systems, Rome, Italy, July 2015.

[25] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nikolai Zeldovich. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In Proceedings of the 26th Symposium on Operating Systems Principles (SOSP), pages 51–68, 2017.

[26] Seth Gilbert, Calvin Newport, and Chaodong Zheng. Who Are You? Secure Identities in Ad Hoc Networks. In Proceedings of the 28th International Symposium on Distributed Computing (DISC), pages 227–242, 2014.

[27] Seth Gilbert and Chaodong Zheng. SybilCast: Broadcast on the Open Airwaves. In Proceedings of the 25th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pages 130–139, 2013.

[28] Rachid Guerraoui, Florian Huc, and Anne-Marie Kermarrec. Highly Dynamic Distributed Computing with Byzantine Failures. In Proceedings of the 2013 ACM Symposium on Principles of Distributed Computing (PODC), pages 176–183, 2013.
[29] Diksha Gupta, Jared Saia, and Maxwell Young. Proof of Work Without All the Work. *arXiv preprint arXiv:1708.01285*, 2017.

[30] Diksha Gupta, Jared Saia, and Maxwell Young. Proof of Work Without All the Work. In *Proceedings of the 19th International Conference on Distributed Computing and Networking (ICDCN)*, 2018.

[31] Diksha Gupta, Jared Saia, and Maxwell Young. Peace through superior puzzling: An asymmetric sybil defense. In *33rd IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2019.

[32] Diksha Gupta, Jared Saia, and Maxwell Young. Invited paper: Resource burning for permissionless systems. In *Accepted to the 27th International Conference on Structural Information and Communication Complexity (SIROCCO)*, 2020. URL: [https://arxiv.org/abs/2006.04865](https://arxiv.org/abs/2006.04865).

[33] Ruomu Hou, Irvan Jahja, Loi Luu, Prateek Saxena, and Haifeng Yu. Randomized view reconciliation in permissionless distributed systems. *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 2528–2536, 2018.

[34] Mercy O. Jaiyeola, Kyle Patron, Jared Saia, Maxwell Young, and Qian M. Zhou. Tiny Groups Tackle Byzantine Adversaries. *CoRR*, abs/1705.10387, 2017. URL: [http://arxiv.org/abs/1705.10387](http://arxiv.org/abs/1705.10387).

[35] R. John, J. P. Cherian, and J. J. Kizhakkethottam. A Survey of Techniques to Prevent Sybil Attacks. In *Proc. of the Intl. Conference on Soft-Computing and Networks Security (ICSNS)*, pages 1–6, 2015.

[36] Jonathan Katz, Andrew Miller, and Elaine Shi. Pseudonymous secure computation from time-lock puzzles. *Citeseer*, 2014.

[37] Seoung Kyun Kim, Zane Ma, Siddharth Murali, Joshua Mason, Andrew Miller, and Michael Bailey. Measuring ethereum network peers. In *Proceedings of the Internet Measurement Conference 2018*, pages 91–104, 2018.

[38] Frank Li, Prateek Mittal, Matthew Caesar, and Nikita Borisov. Sybil-Control: Practical Sybil Defense with Computational Puzzles. In *Proceedings of the Seventh ACM Workshop on Scalable Trusted Computing*, pages 67–78, 2012.
[39] Y. Liu, D. R. Bild, R. P. Dick, Z. M. Mao, and D. S. Wallach. The Mason Test: A Defense Against Sybil Attacks in Wireless Networks Without Trusted Authorities. *IEEE Transactions on Mobile Computing*, 14(11):2376–2391, 2015.

[40] Loi Luu, Viswesh Narayanan, Chaodong Zheng, Kunal Baweja, Seth Gilbert, and Prateek Saxena. A Secure Sharding Protocol For Open Blockchains. In *Proc. of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 17–30, 2016.

[41] Ars Technica. Mining bitcoins takes power, but is it an Environmental Disaster? [http://arstechnica.com/business/2013/04/mining-bitcoins-takes-power-but-is-it-an-environmental-disaster](http://arstechnica.com/business/2013/04/mining-bitcoins-takes-power-but-is-it-an-environmental-disaster), 2013.

[42] The Economist. Why bitcoin uses so much energy. [https://www.economist.com/the-economist-explains/2018/07/09/why-bitcoin-uses-so-much-energy](https://www.economist.com/the-economist-explains/2018/07/09/why-bitcoin-uses-so-much-energy), 2018.

[43] Andrew Miller, James Litton, Andrew Pachulski, Neil Spring, Neal Gupta, Dave Levin, and Bobby Bhattacharjee. Discovering Bitcoin’s Public Topology and Influential Nodes, 2015. [http://cs.umd.edu/projects/coinscope/coinscope.pdf](http://cs.umd.edu/projects/coinscope/coinscope.pdf).

[44] D. L. Mills. Internet time synchronization: the network time protocol. *IEEE Transactions on Communications*, 39(10):1482–1493, 1991.

[45] D. L. Mills. Improved algorithms for synchronizing computer network clocks. *IEEE/ACM Transactions on Networking*, 3(3):245–254, 1995.

[46] David L. Mills. On the accuracy and stability of clocks synchronized by the network time protocol in the internet system. *SIGCOMM Comput. Commun. Rev.*, 20(1):65–75, December 1989. [doi:10.1145/86587.86591](https://doi.org/10.1145/86587.86591).

