# Netting Phish in the IPFS Ocean: Real-Time Monitoring and Characterization of Decentralized Phishing Campaigns

Anas Kastantin
KOR Labs / LIG, Univ. Grenoble Alpes
Grenoble, France

Leonhard Balduf
TU Darmstadt
Darmstadt, Germany

Onur Ascigil
Lancaster University
Lancaster, U.K.

Saidu Sokoto
City St George's, Univ. of London
London, U.K.

Björn Scheuermann
TU Darmstadt
Darmstadt, Germany

Andrzej Duda
KOR Labs / LIG, Univ. Grenoble Alpes
Grenoble, France

Michał Król
City St George's, Univ. of London
London, U.K.

Maciej Korczyński
KOR Labs / LIG, Univ. Grenoble Alpes
Grenoble, France

## Abstract

The InterPlanetary File System (IPFS) is the largest decentralized content-centric storage network. While its architecture enables resilient, distributed content delivery, it can be abused to host and disseminate malicious content. Public IPFS HTTP gateways further expand this threat surface, enabling attackers to deploy phishing websites and leverage gateway reputation to evade detection. This model can keep content available even after attackers go offline and challenges traditional phishing detection systems.

We present a framework for monitoring and characterizing phishing on IPFS, leveraging a measurement platform that integrates multi-source data, including IPFS traffic and passive DNS. Over 11 months, we detect 10,489 phishing CIDs, grouped into 448 phishing clusters. 80% of detected CIDs originate from only 69 clustered campaigns indicating that targeting a small number of dominant clusters could yield high mitigation leverage. We also identify 588 gateways involved in dissemination, including 573 outside public gateway lists, and show that attackers can exploit caching across reputable gateways to amplify attacks and extend content availability. Finally, we find that traditional Web phishing countermeasures and IPFS blocklists provide insufficient protection.

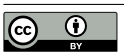
Our findings support practical mitigation and offer broader insights for trust and safety in decentralized web infrastructures.

## CCS Concepts

• **Security and privacy** → **Phishing**; *Web protocol security*; Distributed systems security; • **Networks** → *Network measurement*; *Peer-to-peer protocols*; Peer-to-peer networks.

## Keywords

IPFS, phishing, decentralized web, HTTP gateways, network measurement, passive DNS, threat intelligence




*WWW '26, Dubai, United Arab Emirates*

ACM ISBN 979-8-4007-2307-0/2026/04
https://doi.org/10.1145/3774904.3792188

**ACM Reference Format:**
Anas Kastantin, Leonhard Balduf, Onur Ascigil, Saidu Sokoto, Björn Scheuermann, Andrzej Duda, Michał Król, and Maciej Korczyński. 2026. Netting Phish in the IPFS Ocean: Real-Time Monitoring and Characterization of Decentralized Phishing Campaigns. In *Proceedings of the ACM Web Conference 2026 (WWW '26), April 13–17, 2026, Dubai, United Arab Emirates.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3774904.3792188

**Resource Availability:**
Source code of this paper has been made publicly available at https://doi.org/10.5281/zenodo.18351495.

## 1 Introduction

Interplanetary Filesystem (IPFS) is one of the largest platforms in the decentralized web, with ≈30,000 online IPFS nodes, spread across 2,700 Autonomous Systems (ASes), and 152 countries [47]. The platform is seeing growing uptake, with more than 3 M web client accesses and over 300 k unique nodes serving content in the peer-to-peer (P2P) network every week [17]. IPFS underpins an ecosystem of decentralized applications [9], including social networking [5, 10], data storage [13, 32, 38], content search [4, 36], messaging [46], streaming [2, 6, 51], gaming [8, 14], and e-commerce [3, 7]. It is also widely used as external storage for blockchain-based applications such as non-fungible token (NFT) platforms [15, 24].

Under the hood, IPFS is a content-centric P2P network in which each piece of content is identified by a content identifier (CID). Anyone can add content to the platform by advertising themselves as a *provider* for the corresponding CID. By default, anyone who downloads a piece of content automatically becomes its provider, enabling auto-scaling data dissemination.

P2P networks often host illegal or harmful content [21, 29]. Similarly, previous studies showed that IPFS contains terrorist propaganda, child sexual abuse material (CSAM), and copyrighted material [44]. However, the impact of IPFS malicious content may be significantly wider as: (1) public Hypertext Transfer Protocol (HTTP) gateways make IPFS content widely available over HTTP; (2) an attacker can hide behind public HTTP gateway domains, abusing their reputation to evade domain-level blocking; (3) the auto-scaling replication of IPFS allows an attacker to host phishing websites briefly, after which the content is automatically served by other nodes in the network and gateway caches; and (4) existing defenses

do not match well with IPFS content-addressed retrieval; blocklists such as Google Safe Browsing (GSB) operate at the host/path level; a malicious CID may thus appear as a new uniform resource locator (URL) on a different gateway, evading detection.

Protocol Labs introduced a moderation *BadBits denylist* that consists of hashed CIDs representing harmful content [39]. However, due to a partially manual moderation process, the BadBits denylist struggles to detect and react to malicious content in a timely manner [44]. This often means that phishing CIDs are blocked only once the corresponding campaign is finished. New systems for systematic monitoring and characterization of phishing on IPFS are thus essential to understand its scale, inform mitigation, and preserve user trust in decentralized infrastructures.

**This work.** In this paper, we propose a framework to combat IPFS phishing through multi-source real-time monitoring and the analysis of campaign clusters and gateway behaviors.

We introduce the first systematic measurement platform that continuously identifies and monitors phishing abuse on IPFS. The platform integrates diverse data sources, including CID requests propagated via the IPFS P2P network, passive Domain Name System (pDNS) monitoring, and the automated collection and classification of phishing websites linked to CIDs. This approach provides a broad view of phishing activity observed in our data. We discover 10,489 phishing CIDs—twice the number reported by existing public feeds (e.g., APWG, OpenPhish, PhishTank) and IPFS-specific lists.

We perform a thorough analysis of phishing campaigns on IPFS, revealing prominent clusters of activity. Approximately 80% of the detected phishing CIDs originate from only 69 clustered campaigns. This indicates that a small number of recurring campaign templates account for most activity and thus offer high potential for targeted mitigation. Our study illuminates tactics used to exploit the unique characteristics of IPFS to avoid detection and mitigation. In particular, attackers can generate new CIDs by making minimal and sometimes irrelevant alterations to the phishing content.

We then focus on IPFS gateways to understand how phishing content is accessed and how quickly providers block it. We identify 573 gateways previously unlisted in public gateway lists that can disseminate phishing websites. We also demonstrate how attackers can exploit gateway behavior and caching mechanisms to amplify phishing campaigns and increase their resilience to takedown. We then evaluate usage patterns with a survival analysis to determine blocking times.

Finally, we propose a CID-based phishing denylist that includes evidence of maliciousness. The list is fully compatible with the current IPFS blocklisting format, enabling easy uptake. We provide a set of recommendations for the ecosystem, including the optimal points of intervention and ways to enhance traditional Web phishing countermeasures to encompass content-centric threats.

## 2 Background

This section reviews IPFS and contrasts phishing delivery and mitigation on the Web and IPFS.

### 2.1 IPFS Primer

IPFS [20] is a set of protocols that facilitate decentralized content-addressable data storage and retrieval [16, 47]. At its core, it uses a content-based addressing scheme employing CIDs—hash-based, immutable, self-certifying names that decouple the address of the content from its storage location. Any peer can serve content for a CID, and clients can verify integrity by recomputing the hash.

**Content Discovery.** IPFS discovers content using a Kademlia-based [34] distributed hash table (DHT) that stores *provider records* mapping a CID to peers that can serve it. In parallel, peers query their directly connected neighbors with Bitswap [25] also used for the subsequent data transfer.

