

Routing Centralization and Censorship in IPFS

Jan Matter¹ and Muoi Tran²[0000–0002–6049–5688]

¹ ETH Zürich

² Chalmers University of Technology and University of Gothenburg
muoi@chalmers.se

Abstract. The InterPlanetary File System (IPFS) promises decentralized storage, in which no single party can render content unavailable. However, most content providers have converged onto a few large public-cloud operators [5, 38]. We show that this hosting concentration also collapses onto a few autonomous systems (ASes) and largely unprotected IP prefixes. IPFS is thus exposed to interdomain routing attacks, namely passive interception and BGP hijacking, that have been demonstrated against other decentralized systems such as Bitcoin and Tor. Measuring the live IPFS topology and simulating these attacks against 3,000 content identifiers (CIDs), we find that a single malicious AS can block more than 75% of content from over 57% of requesters, and that hijacking only about 100 IP prefixes already reaches at least 84% of this maximum, making partial censorship inexpensive. We further propose and evaluate countermeasures, i.e., network-wide collaborative replication, combined with a few backup providers hardened against BGP hijacking, that reduce the achievable censorship to about 20% of requesters.

1 Introduction

The InterPlanetary File System (IPFS) has become the de facto platform for decentralized storage [6, 46, 47], underpinning content-delivery, social, and data-persistence services (e.g., Audius, Matters News, Firebase) and gaining native support in browsers such as Brave and Opera [19]. Central to its design is content addressing, in which each object is identified by a self-certifying Content Identifier (CID) derived from a Merkle directed acyclic graph over its constituent chunks and located through a Kademlia distributed hash table (DHT) [27] in a peer-to-peer network of content providers. By substituting content addressing for name resolution, IPFS lets any content requester verify integrity independently of the providers, and by replicating the content across multiple providers, it ensures that no single party can render a given CID unavailable in principle.

In practice, IPFS’s operation shows well-documented tensions with its decentralization goals. Its own maintainers introduced centralized components (e.g., indexers, nodes, and HTTP gateways) that trade decentralization for lookup latency and scalability a purely peer-to-peer design cannot attain [47]. Independent studies document a complementary, less intentional concentration: a large fraction of IPFS nodes and providers are hosted within a few public-cloud operators [4, 46], raising data-management and governance concerns [38]. The system is thus more centralized at the *hosting* layer than its design suggests.

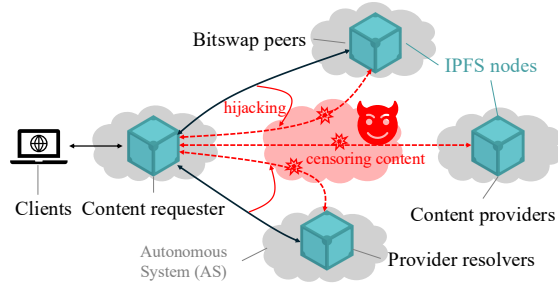


Fig. 1: High-level overview of routing-based censorship in IPFS. The malicious autonomous system (AS) intercepts content requester’s communication, and censors its exchange with other peers, content providers, or provider resolvers.

Little attention, however, has been paid to another centralization dimension, the *interdomain routing layer*, at which reachability between IPFS components is ultimately determined by a network of autonomous systems (ASes) running the Border Gateway Protocol (BGP) [33]. The centralization can be an AS that naturally sits on the routing paths between many IPFS components and intercepts the traffic exchanged among them. Crucially, this centralization is not limited to on-path ASes because a malicious AS can falsely claim ownership of IP prefixes and hijack traffic destined to them [12], drawing itself onto paths it would not otherwise be on. The bottleneck is therefore *broader* than the known hosting concentration, and it may reside in ASes that appear entirely unrelated to, and geographically distant from, the content itself. Analogous routing centralization also exists in other systems, e.g., a single AS can hijack a large share of Bitcoin mining power [1, 45] or connections of Tor users [41].

Routing centralization enables large-scale attacks against multiple targets of the dependent service simultaneously, e.g., [1, 35, 45, 41]. Given that a malicious AS can be naturally on-path of IPFS components or places itself there via hijacking, we focus on the *availability* of content, the one property for which IPFS has no built-in protection.³ Concretely, we study routing-based *content censorship*, a threat particularly relevant to nation-state adversaries who may also control malicious ASes [32]. The threat is legitimate, as IPFS was already censored once during the 2017 Catalan referendum, when the Spanish government attempted to block an IPFS-based voting application [26]. To assess this threat, we measure the *live* IPFS topology, simulate routing attacks over it, and evaluate the censorship against content retrieval; see Figure 1. To censor a content requester, the adversary must block its connections to peers that cache the content and serve it via Bitswap, then block its connections to either the peers holding provider records or the providers themselves directly, depending on the specific content. Thus, when targeting multiple requesters or contents, blocking both resolvers and providers yields the maximum censorship.

³ IPFS has self-certifying CIDs for integrity and encrypted channels for confidentiality.

Our results reveal that complete censorship is possible against a small set of contents (e.g., only 0.7–7.8% of tested CIDs) when involving a disruptive hijacking of several thousand IP prefixes. We find that partial censorship targeting specific requesters appears to be far more achievable, e.g., more than 75% of CIDs can be blocked from over 57% of requesters, and hijacking ≈ 100 IP prefixes already reaches at least 84% of the maximum blockage in every dataset. To mitigate the censorship, we show that a global collaborative content-provisioning protocol can reduce the achievable censorship to roughly 20% of requesters.

To summarize, we claim the following contributions:

- We characterize the routing centralization in the IPFS that lies in the communication between content requesters, content providers, and resolvers.
- We simulate routing attacks over the live IPFS topology to determine which content can be censored from all requesters, quantify the affected fraction for content that cannot be fully blocked, and measure the attack costs.
- We propose and simulate mitigations at both the individual-provider and network-wide levels that strengthen content availability against routing attacks.