[47] Satyajayant Misra, Abu Saleh Md Tayeen, and Wen Xu. Sybilexposer: An effective scheme to detect sybil communities in online social networks. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2016.

[48] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge University Press, 2017.
[49] Aziz Mohaisen and Scott Hollenbeck. Improving Social Network-Based Sybil Defenses by Rewiring and Augmenting Social Graphs. In *Proceedings of the 14th International Workshop on Information Security Applications (WISA)*, pages 65–80, 2014.

[50] Aziz Mohaisen and Joongheon Kim. The Sybil Attacks and Defenses: A Survey. *Smart Computing Review*, 3(6):480–489, 2013.

[51] Diogo Mónica, Loao Leitao, Luís Rodrigues, and Carlos Ribeiro. On the Use of Radio Resource Tests in Wireless Ad-Hoc Networks. In *Proceedings of the 3rd Workshop on Recent Advances on Intrusion-Tolerant Systems*, pages F21–F26, 2009.

[52] Tal Moran and Ilan Orlov. Simple proofs of space-time and rational proofs of storage. In *Annual International Cryptology Conference*, pages 381–409. Springer, 2019.

[53] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System, 2008. [http://bitcoin.org/bitcoin.pdf](http://bitcoin.org/bitcoin.pdf).

[54] Till Neudecker. Bitcoin Cash (BCH) Sybil Nodes on the Bitcoin Peer-to-Peer Network, 2017. [http://dsn.tm.kit.edu/publications/files/332/bch_sybil.pdf](http://dsn.tm.kit.edu/publications/files/332/bch_sybil.pdf).

[55] Till Neudecker. Personal correspondence, 2019.

[56] Till Neudecker, Philipp Andelfinger, and Hannes Hartenstein. Timing analysis for inferring the topology of the bitcoin peer-to-peer network. In *Proceedings of the 13th IEEE International Conference on Advanced and Trusted Computing (ATC)*, July 2016.

[57] James Newsome, Elaine Shi, Dawn Song, and Adrian Perrig. The Sybil Attack in Sensor Networks: Analysis & Defenses. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 259–268, 2004.

[58] Hosam Rowaihy, William Enck, Patrick McDaniel, and Thomas La Porta. Limiting sybil attacks in structured peer-to-peer networks. In *Proceedings of IEEE Infocom Mini-Symposium. IEEE Press, Washington, DC, USA*, pages 1201–1206, 2005.

[59] Hosam Rowaihy, William Enck, Patrick McDaniel, and Tom La Porta. Limiting sybil attacks in structured p2p networks. In *IEEE Interna-
[60] Christian Scheideler. How to spread adversarial nodes? rotate! In Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing (STOC), 2005.

[61] Christian Scheideler and Stefan Schmid. A Distributed and Oblivious Heap. In Proceedings of the 36th International Colloquium on Automata, Languages and Programming: Part II, ICALP ’09, pages 571–582, 2009.

[62] Micah Sherr, Matt Blaze, and Boon Thau Loo. Veracity: Practical Secure Network Coordinates via Vote-based Agreements. In Proceedings of the USENIX Annual Technical Conference, pages 13–13, 2009.

[63] Ali Shoker. Sustainable blockchain through proof of exercise. In 2017 IEEE 16th International Symposium on Network Computing and Applications (NCA), pages 1–9. IEEE, 2017.

[64] Daniel Stutzbach and Reza Rejaie. Understanding Churn in Peer-to-Peer Networks. In Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement (IMC), pages 189–202, New York, NY, USA, 2006. ACM. URL: http://doi.acm.org/10.1145/1177080.1177105

[65] Michael Walfish, Mythili Vutukuru, Hari Balakrishnan, David Karger, and Scott Shenker. DDoS Defense by Offense. ACM Transactions on Computer Systems, 28(1):3, 2010.

[66] L. Wang and J. Kangasharju. Real-World Sybil Attacks in BitTorrent Mainline DHT. In IEEE Global Communications Conference (GLOBECOM), pages 826–832, 2012.

[67] L. Wang and J. Kangasharju. Measuring Large-Scale Distributed Systems: Case of BitTorrent Mainline DHT. In IEEE 13th International Conference on Peer-to-Peer Computing (P2P), pages 1–10, 2013.

[68] Wei Wei, Fengyuan Xu, Chiu C. Tan, and Qun Li. SybilDefender: A Defense Mechanism for Sybil Attacks in Large Social Networks. IEEE Transactions on Parallel & Distributed Systems, 24(12):2492–2502, 2013.
[69] Zhi Yang, Christo Wilson, Xiao Wang, Tingting Gao, Ben Y. Zhao, and Yafei Dai. Uncovering Social Network Sybils in the Wild. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference (IMC)*, pages 259–268, 2011.

[70] Maxwell Young, Aniket Kate, Ian Goldberg, and Martin Karsten. Towards Practical Communication in Byzantine-Resistant DHTs. *IEEE/ACM Transactions on Networking*, 21(1):190–203, February 2013.

[71] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. SybilGuard: Defending Against Sybil Attacks via Social Networks. *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, 36:267–278, August 2006.