**Content Publication and Retrieval.** To publish content, a provider computes its CID and announces availability by publishing a provider record to the DHT. To retrieve content, clients query neighbors via Bitswap and query the DHT for providers, then fetch the object from an available peer using Bitswap.

**HTTP Gateways.** Accessing IPFS directly requires running an IPFS node, so many users rely on *HTTP gateways* that translate HTTP GET requests into IPFS retrieval. Upon receiving a request for a CID, a gateway fetches the content from IPFS and returns it over HTTP. Due to the immutability of content-addressed data, gateways can implement aggressive caching policies.

A CID can be requested from a gateway with the Fully-Qualified Domain Name (FQDN) **`gateway.com`** using one of two URL formats: 1) in the FQDN (e.g., **`https://`***cid*`.gateway.com/`**, the recommended method) or 2) in the URL path (e.g., **`https://gateway.com/ipfs/`***cid*).

### 2.2 Phishing Delivery and Mitigation

**Traditional Web.** Phishing is a social-engineering attack in which an adversary impersonates a trusted entity to deceive users into disclosing sensitive information (e.g., credentials) or performing harmful actions. Traditional Web phishing campaigns achieve scale by distributing hosting across three main channels: (i) attacker-controlled infrastructure [19, 33, 43], (ii) compromised legitimate websites [33, 45], and (iii) platform-mediated abuse, which includes the services of URL shorteners (e.g., **`bit.ly`**) [35] or platform-hosted subdomains (e.g., **`*.netlify.app`**).

Traditional phishing mitigation relies on URL blocklists and reporting feeds ingested by browsers, mail filters, and security vendors (e.g., APWG [1], OpenPhish [11], PhishTank [12]). These mechanisms are effective on the Web but are largely URL-scoped. Indicators reference specific hosts and paths, so small changes (subdomain, path) can yield fresh, uncovered URLs, and propagation speed varies across feeds.

**IPFS.** In contrast to the host-centric Web, IPFS uses content-based addressing. The same CID can be fetched via many public gateways, weakening traditional domain/URL/host-level defenses and enabling rapid migration when content is blocked. Cache persistence may outlive the original publisher, further complicating takedowns. Prior work [44] shows that while some gateways serving malicious content are blocked by GSB, many others remain reachable, allowing low-cost rotation. These properties prolong availability and reduce attacker effort.

In 2021, Protocol Labs introduced the *BadBits* denylist, a public denylist of content deemed "harmful" (e.g., copyright violations, phishing) for gateways to block [39]. To date, it includes $> 480,000$ entries, each a hex-encoded SHA256 of a normalized CID, enabling

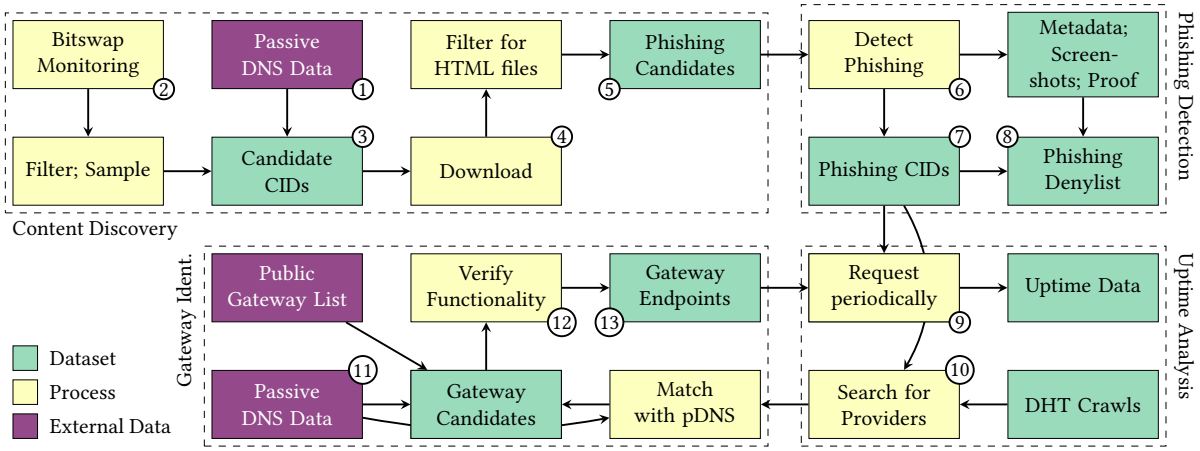



**Figure 1: System overview.**

membership checks without revealing the CID itself. The list carries no metadata: the BadBits denylist neither categorizes content nor provides evidence (likely to avoid CSAM handling). Because public gateways sit between IPFS and browsers, the delays or gaps in such denylists directly translate into Web user exposure.

## 3 Methodology and Collected Datasets

In this section, we describe our methodology, system design, and the datasets collected in this work. Figure 1 illustrates our end-to-end pipeline. Ethical considerations are discussed in Appendix C.

**Content Discovery and Fetching.** Fetching IPFS data requires the CID of the data. For that, we use two sources: *(i)* passive DNS CID discovery and *(ii)* Bitswap-based CID collection.

In the passive DNS CID discovery, we monitor SIE Europe pDNS[1] and identify 3.3 M CID strings embedded in subdomains using regular expressions (①).

For Bitswap-based CID collection, we passively monitor Bitswap request traffic using two nodes (②) without limiting the number of inbound connections. Related work [16] showed that this is sufficient to receive requests from the majority of the network. We filter the requests by source, only including the requests generated by gateways. While we likely monitor more than half of the entire network traffic, computational constraints force us to sample a fixed number of typically 100,000 CIDs per day, corresponding to $< 1\,\%$ of daily traffic. Our work thus forms a lower bound on the scale of phishing on IPFS. Through Bitswap monitoring, we collect a total of 32.5 M CIDs.

Both sources of CIDs (③) feed into the downloading and metadata extraction infrastructure (④). Between the 18th of September 2024 and the 13th of July 2025, we attempted to download a total of $40.1 \times 10^6$ objects and succeeded in 92.80 % of cases. Note that this number is *higher* than the number of submitted CIDs, as a CID can encode a single file, a directory, or a range of other data. We refer interested readers to related work for more information [16, 47].

We identify the content type of the downloaded files using the `libmime` Multipurpose Internet Mail Extensions (MIME) detector and discard any non-HTML content, which results in our dataset of phishing candidates (⑤).

**Phishing Detection.** We analyze our phishing candidates using a lightweight phishing detection module (⑥). Rather than proposing an end-to-end classifier from scratch, we benchmark three state-of-the-art approaches: visual logo matching (Phishpedia [30]), logo detection combined with Credential-Taking Intent (CTI) (PhishIntention [31]), and DOM-structure fingerprinting via Simhash [41]. Based on benchmarking on a labeled corpus of 5,000 pages, we select Simhash for its accuracy–efficiency trade-off, and tailor it to IPFS. We detail the process in Appendix A. In a nutshell, we hash each parsed HTML with SimHash and query a fingerprint index. A hit labels the page as phishing and assigns it to the matched campaign. On a miss, we automatically check for CTI signals (e.g., login/password forms). If CTI is present, we trigger a cold-start step: we manually verify the sample and add its resulting fingerprint to the index. We identify a total of 10,489 phishing CIDs (⑦). Each identified phishing CID is then added to a phishing denylist (⑧), together with a rendered screenshot for documentation.

**Gateway Identification.** To investigate IPFS phishing delivery paths, we enumerate HTTP gateways by combining three signals: *(i)* the public gateway list,[2] *(ii)* pDNS logs (Figure 1, ⑪), and *(iii)* *provider records* correlation with pDNS.

pDNS logs enable us to detect all the gateways embedding CIDs in FQDNs. However, this method misses the gateways that include CIDs in the URL path. For each phishing CID, we thus perform daily provider record lookups, store the returned peer identifiers and their observed IP addresses (⑩). We then query pDNS to recover hostnames/FQDNs seen for those IPs. These snapshots reveal which peers advertise each CID and let us link provider-record IPs to candidate gateway domains.