2 Background

2.1 IPFS Network

IPFS operates over a peer-to-peer (P2P) network connected via libp2p [23]. IPFS also includes a few centralized components to meet performance demands and enable clients (e.g., web browsers) to retrieve content without running a node.

Nodes. Each node holds a unique and stable *peer ID* derived from its public-private key pair. In the P2P network, a node’s peer ID binds to one or more concrete endpoints (IP address, transport protocol, port) while an IP address may host many peer IDs (e.g., nodes behind a NAT or co-located on a machine).

Connections. Nodes locate other peers via a standard Kademlia distributed hash table (DHT) [27]. A newly joined node queries pre-configured bootstrap nodes for lookups of random peer IDs and its own ID to populate its routing table and confirm ID uniqueness. Nodes reachable by others act as DHT *servers*, maintaining a routing table of up to $k = 20$ peers per prefix bucket and answering routing queries; unreachable nodes act as *clients*. A server node typically has a few hundred peer connections over secure channels (e.g., TLS 1.3 [24]).

Centralized components. To reduce lookup latency, IPFS includes a few centralized components as a shortcut around the P2P network [47]. Notably, an *InterPlanetary Network Indexer* (IPNI) is a high-performance server that maps CIDs directly to content providers, bypassing the DHT walk that lookups otherwise require. IPNI endpoints (e.g., `cid.contact`, enabled by default in Kubo) act as resolvers alongside DHT peers. Also, *HTTP gateways* are server nodes that let users fetch content over ordinary HTTP without running a node.

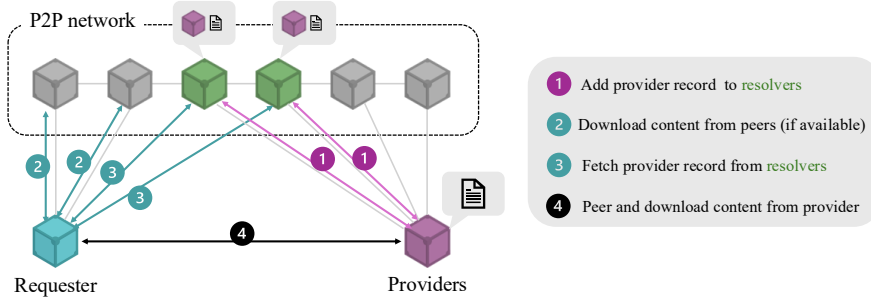


Fig. 2: Publishing and retrieving content in IPFS. The content provider publishes provider records (content C plus its peer ID) to k peers closest to C , called resolvers (1). The requester first asks for content from its peers (2). Otherwise, the requester uses DHT walks to find resolvers and fetch provider records (3). The requester then connects to the provider and downloads C via Bitswap (4).

2.2 Publishing and Retrieving Content

Content. A file in IPFS is represented as a Merkle-directed acyclic graph with fixed-size chunks as leaves. Each chunk receives a *Content Identifier* (CID), a label produced by hashing the chunk together with some metadata. CIDs are immutable (since any change to a chunk yields a new CID), self-certifying for data integrity, and host-agnostic for retrieval without specifying content location.

Publishing content. Figure 2 illustrates content publishing and retrieval in IPFS. To publish content, a provider creates a *provider record* binding its peer ID to the CID C , then performs a DHT walk to the $k = 20$ peers whose IDs are closest (in XOR distance) to C and pushes the record to them (step 1). These peers, called *resolvers* [5], are the lookup bridge between requesters and providers. IPFS adds robustness through ephemeral replication, in which a node that downloads content may itself serve it for a few hours while it remains cached, and a node that *pins* content publishes its own provider record to serve it persistently. Dedicated pinning services (e.g., Filebase [15], Pinata [30]) also host content on highly available nodes for a fee, adding alternative providers for the original content.

Retrieving content. A requester first asks its already-connected peers for C via the Bitswap protocol [34], in which peers exchange *want* and *have* lists of CIDs (step 2); this resolves popular content in seconds but is unreliable for new or rare CIDs. Failing that, the requester walks the DHT to the resolvers for C to discover its providers, opportunistically requesting the content and provider records from nodes it encounters, until 20 providers are found or all k resolvers have been queried (step 3). It then resolves the providers' multiaddresses through a further DHT query, connects to a provider, and downloads C over Bitswap (step 4).

2.3 Routing Attacks

There are almost 80K Autonomous Systems (ASes) that use the Border Gateway Protocol (BGP) [33] to exchange reachability for more than 1M IP prefixes [44].

BGP. Under BGP, each AS originates one or more IP prefixes through route advertisements, which are then propagated hop-by-hop, each prepending its own AS number, until all ASes learn the routes to forward traffic toward the prefixes. Route propagation and selection follow inter-AS business relationships, chiefly customer-provider and peer-to-peer [10]. For route propagation, the provider exports all its best routes to the customer and announces the customer's prefixes to all its neighbors while two peering ASes exchange traffic solely between their customer ASes and their own prefixes. For route selection, an AS prefers customer-learned routes over peer- and provider-learned ones, breaking ties by shortest AS path and then using arbitrary criteria (e.g., smaller AS number) [16].

Passive interception. Any AS on the path between two nodes observes network-level metadata (e.g., headers, packet sizes) and can delay or drop packets, turning transit into a denial-of-service and, for IPFS endpoints, into content censorship. We call this attack *passive* since it does not need routing manipulation. Its effectiveness depends on the AS's topological position, i.e., the fraction of connections that naturally traverse it. It is also worth noting that because BGP routing is asymmetric, the attacker needs only sit on one direction of the routing path [41].