We validate candidates (the domains with CID-in-subdomain patterns and hostnames mapped from provider-record IPs) by fetching a widely shared reference CID used in the IPFS official tutorials[3]

[1] https://www.sie-europe.net/

[2] https://ipfs.github.io/public-gateway-checker/

[3] https://docs.ipfs.tech/how-to/command-line-quick-start/#initialize-the-repository

(⑫). The domains that resolve the reference are classified as functional IPFS gateway endpoints (⑬). We identify 588 gateways.

**Uptime Measurements.** We investigate how long each phishing CID remains available through both the IPFS P2P network and the gateways. For IPFS, we use the phishing CID provider records described above. For gateways, we periodically request our phishing denylist CIDs from the list of identified gateways (⑨). Every 2-3 days, we send probes that produce a log for every ⟨gateway, CID⟩ pair available at that time.

**Gateway Caching.** We then seek to distinguish between the cases when a gateway requests the content from IPFS or uses a cached copy. At the start of each run, we test whether a gateway is *proxied* via the Cloudflare CDN. We issue an HTTP request and classify the gateway as *caching*, if the response exposes Cloudflare-specific headers (e.g., `CF-Cache-Status`). CDN caching can make a CID appear available even if the origin gateway no longer serves it. We therefore record Cloudflare cache state before and after each fetch to distinguish CDN-served availability from true origin availability. For caching gateways, we measure twice the cache state for each CID. Immediately *before* the content fetch, we query the CID URL and record `CF-Cache-Status` as `HIT` or `MISS`, and immediately *after* the fetch, we repeat the query and record the status again. The CID retrieval itself is performed with the CDN instructed to ignore any cached copy.

The *pre-fetch* probe checks whether a copy of the CID is already present at the CDN, independently of the origin gateway condition. The *post-fetch* probe is issued only following a successful retrieval, capturing the cache state immediately thereafter. Together, these paired observations quantify the prevalence of pre-existing cached CIDs and the caching behavior following a successful retrieval.

## 4 Results

This section presents the results of our measurements and analyses. We begin by examining the overall collected content, followed by an assessment of IPFS activity. Finally, we illustrate how attackers can exploit inherent IPFS characteristics to amplify phishing attacks.

### 4.1 Collected Content

**Content Type Distribution.** We first classify $37.2 \times 10^6$ downloaded files. Table 1 shows the most popular MIME types. IPFS is primarily used to host structured and binary files under the `application/*` MIME family (48.04%), web content HTML (28.98%), and images (22.04%). Since our focus is on phishing characterization, the following sections restrict the analysis to the `text/html` subset.

**HTML Content Themes.** We group HTML files via structural clustering and regex-based heuristics to identify dominant content themes. Wikipedia mirrors overwhelmingly dominate the dataset (93.4%) resulting from a popular community initiative.[4] Excluding Wikipedia reveals a more diverse ecosystem with Filecoin documentation (0.7%), JuiceBox project pages (0.2%), phishing sites (0.1%), and smaller clusters of miscellaneous pages (5.5%). While phishing pages account for only a small fraction overall, their relative weight becomes noticeable once Wikipedia is excluded (1.52% of the remaining pages), highlighting the phishing problem on IPFS. We notice an even higher ratio of phishing content when looking at pDNS traffic alone. From 3.3 M pDNS requests related to IPFS, 85.7 k (2.66%) were directed to known phishing resources.

**Table 1: Distribution of MIME types among collected files.**

| MIME Type | Count | Share |
|---|---|---|
| text/html | 10,386,280 | 28.98% |
| application/json | 9,997,761 | 27.90% |
| application/octet-stream | 4,104,575 | 11.45% |
| image/jpeg | 3,464,983 | 9.67% |
| image/png | 2,630,405 | 7.34% |
| application/pdf | 2,393,749 | 6.68% |
| image/webp | 1,190,809 | 3.32% |
| others | 1,669,483 | 4.66% |

**Phishing CIDs.** Over the observation period, our deployed pipeline identifies 10,489 phishing HTML pages (≈53 per day). Most detections come from our Simhash-based similarity index, which matches pages against known phishing templates. A smaller subset of 3.4 pages per day (6.4%) relies on heavy client-side obfuscation or script-generated markup. For these, we re-render the page to reconstruct the realized DOM before analysis. In parallel, the CTI module flags 11.9 pages per day as potential phishing candidates. Our manual verification confirms 34% as phishing. The remaining CTI positives are primarily benign developer templates (e.g., dashboard scaffolding) that resemble credential-taking pages.

### 4.2 IPFS Activity Trends

We investigate temporal activity trends on IPFS. We first consider all the CIDs in the network and then contrast the results with the phishing-related content.

**Overall Activity on IPFS.** We use a dataset of Bitswap traces from prior measurements collected from May 2020 through June 2025 [16].[5] Figure 2 shows the number of unique phishing CIDs and unique requests issued for them via Bitswap per month. We see major activity increase towards the end of 2023, particularly w.r.t. how many requests are performed per CID. The volume decreases sharply in 2024, but shows signs of increased activity in 2025 again.

**Phishing Activity on IPFS.** We then observe activity trends for all the phishing CIDs in our datasets. We combine CIDs detected by us with those detected by APWG, OpenPhish, Phishtank, and BadBits denylist. We limit the analysis to phishing CIDs for which we can infer a first-seen date (93.27 %).

Figure 3 shows how many phishing CIDs are first observed on Bitswap each month. The resulting time series mirrors the overall request trends from Figure 2. Activity peaks in 2023, drops in 2024, and then increases again from late 2024 into 2025. During this resurgence, we observe spikes of approximately 750 newly observed phishing CIDs per day, suggesting the launch of larger campaigns. The close coupling between overall and phishing trends suggests that IPFS-hosted phishing scales with broader network conditions.

[4] https://github.com/ipfs/distributed-wikipedia-mirror

[5] Their monitor experienced intermittent downtime in 2024

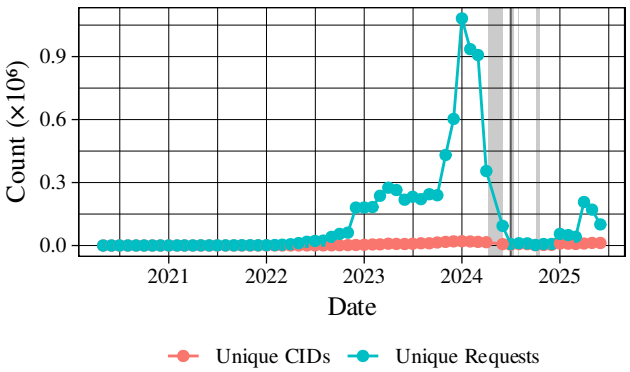



**Figure 2: Number of uniquely requested CIDs and unique requests for them per month via Bitswap. Monitor downtimes are highlighted.**

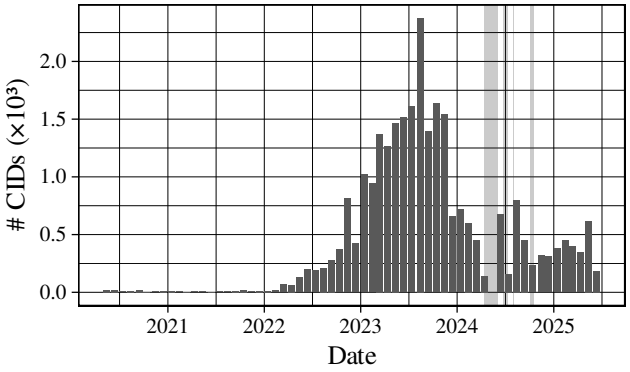



**Figure 3: Number of new phishing CIDs first seen via Bitswap per month. Monitor downtimes are highlighted.**

## 4.3 Phishing CID Provider Comparison

To evaluate the effectiveness and coverage of our detection system, we compare our collected CIDs with anti-phishing services and BadBits denylist, an IPFS-specific blocklist. As CIDs can have multiple formats [6], we translate all CIDs from all the sources to the canonical hashing scheme (i.e., v1) for comparison.