Active BGP hijacking. BGP lacks authentication; thus, a malicious AS can inject bogus routes, known as a *BGP hijack*, to divert and intercept traffic for a target prefix [37]. If the bogus announcement is more specific than the legitimate one, the adversary attracts all traffic for that sub-prefix because BGP prefers the most specific matching route.⁴ An equally-specific announcement instead competes directly with the legitimate one, capturing a share of traffic determined by the relative topology, e.g., about 50% when the attacker-to-victim and origin-to-victim paths are comparable in length [17]. Hijacks also vary in complexity: an *origin* hijack announces the victim's prefix under the attacker's own AS, whereas a *forged-origin* hijack spoofs the victim's AS as the origin. The Resource Public Key Infrastructure (RPKI) [25] cryptographically binds each prefix to its legitimate origin AS and caps the prefix length it may announce (the ROA max-length). When widely deployed, it blocks origin hijacks and over-length announcements, but a forged-origin announcement up to the max-length stays valid, so RPKI stops neither forged-origin hijacks nor passive interception.

3 Routing Centralization in IPFS

In this section, we first define the threat model (§3.1), then describe how to find the targets of given CIDs (§3.2). We then show the concentration of those targets on the live IPFS network (§3.3) and their exposure to BGP hijacking (§3.4).

⁴ Operators usually filter prefixes longer than /24 for IPv4 and /48 for IPv6 [20].

3.1 Threat Model

Attack goals. The adversary aims to undermine the *availability* of content stored in the IPFS network, identified by its CID. At scale, it can disrupt many CIDs at once, rendering entire services inaccessible. Alternatively, it can target specific content accessed by particular user groups, motivated, e.g., by geopolitical interests (such as the Spanish censorship [26]), blocking malicious or abusive material [43], or enforcing copyright [36].

Attack capabilities. We consider a *malicious AS* who can tamper with any traffic traversing it. The malicious AS can also mount BGP hijacking (cf. §2.3) to draw onto its paths traffic it would not otherwise carry. Such capabilities are consistent with prior routing attacks against other systems [1, 35, 45, 41, 7], and are within reach of ISPs, transit networks, and nation-state adversaries [28, 48]. A determined attacker can even establish a new AS with purchased transit for a few thousand dollars [42]. We further assume the adversary operates a few IPFS nodes that cost a few dollars per month [29], to observe inbound peer connections and query the network for peers serving a given CID. We consider a single malicious AS at a time; our results are therefore a lower bound on what a coalition or a nation-state controlling multiple domestic ASes achieves (cf. §6). We leave multi-adversary analysis to future work.

3.2 Discovering Providers, Resolvers, and Caching Peers

Since a requester obtains content either directly from peers that cache it or from its registered providers, which it discovers through resolvers holding provider records, censorship requires disrupting its connections to these sources. Thus, for a given CID, the adversary must identify three targets: (i) the IPFS nodes currently caching the content and able to serve it via Bitswap, (ii) the content providers, and (iii) the resolvers responsible for discovering the providers. These compose the two attack vectors we study throughout, i.e., the *provider vector* (providers together with caching peers) and the *resolver vector* (resolvers together with caching peers). Blocking either vector denies the requester a distinct route to the content, and combining them maximizes the censorship (§4.3).

The reconnaissance proceeds in three steps. To find caching nodes, the adversary runs an IPFS node with its peering limits removed (i.e., a supernode with an unlimited number of connections), queries all its peers for the target CID via Bitswap `want` messages, and records the addresses that respond. To find resolvers, the adversary looks up the 30 DHT peers closest in XOR distance to the CID⁵ and adds the default IPNIs (e.g., `cid.contact` in Kubo v0.21.0). Finally, the adversary performs a standard CID lookup to yield the providers' peer IDs and multiaddresses, and all discovered IPv4/IPv6 addresses are mapped to their prefixes and ASes.⁶

⁵ The 20 nearest are the default provider resolvers, and we include 10 extra as a safety margin against peers that joined since the last provider announcement.

⁶ A more aggressive attacker could additionally crawl the DHT and actively connect to further nodes; we omit this for comparability.

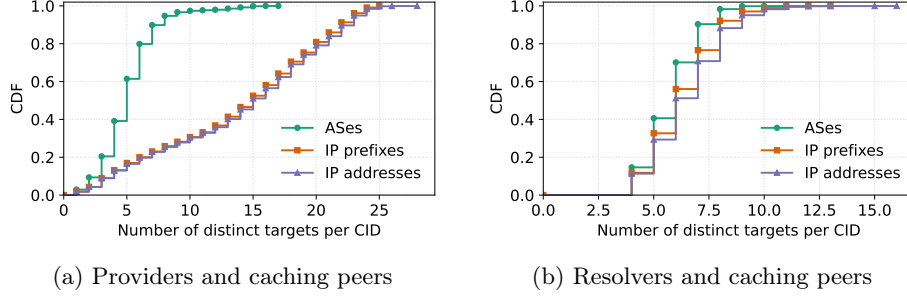


Fig. 3: CDF of distinct ASNs, IP prefixes, and IP addresses per CID across the three datasets (1,000 CIDs each). For the provider vector (a), most CIDs concentrate in a few ASes (61% in five or fewer) and prefixes (81% in twenty or fewer); for the resolver vector (b), the distinct-IP count usually falls below the 20-way replication, since resolvers are sampled from an already concentrated hosting population and several peer IDs share one address or prefix.

Implementation. We instantiate this reconnaissance by extending the Bitswap monitoring infrastructure of [4] with additional nodes, some accepting unlimited connections. The monitors log incoming Bitswap wantlists, and a rolling FIFO window selects CIDs frequently requested by many independent peers, which we take to be of high relevance. For each selected CID, we resolve its providers via a DHT lookup and its caching peers via the unlimited-connection nodes, additionally querying the same CID from monitors kept at default limits to capture caching peers visible only to ordinary nodes. All discovered addresses are mapped to prefixes and ASes with pyasn [3] (snapshot of June 24, 2025). In total, we collected three datasets of 1,000 CIDs each in July 2025, one week apart, which took about a day per dataset.

3.3 Concentration in ASes and IP Prefixes

We reconfirm that IPFS content delivery is highly concentrated at the hosting layer [4, 46, 38]. Figure 3 shows, for both attack vectors, the per-CID distribution of distinct ASes, IP prefixes, and IP addresses across the fused three datasets. A median CID is served from only 5 ASes and 15 prefixes on the provider vector, and 6 ASes and 6 prefixes on the resolver vector. Also, for provider nodes, 61% of CIDs are hosted within only five ASes and 81% within twenty prefixes.

We note two censorship-related observations. First, only a thin tail of CIDs is genuinely well distributed and is harder to block completely, e.g., just 3% reach ten or more provider ASes. Second, resolvers are *far more* concentrated than providers. Despite having 20 replications of every provider record, they are hosted on only a median of six distinct prefixes, thirteen prefixes at maximum, and only four to twelve ASes. One possible reason is that many IPFS server nodes may share a NATed IP address. Yet, we can confirm that replicating a provider record does not yield the expected diversity at the routing layer.

3.4 Exposure to BGP Hijacking

The degree of concentration bounds the number of targets an adversary must attack, but it does not determine how attackable each target is. A prefix that can be hijacked irrespective of the attacker’s location is considerably more valuable to the adversary than one that can only be contested from a favorable topological position. We therefore characterize a second dimension of the attack surface, namely the exposure of provider, resolver, and caching-peer prefixes to BGP hijacking. To this end, we retrieve the RPKI status of each prefix from the IRR Explorer endpoint [8] and classify it into one of the following four categories:

1. unprotected, prefix shorter than /24 (IPv4) or /48 (IPv6);
2. unprotected, prefix exactly /24 or /48;
3. RPKI-protected, announced shorter than the maximum length authorized by its RPKI record (the ROA max-length);
4. RPKI-protected, announced at the ROA max-length.

These categories differ in whether a hijack succeeds *independently* of the attacker’s topological position. In category 1, the attacker announces a more-specific /24 (resp. /48) covering the target and, by BGP’s longest-prefix-match rule, diverts all traffic regardless of position. Category 3 is analogous, as announcing up to the ROA max-length remains RPKI-valid and again captures all traffic. Categories 1 and 3 are thus *position-independent*. In category 2, a more-specific announcement would be filtered by standard operator practice [25], so the bogus route competes at equal specificity and wins only where the attacker’s AS-path is shorter. In category 4, no valid more-specific route exists, and the attacker wins only where it sits at least one hop closer to the target than the legitimate origin. Categories 2 and 4 are therefore *position-dependent*, and their success is evaluated per attacker AS in §4.3.

Figure 4 reports how the prefixes of both attack vectors distribute across these categories over the fused three datasets. The concentrated attack surface is also overwhelmingly exposed. For the provider vector, 81% of target prefixes fall into the position-independent categories 1 and 3 (4% in category 1, 77% in category 3), hijackable from any vantage point, leaving only 19% position-dependent. The resolver vector is better protected but still exposed, with 59% of its prefixes position-independent. The vectors differ most sharply at the margin that decides a *complete* block, that is, whether *every* prefix of a CID can be hijacked from a single position. For 5% of CIDs, the entire provider vector is position-independent, so one well-placed announcement suffices to block it. For the resolver vector, this never occurs, as 99.7% of CIDs retain at least one prefix in *each* of the well-protected categories 2 and 4. Fully hijacking every provider of a CID is, therefore, markedly easier than hijacking every resolver. Across both the concentration and the exposure measured above, IPFS content delivery thus rests on a small set of targets that are largely defenseless against routing attacks. The rest of the paper quantifies the censorship this enables, evaluating the position-dependent cases per attacker AS in §4.3.

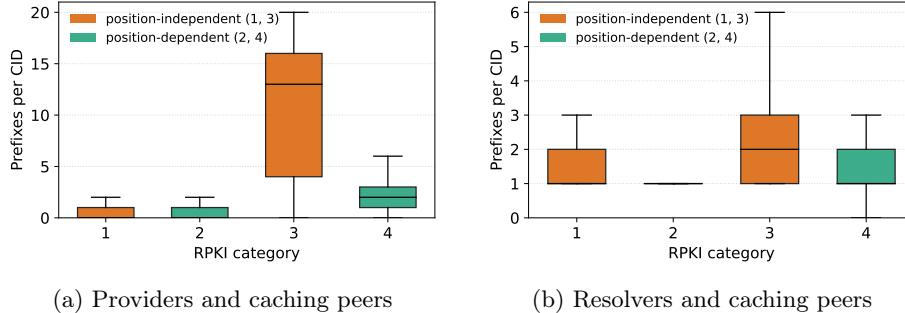


Fig. 4: Distribution of target IP prefixes across the four RPKI categories, combined over the three datasets (3,000 CIDs). For both vectors, most prefixes are position-independently hijackable (categories 1 and 3); the resolver vector (b) always retains at least one prefix in each well-protected category (2 and 4), making a complete hijack harder than for the provider vector (a).

4 Routing-Based Censorship of IPFS Content

Having established that IPFS content delivery is concentrated in a few ASes and IP prefixes (§3.3) that are largely unprotected against BGP hijacking (§3.4), we now evaluate the effectiveness of routing-based censorship attacks. We first describe the attack strategies and the setup of our simulation (§4.1). We then quantify whether content can be blocked from all requesters (§4.2), from a majority of requesters (§4.3), and at what cost in hijacked prefixes (§4.4).

4.1 Attack Vectors and Simulation Setup

Passive interception. An AS passively intercepts a connection when it lies on the BGP-selected path in either direction between the requester and target. Recall that since the two directions route independently, lying on either suffices. We compute these paths with the implementation of [2], which builds an AS-level topology from the CAIDA AS-relationship [10] and IXP [11] datasets and derives a routing tree, resolving competing paths by shortest AS-path length and breaking ties by smaller AS number [17]. This model omits local preference and BGP communities but approximates real Internet routing well for routing-attack analysis [45].