**Comparison with APS.** We first evaluate our dataset against CIDs extracted from phishing URLs in the OpenPhish Community feed [11], APWG [1], and PhishTank [12] (cf. Section 2.2). We collectively refer to them as anti-phishing services (APS). We use the data collected over the same observation period as our monitoring timespan and extract the corresponding CIDs from the URLs. As shown in Figure 4, there is a limited overlap between our detections and external feeds. While some high-profile campaigns appear across datasets, APS contain 4,224 CIDs that do not appear in our detections. Conversely, 10,153 (96.79%) of our CIDs are unique to our system.

The limited overlap reflects the differences in scope and methodology. Traditional feeds are general-purpose and driven by user submissions, web-centric crawling, and other reporting workflows.

[6] https://docs.ipfs.tech/concepts/content-addressing/

In contrast, our CID-first pipeline discovers content via pDNS and continuous IPFS monitoring. APS may thus capture phishing CID that does not surface in our pDNS/Bitswap vantage (e.g., different access paths or gateways). Our Bitswap monitoring also samples a small fraction of daily traffic, making our measurements a lower bound. At the same time, as we show later, IPFS phishing campaigns can generate a large number of unique CIDs and our methodology is more efficient in detecting them. This suggests that traditional feeds miss much IPFS-hosted phishing.

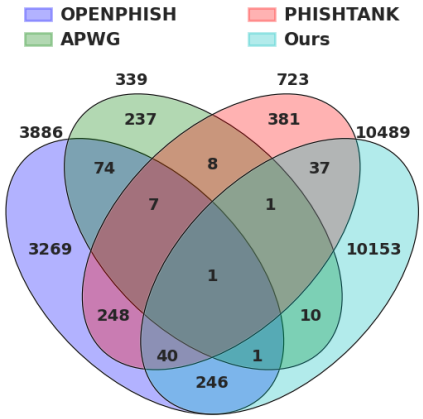



**Figure 4: CID overlap across different anti-phishing services and our list.**

**Comparison with IPFS-Specific Blocklists.** We compare detected CIDs with the IPFS-specific BadBits denylist curated by Protocol Labs. Figure 5 shows the three-way overlap. Notably, BadBits denylist adds no additional CIDs beyond the union of APS and our detections, suggesting it largely reflects indicators surfaced by other pipelines rather than expanding coverage. Our system alone contributes 4,381 unique phishing CIDs (42% of our detections) not present in other lists. Conversely, APS reports 2,292 phishing CIDs that we do not observe in our Bitswap monitoring, indicating that web-facing threat feeds and in-network measurements capture complementary slices of IPFS abuse (e.g., due to different vantage points and timing). Overall, our monitoring complements existing threat feeds, uncovering phishing content missed by both traditional providers and the IPFS-specific BadBits denylist service.

## 4.4 Phishing Campaigns

Using Simhash fingerprints, we cluster 10,489 phishing pages to infer campaigns. Here, a cluster corresponds to a set of at least two pages that have similar fingerprints and are therefore likely to originate from the same phishing kit or template. We find that 166 clusters account for 90% of pages, 69 clusters cover 80%, and just 11 clusters account for 50%, indicating that a small number of kits or templates dominate phishing activity.

Analyzing the top 69 clusters, we identify the main targeted brands: (1) Webmail / generic phishing: 68.9% (46 clusters) (2) Microsoft-related (LinkedIn, OneDrive, Office, SharePoint): 8.5% (16

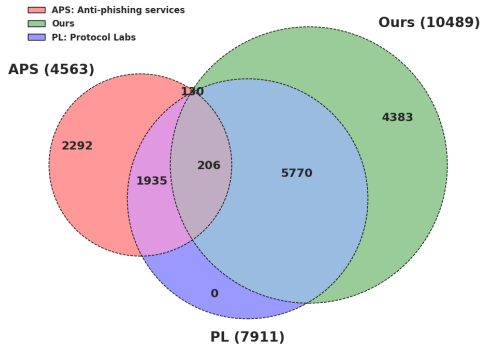



**Figure 5: CID Overlap across anti-phishing services, the BadBits denylist, and our list.**

clusters) (3) WeTransfer: 1.6% (3 clusters) (4) Adobe: 0.5% (2 clusters) (5) AT&T: 0.4% (1 cluster) (6) DHL: 0.5% (2 clusters)

Webmail and generic phishing are by far the most common. This is likely because: (1) attackers can easily swap logos or tweak branding to tailor the lure, as we observe in the wild; and (2) attackers can reuse generic input forms and webmail-style portals across many targets, since webmail is widely deployed.

We manually examine clusters to identify how phishing pages vary within the same campaign. We find that attackers often generate new instances through superficial modifications (e.g., comments, whitespace, or swapped logo URLs) or by changing small code fragments that determine the intended recipient, consistent with phishing-kit reuse across actors. In particular, for many clusters, attackers modify only HTML comments between versions, leaving all other content unchanged. We spot such modifications in the second- (1,459 instances) and the third-largest clusters (256 instances). These minor edits exploit IPFS CID generation. Any modification, no matter how small, yields a new CID. This enables attackers to multiply links at minimal cost while evading static denylist mechanisms.Appendix B provides additional details on these campaign variations.

## 4.5 Gateway Perspective

Next, we examine the role of IPFS gateways in exposing, accessing, and potentially mitigating phishing content.

**Gateway and CID Categories.** For visibility, we separate gateways and CIDs into groups. We classify gateways into: *Popular* (receiving high query volumes in pDNS), *Self-hosting* (publishing at least one provider record for one of our phishing CIDs), and *Other* (all remaining gateways). We also analyze three, non-mutually-exclusive CID sets: *APS* (our CIDs also reported by APS), *Badbits* (our CIDs also on BadBits denylist during our measurement window), and *Ours* (all our CIDs). When we report per-category rates for a gateway, the denominator is the gateway count of *online* CIDs (i.e., retrieved or blocked at least once throughout the measurement period) belonging to the class.

**Blocking Behavior.** Figure 6 shows that *Popular* gateways block a larger share of CIDs than the other groups, while *Self-hosting* gateways exhibit the lowest blocking fractions across all CID categories.

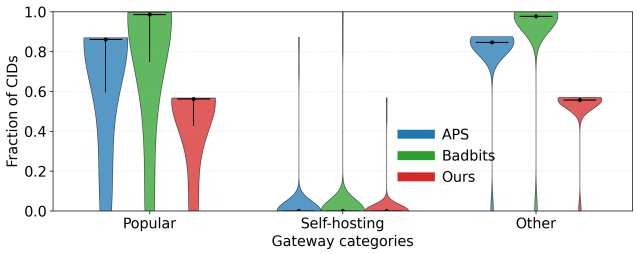



**Figure 6: Distribution of per-gateway blocking fractions across gateway and CID categories. Each violin shows, across gateways in a category, the fraction of that group's *online* CIDs (per class) that were blocked (returning HTTP 410/451).**

*Badbits* CIDs are, blocked more frequently than *APS* and *Ours* in both the *Popular* and *Other* groups. This is expected as many gateway implementations natively support BadBits denylist. During the measurement period, 196 gateways updated their blocking policies, initially allowing, but later blocking, a total of 569 CIDs.