Active BGP hijacking. The malicious AS announces a bogus route toward a target prefix to attract the requester’s traffic. Here, we hijack only the traffic from requesters to provider and resolver prefixes, but not the other way around, because hijacking traffic toward the requester scales poorly and is more readily detected. The outcome follows the prefix’s RPKI category (§3.4). Specifically, the position-independent categories 1 and 3 are hijacked from every requester, whereas for the position-dependent categories 2 and 4, we locate the intercepting ASes through a breadth-first search starting from the requester.

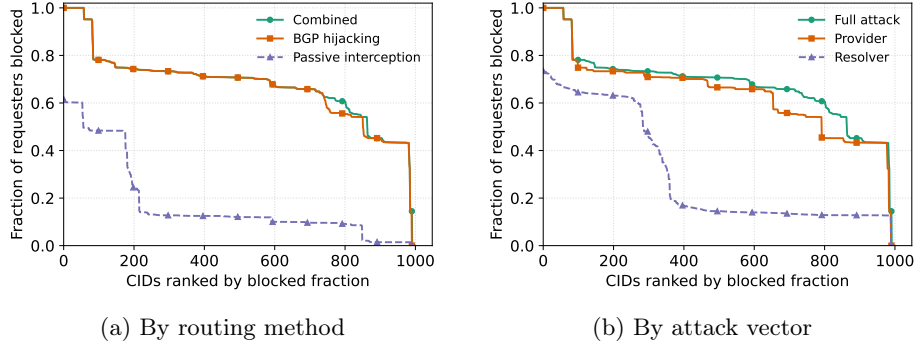


Fig. 5: Maximum fraction of requesters blocked by a single AS for dataset 1, with CIDs sorted by that fraction. (a) BGP hijacking blocks a median of 71% of requesters, far exceeding passive interception; their combination adds little. (b) The provider vector dominates the resolver vector, and the full attack rarely exceeds the provider vector alone.

Requester set. To enumerate the nodes that could act as requesters, we use the crawler of [5], which recursively queries each peer’s DHT routing table to reconstruct the server-node connection graph. A full crawl completes in about five minutes, so we treat the topology as static, a standard peer-to-peer assumption. We ran two crawls, on 12 and 13 July 2025, and merged and deduplicated by IPv4 and IPv6 address. The second crawl recovers momentarily unreachable nodes, and deduplication avoids overweighting services that run many nodes behind one address. Client nodes are absent from DHT routing tables, so the analysis is restricted to server nodes (including gateways, which run ordinary server nodes) and characterizes availability toward IPFS server nodes rather than end users. Each requester address is mapped to its AS and prefixes as in §3.2.

Attacker set. For the adversary, we consider the 100 highest-ranked ASes by ASRank [9] that have stronger connectivity and thus a more favorable position for both interception and hijacking. This bounds the simulation’s cost and avoids overfitting to obscure ASes, while retaining those within reach of an ISP or nation-state adversary. We model a single adversary AS acting alone and note that additional colluding ASes could only extend its reach; thus, our results are a lower bound with respect to adversary size. When one of our default-limit monitor nodes (i.e., an ordinary IPFS node) observed a caching Bitstamp provider that the attacker’s supernode with unlimited connections did not (§3.2), we conservatively treat the CID as unblockable, since the adversary cannot suppress a caching peer it never discovered. We observe that this edge case occurred for only 107 (or 3.5%) of the 3,000 CIDs.

4.2 Feasibility: Complete Blockage

We first ask whether any CIDs can be blocked from all requesters simultaneously, and if so, for what fraction of CIDs such complete blockage is achievable. Only a small fraction of CIDs, between 0.7% and 7.8% across the three datasets, can be blocked entirely, with considerable variation between datasets that is plausibly explained by the modest sample size and by dependence among samples. Every CID that could be blocked completely lacked a provider on a well-protected prefix. Indeed, all of their provider prefixes fell into the position-independent categories 1 and 3, which are hijackable from any location and therefore impose no requirement that the adversary be near the requester. As Figure 4 shows, category 3 in particular, namely RPKI-protected prefixes announced below their max-length, accounts for a large share of provider prefixes.

4.3 Effectiveness: Partial Blockage

For the majority of CIDs that cannot be blocked entirely, we ask what fraction of requesters a single adversary can nonetheless censor. Figure 5 reports the maximum such fraction for dataset 1, broken down by routing method (Figure 5a) and by attack vector (Figure 5b). Passive interception alone blocks fewer than 20% of requesters for most CIDs (median 12%), whereas BGP hijacking blocks a median of 71% and exceeds 70% for a majority of CIDs, while combining hijacking with passive interception adds little (median 71%). The two vectors target the two stages of content retrieval, i.e., the provider vector attacks the delivery of the content blocks, and the resolver vector attacks the discovery of the provider records. Delivery is the more exposed stage here, as the provider vector blocks a median of 66% of requesters, in line with the RPKI exposure of provider prefixes (§3.4), whereas the resolver vector alone reaches only a median of 15%. Adding the resolver vector raises the median blocked fraction from 66% to 71%, so the full attack rarely improves meaningfully on the provider vector alone.

The effectiveness of hijacking depends strongly on the adversary. Averaged over dataset 1, the blockage rates of the 100 highest-ranked ASes differ markedly from one another. Three ASes, namely AS174, AS1299, and AS1031, stand out, blocking on average 67%, 55%, and 44% of CID-requester pairs respectively, whereas most others reach only about 30%. Well-connected ASes such as AS174 therefore hold a pronounced advantage as censorship attackers.

4.4 Cost: Minimal Prefix Hijacking

In BGP hijackings, each bogus announcement increases the adversary’s exposure and detectability. A rational adversary, therefore, may want to minimize the number of hijacked prefixes while still maximizing censorship. To estimate this trade-off, we apply a greedy prefix-selection algorithm that repeatedly adds the prefix yielding the largest immediate gain in blocked requesters and, when no prefix yields an immediate gain, the prefix that is closest to producing one.

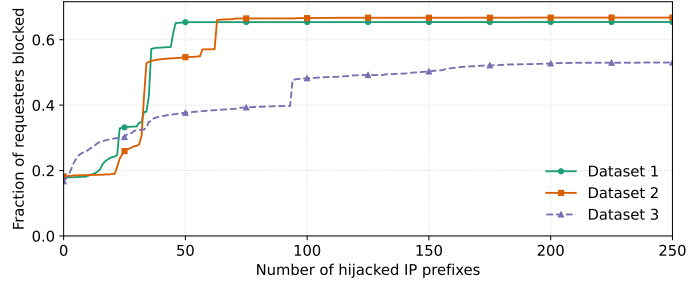


Fig. 6: Average fraction of requesters blocked as a function of the number of hijacked IP prefixes, per dataset. A small number of prefixes achieves most of the blockage that unlimited hijacking attains.

As Figure 6 shows, a small number of prefixes suffices. The first 100 of the 2,791 distinct target prefixes already achieve most of the censorship. Specifically, in each dataset, they reach at least 84% of the maximum blocked fraction that unlimited hijacking attains, after which additional prefixes yield diminishing returns. This confirms the practicality of the attack.

5 Countermeasures

Censorship attacks are effective because the content providers are concentrated in a few, unprotected ASes and IP prefixes that a single AS can hijack or intercept. The primary defense is therefore to place at least one provider behind an RPKI-protected, max-length prefix, which mitigates the hijacking impacts. Furthermore, replicating content across providers diverse in ASes and IP prefixes also makes intercepting and hijacking all traffic more difficult for the adversary in general. We evaluate two forms of such replication, i.e., content-owner-driven IPFS clusters (§5.1) and a network-wide collaborative-pinning scheme (§5.2).

5.1 AS- and Prefix-protected IPFS Clusters

IPFS clusters let a set of nodes collaboratively pin a document to a configurable replication count [21]. A content owner can thus realize both defenses at once, spreading the providers of a CID across ASes and prefixes while placing some behind RPKI-protected, max-length prefixes.

We examined three public clusters listed on `collab.ipfscluster.io` and found their AS and prefix spread to vary widely.

- *Project Gutenberg (Spanish)* spans 15 ASes with 23 IPv4 and 7 IPv6 prefixes.
- *IPFS Websites* spans 2 ASes with 5 IPv4 and 4 IPv6 prefixes.
- *Wikipedia* spans a single AS with 4 IPv4 and 3 IPv6 prefixes.

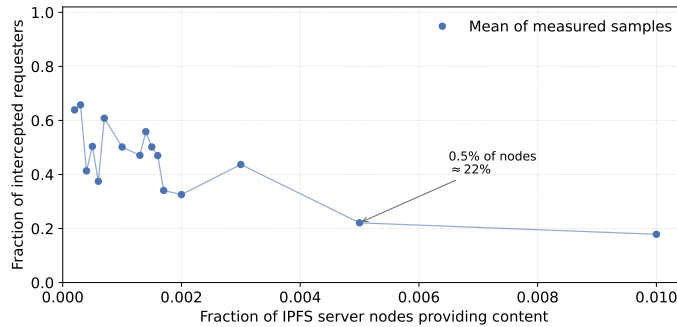


Fig. 7: Maximum fraction of requesters intercepted by the single most effective top-100 AS, versus the fraction of IPFS server nodes participating in network-wide collaborative pinning of a CID (100% = 16,107 reachable server nodes).

We simulated the censorship attack against 1,000 CIDs from each cluster, and it blocked no requesters. AS diversity does not explain this resilience, since the Wikipedia cluster is served entirely from AS54825. Instead, every cluster serves content from 147.75.63.0/24, a max-length (/24) prefix in AS54825 that a ROA protects under RPKI, which defeats the more-specific and origin-hijack variants our attack relies on. A single provider behind an RPKI-protected, max-length prefix therefore defeats the hijacking attack on its own, and distributing providers across additional prefixes hardens the content further against passive interception, since the adversary must then occupy several disjoint paths at once.

5.2 Network-wide Collaborative Pinning

IPFS clusters place the burden of protection on content owners, which conflicts with the goal of making all content censorship-resistant by default. We therefore propose to generalize the cluster idea at the protocol level. Particularly, any node asks the network to pin a CID at a default replication factor, and a random subset of nodes stores a copy. In return, the requesting node stores content for others in proportion to what it pins, namely the pinned volume times the replication factor, making this scheme reciprocal instead of altruistic. What remains open is enforcement, that is, the accounting and penalty mechanism that makes this obligation binding on selfish or non-compliant nodes. We therefore quantify only the protection that random, network-wide replication provides, and we assume that every IPFS server node pins a given CID with equal probability.

Figure 7 reports the maximum fraction of requesters that the most effective top-100 AS can intercept as replication grows. Replicating a CID to 0.5% of server nodes (or 80 nodes) drives the achievable interception down to roughly 20%. Beyond this point, additional replication yields little. The obligation that an enforcement mechanism must sustain is correspondingly small, at roughly 80 stored copies per CID across the entire network.

We note that random replication alone cannot mitigate the censorship attacks. Indeed, a single well-placed AS still occupies the paths to some providers; yet, the content spreads widely. Combining random replication with a few strong nodes that serve content over RPKI-protected, max-length prefixes (§5.1) lowers interception further, toward zero. In sum, broad decentralization limits passive interception, and the hardened prefixes resist hijacking.