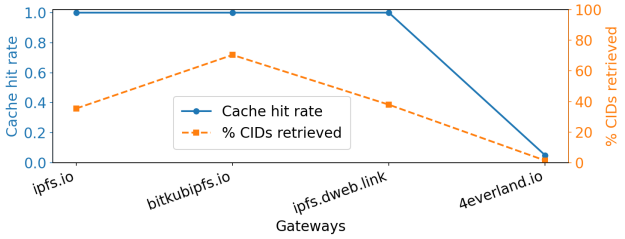



**Figure 7: Cloudflare cache hit rate (pre-fetch `CF-Cache-Status=HIT`, conditioned on verified retrieval) and average share of "Our" CIDs retrieved per run for Cloudflare-proxied gateways.**

**Phishing Traffic via Gateways.** Despite expecting attackers to use obscure gateways to evade detection, our pDNS data shows phishing is mainly served via popular gateways. Table 2 summarizes the distribution of unique phishing CIDs and total requests across our IPFS gateway list. Interestingly, many phishing CIDs

**Table 2: Distribution of unique phishing CIDs and requests across selected IPFS gateways.**

| Gateway | Unique CIDs | Total Requests |
|---|---|---|
| `ipfs.dweb.link` | 3,305 | 67,917 |
| `ipfs.io` | 529 | 5,592 |
| `ipfs.nftstorage.link` | 420 | 3,765 |
| `ipfs.w3s.link` | 409 | 7,791 |
| `ipfs.fleek.cool` | 52 | 362 |
| `astyanax.io` | 47 | 100 |
| `4everland.io` | 34 | 156 |
| `storry.tv` | 4 | 38 |
| `cloudflare-ipfs.com` | 1 | 10 |

were requested through multiple gateways, often with traffic fairly balanced among them. All the initially online CIDs remained retrievable through at least one gateway throughout the measurement period, highlighting the persistence and resilience of phishing content on IPFS. This suggests that some threat actors are leveraging IPFS gateways to maximize content availability and resilience.

**Cloudflare-Proxied Gateways.** We identify four Cloudflare-proxied gateways by the presence of Cloudflare headers (e.g., `CF-Cache-Status`) and compute cache metrics for the online CIDs that were successfully retrieved. These contain two official IPFS gateways operated by Protocol Labs (`ipfs.io`, `ipfs.dweb.link`).

As shown in Figure 7, the pre-fetch cache hit rate for official gateways is nearly 1.0. This indicates that the retrieved phishing CIDs were already present in the CDN cache. Averaged over runs, these official gateways served approximately 38% of our CIDs. We also detect a non-official (in the Others category) Cloudflare-proxied gateway (`bitkupipfs.io`) with a pre-fetch hit rate near 1.0 that hosts ≈78% of our phishing CIDs. The high cache availability adds an additional layer of resilience for phishing content, making takedown efforts more challenging as cached copies remain accessible even if the origin gateway restricts access.

**Gateways that Amplify Content Availability.** We next investigate whether gateways themselves also contribute to persistence of content (e.g., through caching at the gateway back-end IPFS node). We ran the following experiment: *(i)* we add a locally generated file with random content to our IPFS node running in a public cloud and fetch it through each gateway (round 1); *(ii)* next, we remove the file from our IPFS node so that it is no longer provided; and *(iii)* we repeat the file retrievals through all the gateways (round 2) when, in principle, it should have no providers.

Surprisingly, all the gateways that return the CID in round 1 also return it in round 2, despite our IPFS node being unable to provide any content. We confirm that only our local node is the only file provider before round 1, whereas *19 new providers* appear after round 1. This suggests that retrieving content via some gateways amplifies content availability: the back-end IPFS nodes of gateways not only cache content for the HTTP traffic but also announce themselves as providers on the IPFS network. This enables remaining gateways to fetch the file from the network later on.

By mapping the provider IPs returned before round 2 to known gateways, we can link 10 providers to gateways in our list. The remaining nine providers are likely from deployments where the gateway (front-end) and IPFS back-end reside on different hosts, and therefore, the provider IPs are different from the gateway IPs.

## 4.6 Provider Perspective

Finally, we examine phishing files availability from a provider perspective. We collect a snapshot of provider records for all phishing CIDs approximately every 24 hours.

**Provider Churn.** Figure 8 reports the daily count of unique providers and the inter-snapshot churn (computed from consecutive snapshots) Over the measurement period, the number of distinct peers advertising these CIDs varies substantially (minimum: 74, mean: 152, maximum: 340). To quantify these dynamics, we compute the daily churn between consecutive snapshots using the Jaccard formulation. Let $S_t$ be the set of provider peer IDs observed on day $t$. We report:

$$\text{churn}_t = 1 - \frac{|S_t \cap S_{t-1}|}{|S_t \cup S_{t-1}|},$$

i.e., one minus the Jaccard similarity between successive provider sets. We observe an average daily churn of ≈71% over the measurement period. This indicates that the set of providers changes substantially from day to day and that availability is supported by a rapidly shifting provider population rather than a stable core.

**Gateways vs. Providers.** We also cross-check gateway retrievals against the provider snapshots. For each day, we take the set of CIDs successfully retrieved from official IPFS gateways and compare it to the same–day provider snapshot to identify CIDs with zero observed providers. Figure 8 shows that, on average, ≈32% of the successfully retrieved CIDs had no providers in our snapshot that day. This further confirms that gateway accessibility can substantially exceed what is visible from provider records alone as because gateways serve content from their own caches/pinning infrastructure. However, this figure should be interpreted cautiously. The gateway tests and provider snapshots are taken on the same calendar date but not at the exact same time, so short–lived peers may appear or disappear between measurements.

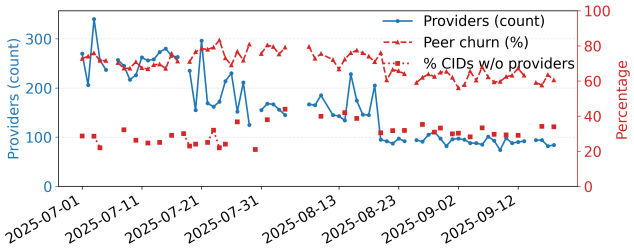



**Figure 8: Number of providers, provider churn, and the % of retrieved CIDs by official gateways with no providers.**

The figure illustrates that the provider landscape for phishing CIDs is highly dynamic, yet IPFS gateways and caching mechanisms ensure that content remains consistently available.

## 5 Discussion and Recommendations

In this section, we discuss the IPFS-based phishing ecosystem and provide mitigation recommendations.

**Points of Intervention.** On the Web, phishing typically involves just the victim and the hosting server. On IPFS, delivery is mediated by content providers, DHT resolvers/indexers, and HTTP gateways. Mitigation is possible at each layer, with varying effectiveness.

Blocking phishing at the individual provider level is impractical: nodes are decentralized, high-churn, making coordinated enforcement unrealistic. Instead, providers should be encouraged to use denylists to avoid unintentionally redistributing harmful material. Kubo supports denylist enforcement but disables it by default; we recommend enabling it by default and prompting users at install time to choose trusted blocklist sources. This action would curb the distribution of phishing and other harmful content, e.g., CSAM.

Filtering at the routing layer (i.e., DHT, Indexers) conflicts with the IPFS censorship-resistance ethos: routing exists to enable discovery, not to enforce policy, so adding filters undermines that role.

Gateways are the most practical choke points: they are stable, lower-churn, and accountable to abuse processes. Since phishing becomes a user risk mainly when served over HTTP, gateway blocking mitigates harm while preserving the IPFS censorship-resistant design—all content remains available via P2P but malicious content is no longer trivially reachable from the Web.