6 Discussion

Sampling and impact weighting. Our crawled, IP-deduplicated requester set weights every requester and CID equally, whereas an IPFS node may serve millions of end users or none. Our results may therefore not capture the true severity of user impact. Weighting requesters by observed Bitswap demand or gateway request volume is a natural next step.

Static snapshot and self-healing. Our analysis rests on a single snapshot and ignores IPFS’s ephemeral re-provisioning, in which every successful fetch briefly turns a requester into a provider. Against a static adversary, re-provisioning only adds reachability, so our reported figures remain an upper bound on the blocked fraction. Even so, the requesters that re-provide belong to the same hosting population as the original providers, so a new provider often appears in a prefix the adversary already hijacks. Measuring how much relief persists over time requires a longitudinal study, which we leave to future work.

Real-world feasibility. Our idealized model best represents *local* censorship, e.g., within one country, rather than a globally sustained attack that would attract scrutiny. Where an adversary controls every path to the targeted requesters, as a nation-state can domestically, IPFS offers no routing-layer protection, and users must tunnel out through proxies such as VPNs or Tor.

Hardening content discovery. Our mitigations focus on content providers (§5), because the content owner can place them behind RPKI-protected, max-length prefixes. Content discovery admits no such mechanism, because the resolvers for a CID are fixed by the content hash and lie outside any owner’s control, and IPNI endpoints are centralized by design. Hardening content discovery therefore remains an open problem.

Ethics. Our measurements do not affect IPFS users in any way. All evaluation results come from simulation over the public AS-level topology and our collected IPFS network data. We reported our findings to the IPFS maintainers on 7 September 2026, following their security policy.⁷

⁷ <https://github.com/ipfs/community/blob/master/SECURITY.md>

7 Related Work

7.1 Measurements of the IPFS Network

Henningsen et al. [18] introduced a DHT-crawling technique, since adopted by others, including us [4], and reported an average of 44,000 nodes, most operated by private individuals and only a fraction reachable. Because crawlers cannot observe client nodes, which are absent from the DHT, an alternative is to run well-connected server nodes and monitor incoming connections [18, 14]. Determining which CIDs are present is itself non-trivial, since nodes answer only for CIDs they are queried about. Several works, including ours, mine Bitswap traffic [4, 5, 38] or gateway request logs [39] to this end, and a separate line of work measures the performance of IPFS operations [14, 22, 39, 38, 46].

7.2 Censorship and Centralization in IPFS

Prior work on IPFS censorship targets the *DHT and identity layer*. Sybil attacks eclipse individual nodes [31] or deny access to specific content [13], and Sridhar et al. [40] exploit a flaw in Kademlia provider-record resolution that lets a single host cheaply censor a chosen CID; detection and mitigation for this attack are now deployed in IPFS. Our work is orthogonal. We attack the *routing layer*, which the deployed identity-layer defenses do not alter. The enabling condition is centralization. Several studies document IPFS’s convergence onto a few operators [5, 38, 39, 47], with a handful of cloud providers now hosting more than 75% of nodes [38] and IPFS’s own components concentrating it further. We show that this hosting concentration also collapses onto a few ASes and IP prefixes, which turns it into effective and large-scale routing censorship attacks.

7.3 Routing Attacks on Decentralized Systems

BGP routing attacks are well studied for other decentralized systems, though not, to our knowledge, for IPFS. Hijacks can partition cryptocurrency networks [1, 35], and Tran et al. [45] intercept Stratum connections between miners and pools to waste miner effort. This last attack shares our core mechanism of denying a requester the path to its provider, but applies it to a different system and objective. We adapt this mechanism to IPFS and quantify its effectiveness.

8 Conclusion

IPFS promises censorship-resistant storage through decentralization, yet in practice, its content providers concentrate in a few ASes and IP prefixes that are largely unprotected by RPKI. This paper shows that such centralization enables a routing-based censorship attack: over the live IPFS topology, a single AS-level adversary hijacking roughly 100 IP prefixes can block more than 75% of all tested content from over 57% of requesters. We hope this work encourages the IPFS community to treat routing-layer centralization as a first-class threat and to harden both content delivery and discovery against it.

Acknowledgment

We thank anonymous reviewers for their helpful feedback on the paper and the Networked Systems Group (ETH Zurich) for facilitating this research. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

1. Apostolaki, M., Zohar, A., Vanbever, L.: Hijacking bitcoin: Routing attacks on cryptocurrencies. In: Proc. IEEE S&P (2017)
2. von Arx, T., Tran, M., Vanbever, L.: Revelio: A network-level privacy attack in the lightning network. In: Proc. IEEE EuroS&P (2023)
3. Asghari, H., Noroozian, A.: pyasn: Offline IP to ASN Lookup Tool (2023)
4. Balduf, L., Henningsen, S., Florian, M., Rust, S., Scheuermann, B.: Monitoring Data Requests in Decentralized Data Storage Systems: A Case Study of IPFS. In: Proc. IEEE ICDCS (2022)
5. Balduf, L., Keizer, N.V., Korczyński, M., Pavlou, G., Król, M., Ascigil, O., Scheuermann, B.: The Cloud Strikes Back: Investigating the Decentralization of IPFS. In: Proc. ACM IMC (2023)
6. Benet, J.: IPFS - Content Addressed, Versioned, P2P File System (2014)
7. Birge-Lee, H., Sun, Y., Edmundson, A., Rexford, J., Mittal, P.: Bamboozling certificate authorities with BGP. In: Proc. USENIX Security (2018)
8. BV, D., NLNOG, S.: IRR Explorer. <https://irrexplorer.nlnog.net> (2026)
9. CAIDA: AS Rank: Ranking Autonomous Systems Dataset (2026)
10. CAIDA: AS Relationships Dataset (2026)
11. CAIDA: Internet Exchange Points (IXPs) Dataset (2026)
12. Cho, S., Fontugne, R., Cho, K., Dainotti, A., Gill, P.: BGP hijacking classification. In: Proc. IEEE TMA (2019)
13. Cholez, T., Ignat, C.L.: Sybil Attack Strikes Again: Denying Content Access in IPFS with a Single Computer. In: Proc. ARES (2024)
14. Daniel, E., Tschorsch, F.: Passively Measuring IPFS Churn and Network Size. In: Proc. IEEE ICDCSW (2022)
15. FileBase: The InterPlanetary Development Platform (2026)
16. Gao, L., Rexford, J.: Stable Internet routing without global coordination. IEEE/ACM TON (2001)
17. Goldberg, S., Schapira, M., Hummon, P., Rexford, J.: How secure are secure inter-domain routing protocols. ACM SIGCOMM CCR (2010)
18. Henningsen, S., Florian, M., Rust, S., Scheuermann, B.: Mapping the Interplanetary Filesystem. In: IFIP Networking Conference (2020)
19. IPFS: IPFS Ecosystem directory. <https://ecosystem.ipfs.tech/> (2026)
20. Karlin, J., Forrest, S., Rexford, J.: Pretty good BGP: improving BGP by cautiously adopting routes. In: Proc. IEEE ICNP (2006)
21. Labs, P.: IPFS Cluster. <https://ipfsccluster.io/> (2026)
22. Lajam, O.A., Helmy, T.A.: Performance evaluation of ipfs in private networks. In: Proc. DSDE (2021)
23. libp2p: libp2p. <https://docs.libp2p.io> (2026)
24. libp2p: TLS. <https://docs.libp2p.io/concepts/secure-comm/tls/> (2026)