**Gateways as Content Providers.** Gateways exist to proxy between HTTP and IPFS; yet, our measurements show that they often become inadvertent providers after the original publisher vanishes. This happens via two paths: (i) any CID fetched through a gateway is downloaded and then advertised by the gateway itself, and (ii) gateways cache requested content, extending its availability beyond the publisher activity.

While both mechanisms improve availability for benign content, they hinder abuse mitigation. We recommend disabling gateway provider announcements by default: re-providing likely yields limited benefit, is ancillary to gateway functionality, and incurs storage/bandwidth costs. Caching remains operationally valuable. We instead recommend shortening TTLs and periodically validating provider records for cached CIDs, purging entries when no live, legitimate providers remain. Similarly, real-time updates to denylists should propagate to the caching layer, purging content that has become denylisted. These measures curb long-lived exposure of malicious content without materially degrading gateway performance.

**Traditional Web.** Defending users against abuse can no longer rely solely on IPFS-native or content-centric mitigation solutions. Instead, it is crucial to extend and adapt existing Web security countermeasures to recognize and address threats emerging from decentralized infrastructures. This approach is also more sustainable, as traditional security feeds and blocklists—such as GSB—benefit from established operational models and industry support. The phishing denylist provides CID-based indicators but can be readily extended to URL-style blocking compatible with traditional defenses. Achieving this, however, requires a comprehensive and continuously updated gateway inventory, which our monitoring framework supplies. Thus, we hope our work serves as a stepping stone toward the integration of decentralized threat intelligence into the broader Web security ecosystem.

**Challenges of Blocklists and Scalability.** Existing blocklists (e.g., the BadBits denylist) are useful baselines but insufficient. Two issues dominate: (i) limited coverage—many phishing CIDs in our dataset are absent, and (ii) low operator adoption—the BadBits denylist lacks annotations explaining *why* a CID is listed, preventing policy and jurisdiction checks. Furthermore, recent work has shown that the BadBits denylist largely deals with copyrighted material [44], which can overshadow blocklisting for other categories of harmful or illegal content (e.g., terrorist propaganda and CSAM)

We propose enriched blocklists with category labels (e.g., phishing) and lightweight evidence such as screenshots and detection labels. This enables selective enforcement (e.g., phishing-only), improves operator accountability, and facilitates regulatory mandates. Ephemeral gateways may persist, but they are amenable to traditional domain-based takedowns, shifting accountability from benign gateway operators to malicious actors.

## 6 Related Work

**Malicious Content and Moderation in Decentralized Systems.** Decentralized systems resist centralized control, complicating abuse mitigation. Prior work on BitTorrent, KaZaA, and tools like TorrentGuard [22, 26, 28] document the problem in P2P systems. Similar tensions appear in decentralized social networks, where moderation without recentralization remains difficult [18, 27, 42, 52].

**IPFS and its Security Landscape.** IPFS has been studied for design/performance, deployment, centralization trends, and attacks (e.g., eclipse) [16, 17, 23, 40, 47, 49]. Recent work characterizes illicit content and the limits of the BadBits denylist [44]. We build on this line by focusing specifically on phishing.

**Fraud and Abuse Detection in Blockchain Systems.** Adjacent research detects fraud on blockchains using ML and graph-based methods [37, 48, 50], showing the promise of data-driven detection in decentralized settings. We adapt similar principles to content-addressed storage.

**Gap in Existing Work.** Phishing is well-studied on the Web, but not on IPFS. To our knowledge, we present the first systematic, at-scale study of IPFS phishing campaigns, adapting web techniques to decentralized, content-addressed properties and examining how attackers exploit IPFS features.

## 7 Conclusion and Future Work

In this paper, we present a framework for monitoring and characterizing phishing on IPFS. While the IPFS resilient content distribution offers clear benefits, its architecture also enables abuse via web-facing gateways that serve phishing pages. Our measurements quantify the scale and persistence of this misuse at gateways, and indicate that gateway/CDN caching contributes to keeping phishing content accessible.

Looking ahead, the priority is broader and more representative visibility: expanding vantage points and pursuing opt-in collaboration with major gateways for real-time feeds would improve coverage of abusive activity. Beyond visibility, ecosystem levers matter: standardized, evidence-backed CID/URL denylists and simple gateway reputation metrics (e.g., responsiveness to reports, time-to-block) can encourage prompt, transparent moderation without compromising the IPFS decentralized design. Finally, because gateway and CDN caching can accidentally keep harmful content online longer, we should test adaptive rules, such as real-time denylist updates, to better balance performance and security.

## Acknowledgements

We thank the SIE Europe, APWG, PhishTank, OpenPhish for providing data access for this research. This work has been partially funded by KOR Labs and the European Union under Grant Agreement No. 101128042 (project ThreatChase) supported by the European Cybersecurity Competence Centre. It was also co-funded by the LOEWE initiative (Hessen, Germany) within the emergenCITY center [LOEWE/1/12/519/03/05.001(0016)/72].

## References


[1] Anti-phishing working group: Unifying the global response to cybercrime. Accessed: 2025-09-29.
[2] Audius. https://audius.org. Accessed: 2023-05-26.
[3] dClimate. https://www.dclimate.net. Accessed: 2023-05-26.
[4] Deece. https://github.com/navinkeizer/Deece. Accessed: 2023-05-26.
[5] Discussify. https://github.com/ipfs-shipyard/discussify-browser-extension. Accessed: 2023-05-26.
[6] DTube. https://d.tube. Accessed: 2024-01-30.
[7] Ethlance. https://github.com/district0x/ethlance. Access: 2023-22-01.
[8] Gala games. https://app.gala.games/games. Accessed: 2023-05-26.
[9] IPFS ecosystem directory. https://ecosystem.ipfs.tech/. Accessed: 2023-06-03.
[10] Matters News. https://matters.news. Accessed: 2023-05-26.
[11] Openphish. https://openphish.com/. Accessed: 2025-02-02.
[12] PhishTank. https://phishtank.org/. Accessed: 2025-02-02.
[13] Space. https://github.com/ipfs-shipyard/space. Accessed: 2023-05-30.
[14] Splinterlands. https://splinterlands.com/. Accessed: 2023-05-26.
[15] Balduf, L., Florian, M., and Scheuermann, B. Dude, where's my NFT: Distributed infrastructures for digital art. DICG '22, Association for Computing Machinery, p. 1–6.
[16] Balduf, L., Henningsen, S., Florian, M., Rust, S., and Scheuermann, B. Monitoring data requests in decentralized data storage systems: A case study of IPFS. In *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)* (2022), pp. 658–668.
[17] Balduf, L., Korczyński, M., Ascigil, O., Keizer, N. V., Pavlou, G., Scheuermann, B., and Król, M. The cloud strikes back: Investigating the decentralization of IPFS. In *Proceedings of the 2023 ACM on Internet Measurement Conference* (2023), pp. 391–405.
[18] Balduf, L., Sokoto, S., Ascigil, O., Tyson, G., Scheuermann, B., Korczyński, M., Castro, I., and Król, M. Looking at the blue skies of bluesky. In *Proceedings of the 2024 ACM on Internet Measurement Conference* (11 2024), IMC '24, Association for Computing Machinery, p. 76–91.
[19] Bayer, J., Benjamin, B. C., Maroofi, S., Wabeke, T., Hesselman, C., Duda, A., and Korczyński, M. Operational Domain Name Classification: From Automatic Ground Truth Generation to Adaptation to Missing Values. In *Passive and Active Measurement* (2023).
[20] Benet, J. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561* (2014).
[21] Christin, N., Weigend, A. S., and Chuang, J. Content availability, pollution and poisoning in file sharing peer-to-peer networks. In *Proceedings of the 6th ACM Conference on Electronic Commerce* (2005), Association for Computing Machinery, p. 68–77.
[22] Cuevas, R., Kryczka, M., González, R., Cuevas, A., and Azcorra, A. Torrentguard: Stopping scam and malware distribution in the bittorrent ecosystem. *Computer Networks 59* (2014), 77–90.
[23] Daniel, E., and Tschorsch, F. Passively measuring ipfs churn and network size. In *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW)* (2022), IEEE, pp. 60–65.
[24] Das, D., Bose, P., Ruaro, N., Kruegel, C., and Vigna, G. Understanding security issues in the nft ecosystem. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security* (New York, NY, USA, 2022), CCS '22, Association for Computing Machinery, p. 667–681.
[25] De la Rocha, A., Dias, D., and Psaras, Y. Accelerating content routing with bitswap: A multi-path file transfer protocol in ipfs and filecoin, 2021.
[26] Fetscherin, M. Movie piracy on peer-to-peer networks—the case of kazaa. *Telematics and Informatics 22*, 1-2 (2005), 57–70.
[27] Hassan, A. I., Raman, A., Castro, I., Zia, H. B., De Cristofaro, E., Sastry, N., and Tyson, G. Exploring content moderation in the decentralised web: The pleroma case. In *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies* (New York, NY, USA, 2021), CoNEXT '21, Association for Computing Machinery, p. 328–335.
[28] Kalafut, A., Acharya, A., and Gupta, M. A study of malware in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement* (New York, NY, USA, 2006), IMC '06, Association for Computing Machinery, p. 327–332.
[29] Liang, J., Naoumov, N., and Ross, K. W. The index poisoning attack in p2p file sharing systems. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications* (2006), pp. 1–12.
[30] Lin, Y., Liu, R., Divakaran, D. M., Ng, J. Y., Chan, Q. Z., Lu, Y., Si, Y., Zhang, F., and Dong, J. S. Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages. In *30th USENIX Security Symposium (USENIX Security 21)* (2021), pp. 3793–3810.
[31] Liu, R., Lin, Y., Yang, X., Ng, S. H., Divakaran, D. M., and Dong, J. S. Inferring phishing intention via webpage appearance and dynamics: A deep vision based approach. In *30th USENIX Security Symposium (USENIX Security 21)* (2021).
[32] Ltd., R. T. Temporal. https://temporal.cloud, 2020. Accessed: 2023-08-01.
[33] Maroofi, S., Korczyński, M., Hesselman, C., Ampeau, B., and Duda, A. COMAR: Classification of Compromised versus Maliciously Registered Domains. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)* (Los Alamitos, CA, USA, 2020), IEEE Computer Society, pp. 607–623.
[34] Maymounkov, P., and Mazieres, D. Kademlia: A peer-to-peer information system based on the XOR metric. In *International Workshop on Peer-to-Peer Systems* (2002), IPTPS, Springer.
[35] Odgerel, Z., Nosyk, Y., Bayer, J., Maroofi, S., Bedeschi, L., Duda, A., and Korczyński, M. Short Path to Phishing: Identifying Misused URL Shortening Services in the Wild . In *2025 APWG Symposium on Electronic Crime Research (eCrime)* (2025), IEEE Computer Society, pp. 1–13.
[36] Organization, A. Almonit. https://almonit.eth.link/. Accessed: 2023-06-02.
[37] Pathak, V., Uma Maheswari, B., and Geetha, S. Ensemble learning based social engineering fraud detection module for cryptocurrency transactions. In *International Conference on Mining Intelligence and Knowledge Exploration* (2023), Springer, pp. 301–311.
[38] Preston, I. Peergos. https://peergos.org. Accessed: 2023-07-01.
[39] Protocol Labs. Bad bits denylist. Accessed: 2024-02-06.
[40] Prünster, B., Marsalek, A., and Zefferer, T. Total eclipse of the heart–disrupting the {InterPlanetary} file system. In *31st USENIX Security Symposium (USENIX Security 22)* (2022), pp. 3735–3752.
[41] Rao, R. S., and Pais, A. R. An enhanced blacklist method to detect phishing websites. In *Information Systems Security* (Cham, 2017), R. K. Shyamasundar, V. Singh, and J. Vaidya, Eds., Springer International Publishing, pp. 323–333.
[42] Rozenshtein, A. Z. Moderating the fediverse: Content moderation on distributed social media. *J. Free Speech L. 3* (2023), 217.
[43] Silva, R. D., Nabeel, M., Elvitigala, C., Khalil, I., Yu, T., and Keppitiyagama, C. Compromised or Attacker-Owned: A Large Scale Classification and Study of Hosting Domains of Malicious URLs. In *30th USENIX Security Symposium (USENIX Security 21)* (Aug. 2021), USENIX Association, pp. 3721–3738.
[44] Sokoto, S., Balduf, L., Trautwein, D., Wei, Y., Tyson, G., Castro, I., Ascigil, O., Pavlou, G., Korczyński, M., Scheuermann, B., et al. Guardians of the galaxy: Content moderation in the {InterPlanetary} file system. In *33rd USENIX Security Symposium (USENIX Security 24)* (2024), pp. 1507–1524.
[45] Tajalizadehkhoob, S., van Goethem, T., Korczyński, M., Noroozian, A., Böhme, R., Moore, T., Joosen, W., and van Eeten, M. Herding Vulnerable Cats: A Statistical Approach to Disentangle Joint Responsibility for Web Security in Shared Hosting. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS* (2017), ACM, pp. 553–567.
[46] Technologies, B. Berty. https://berty.tech. Accessed: 2023-05-26.
[47] Trautwein, D., Raman, A., Tyson, G., Castro, I., Scott, W., Schubotz, M., Gipp, B., and Psaras, Y. Design and evaluation of ipfs: a storage layer for the decentralized web. In *Proceedings of the ACM SIGCOMM 2022 Conference* (2022), pp. 739–752.
[48] Umer, Q., Li, J.-W., Ashraf, M. R., Bashir, R. N., and Ghous, H. Ensemble deep learning-based prediction of fraudulent cryptocurrency transactions. *IEEE Access 11* (2023), 95213–95224.
[49] Wei, Y., Trautwein, D., Psaras, Y., Castro, I., Scott, W., Raman, A., and Tyson, G. The eternal tussle: Exploring the role of centralization in ipfs. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)* (April 2024), USENIX Association.
[50] Yan, C., Han, X., Zhu, Y., Du, D., Lu, Z., and Liu, Y. Phishing behavior detection on different blockchains via adversarial domain adaptation. *Cybersecurity 7*, 1 (2024), 45.
[51] ZorrillosDev. Watchit. https://watchit.movie/, 2022. Accessed: 2023-05-23.
[52] Zuo, W., Raman, A., Mondragón, R. J., and Tyson, G. Set in stone: Analysis of an immutable web3 social media platform. In *Proceedings of the ACM Web Conference 2023* (2023), pp. 1865–1874.


# A Phishing Detection Evaluation

## A.1 Dataset and Ground Truth

To evaluate alternative phishing detection paradigms, we constructed a labeled corpus of 5,000 HTML pages fetched from IPFS between September 2024 and March 2025. This set contains 2,940 benign pages and 2,060 phishing pages. To mitigate temporal and source bias, benign pages were sampled weekly and capped per large clusters. Non-major sources were manually verified. Candidate phishing pages were identified using static cues (e.g., presence of credential forms, password inputs, or substantial obfuscation) and subsequently confirmed by manual inspection. To avoid domination by popular phishing kits, we restricted the dataset to at most

100 pages per campaign (see Section 4.4). Pages that failed to render within 3 s in a standardized headless browser were excluded; 4,260/5,000 pages (85.2%) remained.

## A.2 Benchmarking Protocol

We evaluated three representative detection paradigms: (i) Phishpedia [30]: visual logo matching on rendered screenshots, (ii) PhishIntention [31]: logo detection combined with credential-taking intention (CTI), (iii) Simhash [41]: DOM-structure fingerprinting via locality-sensitive hashing applied to parsed HTML.