25. Madory, D.: RPKI ROV Deployment Reaches Major Milestone. <https://manrs.org/2024/05/rpki-rov-deployment-reaches-major-milestone/> (2024)
26. Malcolm, J.: No Justification for Spanish Internet Censorship During Catalanian Referendum. <https://www.eff.org/deeplinks/2017/10/no-justification-spanish-internet-censorship-during-catalonian-referendum> (2017)
27. Maymounkov, P., Mazières, D.: Kademlia: A peer-to-peer information system based on the XOR metric. In: Proc. IPTPS (2002)
28. Nakibly, G., Schcolnik, J., Rubin, Y.: Website-Targeted False Content Injection by Network Operators. In: Proc. USENIX Security (2016)
29. Pinata: How much does an IPFS Pinning Service Cost? <https://pinata.cloud/blog/how-much-does-an-ipfs-pinning-service-cost/> (2023)
30. Pinata: Crypto’s File Storage. <https://pinata.cloud/> (2026)
31. Prünster, B., Marsalek, A., Zefferer, T.: Total Eclipse of the Heart – Disrupting the InterPlanetary File System. In: Proc. USENIX Security (2022)
32. Raman, R.S., Shenoy, P., Kohls, K., Ensafi, R.: A Worldwide View of Network-Level Paths to Censorship. In: Proc. USENIX Security (2020)
33. Rekhter, Y., Li, T., Hares, S.: A Border Gateway Protocol 4 (BGP-4) (2006)
34. De la Rocha, A., Dias, D., Psaras, Y.: Accelerating Content Routing with Bitswap: A multi-path file transfer protocol in IPFS and Filecoin (2021)
35. Saad, M., Mohaisen, D.: Three birds with one stone: Efficient partitioning attacks on interdependent cryptocurrency networks. In: Proc. IEEE S&P (2023)
36. der Sar, E.V.: Cloudflare Disables Access to ‘Pirated’ Content on its IPFS Gateway (2023)
37. Sermpezis, P., Kotronis, V., Gigis, P., Dimitropoulos, X., Cicalese, D., King, A., Dainotti, A.: ARTEMIS: Neutralizing BGP hijacking within a minute. IEEE/ACM TON (2018)
38. Shi, R., Cheng, R., Fu, Y., Han, B., Cheng, Y., Chen, S.: Centralization in the Decentralized Web: Challenges and Opportunities in IPFS Data Management. In: Proc. ACM WWW (2025)
39. Shi, R., Cheng, R., Han, B., Cheng, Y., Chen, S.: A Closer Look into IPFS: Accessibility, Content, and Performance. Proc. ACM POMACS (2024)
40. Sridhar, S., Ascigil, O., Keizer, N., Genon, F., Pierre, S., Psaras, Y., Rivière, E., Król, M.: Content Censorship in the InterPlanetary File System. In: Proc. NDSS (2024)
41. Sun, Y., Edmundson, A., Vanbever, L., Li, O., Rexford, J., Chiang, M., Mittal, P.: RAPTOR: Routing attacks on privacy in Tor. In: Proc. USENIX Security (2015)
42. Swer, D.: How I set up my own Autonomous System (2022)
43. Tanner, A., Bleich, K., Galiette, A., Opacki, J.: Threat Actors Rapidly Adopt Web3 IPFS Technology (2023)
44. Tony Bates, Philip Smith, G.H.: CIDR Report. <https://www.cidr-report.org/as2.0/> (2025)
45. Tran, M., von Arx, T., Vanbever, L.: Routing Attacks on Cryptocurrency Mining Pools. In: Proc. IEEE S&P (2024)
46. Trautwein, D., Raman, A., Tyson, G., Castro, I., Scott, W., Schubotz, M., Gipp, B., Psaras, Y.: Design and Evaluation of IPFS: A Storage Layer for the Decentralized Web. In: Proc. ACM SIGCOMM (2022)
47. Wei, Y., Trautwein, D., Psaras, Y., Castro, I., Scott, W., Raman, A., Tyson, G.: The Eternal Tussle: Exploring the Role of Centralization in IPFS. In: Proc. USENIX NSDI (2024)
48. Wu, M., Zohaib, A., Durumeric, Z., Houmansadr, A., Wustrow, E.: A Wall Behind A Wall: Emerging Regional Censorship in China. In: Proc. IEEE S&P (2025)