We used the released pre-trained models and default logo/brand repositories for Phishpedia and PhishIntention. In 6-fold cross-validation, we build the Simhash index from the training data and test it on the held-out fold. We report accuracy, precision, recall, F1 score, and per-page execution time measured on identical hardware. We ran all experiments on an Intel Core i7-13700H with 32 GB RAM. Times are classifier-only to match our pipeline (no screenshot rendering).

**Table 3: Classifier-only performance on the 5 k-page corpus.**

| Method | Acc | Prec | Rec | F1 | Time/page |
|---|---|---|---|---|---|
| Simhash | 0.949 | 1.000 | 0.866 | 0.928 | <1 ms |
| Phishpedia | 0.658 | 1.000 | 0.169 | 0.289 | ~1.5 s |
| PhishIntention | 0.643 | 1.000 | 0.134 | 0.236 | ~1.6 s |

## A.3 Classification Results

Table 3 presents the results across all three paradigms. All methods had zero false positives on $n = 2{,}940$ negatives; by the rule of three, the one-sided 95% upper bound on the true FPR is $3/n = 0.102\%$.

The Simhash-based approach achieved the best overall trade-off, with a recall of 0.866 (95% Wilson CI: 0.850–0.881) and very low computational cost (< 1 ms per page). By comparison, Phishpedia and PhishIntention exhibited substantially lower recalls of 0.169 (95% CI: 0.154–0.185) and 0.134 (95% CI: 0.120–0.149), respectively, despite perfect precision. Their misses arise primarily in brand-agnostic credential capture or pages without logos, where screenshot–logo matching provides a limited signal.

## A.4 Operational Detection Pipeline

Guided by the comparative findings, the deployment uses Simhash for its accuracy–efficiency trade-off. The operational pipeline consists of two main paths:

(1) *Fingerprint path (indexed content):* Parsed HTML is hashed (Simhash) and queried against the existing fingerprint index. If a match is found, the page is categorized as phishing and inherits the matched cluster (campaign). Otherwise, we fall back to the cold-start path.
(2) *Cold-start CTI path (unseen pages):* If no index match is found, we detect credential-taking intention (e.g., credential forms, password fields). When CTI is positive, a sample is manually verified and its fingerprint is inserted into the index.

When we detect substantial client-side generation or obfuscation, we render the page and reapply CTI checks and Simhash fingerprinting to the realized DOM. The overall flow is shown in Figure 9.

## A.5 Summary

We adopt Simhash as the default detector due to its high recall and low runtime, with the caveat that our pipeline uses a manual verification step for cold-start cases before adding fingerprints to the index, which introduces operational cost but keeps compute overhead low. Visual detectors offered high precision, yet were sensitive to logo presence and substantially more expensive to run. Our results should thus be read as a design trade-off tailored to IPFS-scale measurements.

# B Campaign Variations in IPFS

Below, we describe the two most common types of content changes observed within clusters in our dataset (cf. Section 4.4).

## B.1 Example A – Comment-only Edits

**Description.** Attackers can make minor changes to HTML comments while keeping the rest of the page identical, generating new phishing content with their own CIDs. One example cluster contains 1,459 near-identical pages where only a random commented string changes.

**Representative Diff.** In the following, we show two versions of the same cluster and how they appear in the wild.

```
--- version1.html
+++ version2.html
@@ -1,6 +1,6 @@
 <!DOCTYPE html>
 <html lang="en">
-<!-- "7ucqpck1v3353s" -->
+<!-- "fjpqcv1f0p8k54" -->
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
```

There are no other differences in the two versions.

## B.2 Example B — Javascript Changes

**Description.** Attackers can modify JavaScript code. While some edits introduce distinct behaviors, others are largely cosmetic. One example cluster contains 228 phishing pages. The snippets below are extracted from an obfuscated JavaScript payload and illustrate a minimal but functionally significant change between variants. In the original variant, the AJAX request targets:

hxxps://agenziasiforma.it/TVWDXij/dns/send.php:

```
ajax=new XMLHttpRequest(),
ajax[_0x1d71d4(0xce)](_0x1d71d4(0x103),
'hxxps://agenziasiforma.it/TVWDXij/dns/
send.php')
```

In the modified variant, the endpoint is changed to:

hxxps://medyseg.com/bt/send.php:

```
ajax=new XMLHttpRequest(),
ajax[_0x1d71d4(0xce)](_0x1d71d4(0x103),
'hxxps://medyseg.com/bt/send.php')
```

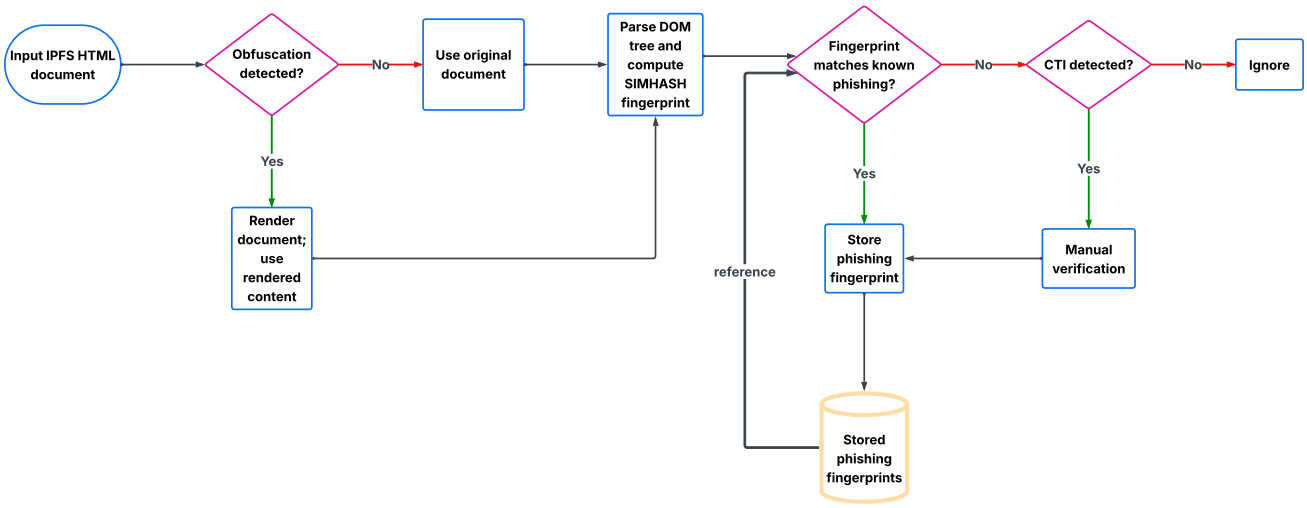



**Figure 9: Phishing detection workflow: Simhash matches known pages; unmatched pages undergo credential-taking intention (CTI) analysis; obfuscated HTML is rendered before analysis; confirmed cases are fingerprinted.**

## C Ethical Considerations

Our study involves monitoring and analyzing IPFS network activity to detect and cluster phishing content. Below, we outline ethical considerations for this work.

(1) **Network Impact.** To monitor IPFS content, we connect to and observe network activity. We are aware that our activities—such as querying nodes and downloading suspected phishing content—introduce some load on the IPFS network. To minimize any potential disruption, we rate-limit our queries and configure our IPFS node not to re-serve any downloaded content to other peers. We assess the resulting overhead as minimal and proportionate to our research objectives (i.e., measuring and characterizing phishing on IPFS).

(2) **Data Privacy.** While content on IPFS is publicly accessible to anyone by design, we still apply careful ethical standards in handling it. We focus solely on detecting and analyzing phishing-related content, and we do not collect or process any personally identifiable information. All content is downloaded solely for the purpose of phishing classification. We do not inspect content manually unless it is necessary to confirm a phishing instance.