Guardians of the Galaxy: Content Moderation in the InterPlanetary File System

Saidu Sokoto, *City, University of London;* Leonhard Balduf, *TU Darmstadt;* Dennis Trautwein, *University of Göttingen;* Yiluo Wei and Gareth Tyson, *Hong Kong Univ. of Science & Technology (GZ);* Ignacio Castro, *Queen Mary, University of London;* Onur Ascigil, *Lancaster University;* George Pavlou, *University College London;* Maciej Korczyński, *Univ. Grenoble Alpes;* Björn Scheuermann, *TU Darmstadt;* Michał Król, *City, University of London*

https://www.usenix.org/conference/usenixsecurity24/presentation/sokoto

This paper is included in the Proceedings of the 33rd USENIX Security Symposium.

August 14–16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

Open access to the Proceedings of the 33rd USENIX Security Symposium is sponsored by USENIX.

Guardians of the Galaxy: Content Moderation in the InterPlanetary File System

Saidu Sokoto*
City, University of London

Leonhard Balduf*
TU Darmstadt

Dennis Trautwein*
University of Göttingen

Yiluo Wei
Hong Kong Univ. of Science & Technology (GZ)

Gareth Tyson
Hong Kong Univ. of Science & Technology (GZ)

Ignacio Castro
Queen Mary, University of London

Onur Ascigil
Lancaster University

George Pavlou
University College London

Maciej Korczyński
Univ. Grenoble Alpes

Björn Scheuermann
TU Darmstadt

Michał Król
City, University of London

Abstract

The InterPlanetary File System (IPFS) is one of the largest platforms in the growing "Decentralized Web". The increasing popularity of IPFS has attracted large volumes of users and content. Unfortunately, some of this content could be considered "problematic". Content moderation is always hard. With a completely decentralized infrastructure and administration, content moderation in IPFS is even more difficult. In this paper, we examine this challenge. We identify, characterize, and measure the presence of problematic content in IPFS (*e.g.* subject to takedown notices). Our analysis covers 368,762 files. We analyze the complete content moderation process including how these files are flagged, who hosts and retrieves them. We also measure the efficacy of the process. We analyze content submitted to denylist, showing that notable volumes of problematic content are served, and the lack of a centralized approach facilitates its spread. While we identify fast reactions to takedown requests, we also test the resilience of multiple gateways and show that existing means to filter problematic content can be circumvented. We end by proposing improvements to content moderation that result in 227% increase in the detection of phishing content and reduce the average time to filter such content by 43%.

1 Introduction

The Internet Planetary File System (IPFS) is one of the largest platforms in the Decentralized Web, with $\approx 30,000$ [74] online IPFS nodes, spread across 2,700 Autonomous Systems (ASes), and 152 countries [72]. IPFS supports a whole ecosystem of decentralized applications [9], including social networking [5, 10], data storage [15, 60, 64], content search [12, 62], messaging [70], streaming [6, 13, 79], gaming [8, 16], and e-commerce [7, 11]. IPFS is also widely used as external storage for blockchain-based applications such as NFT platforms [30, 38]. Growing support to facilitate IPFS access (*e.g.* Cloudflare's HTTP gateways, and integration by browsers like Opera and Brave) has further propelled IPFS usage.

However, the popularity of IPFS comes with challenges. Most notably, there has been growing media attention surrounding the need for content moderation in IPFS [14, 19, 20]. This is not a new problem, and many large platforms (*e.g.* Facebook) have long struggled with the complexities of moderation, such as the removal of Child Sexual Abuse Material (CSAM). In some ways, moderation in IPFS is easier, because its addressing scheme allows each content object to be uniquely identified agnostic to where it is hosted (unlike URLs, which depend on the hosting domain). In other ways, moderation in IPFS is harder, as there is no single point of control to make and enforce moderation decisions. The legal problems faced by earlier peer-to-peer services, such as BitTorrent [34, 35, 41] and Kazaa [42], highlight the importance of this problem. Consequently, various IPFS stakeholders implemented content moderation methods.

The content moderation activities in the IPFS ecosystem are currently led by *Protocol Labs*, the largest single contributor to the IPFS open-source effort [72]. In 2021, Protocol Labs started to compile its *badbits list* [50], listing content deemed as "problematic" after receiving a takedown request. Protocol Labs uses this denylist to filter flagged content from the IPFS gateways it manages. However, in a decentralized system, this is just a partial solution: there are many gateways hosted by different operators that do not necessarily share the same denylist. Thus, Protocol Labs makes the badbits list publicly available, so others can use it if they wish to filter certain content. However, to date, we know little about this process.

In this paper, we conduct the first in-depth analysis of problematic content within the IPFS network and the efforts made to moderate it. We provide a nuanced understanding of the content landscape and offer practical insights for reinforcing content moderation mechanisms. To achieve this, we collect a range of datasets (Section 3), covering (*i*) the content takedown requests received by Protocol Labs, alongside the final badbits list they compile; we extend it with datasets from

*These authors contributed equally to this work.

traditional Web2 providers, creating a more comprehensive *denylist* of problematic content; (*ii*) the presence of such content within the IPFS network, and its related infrastructure; (*iii*) the traffic related to such content; and (*iv*) the implementation of content filtering at various IPFS locations.

We start by studying the content moderation process for the `ipfs.io` gateway, from when content is reported to the actions taken and their impact (Section 4). We show that, while Protocol Labs usually acts upon complaints within 24h, problematic content can be present in the network for years before being reported. This leads us to characterize the content listed in the denylist. We investigate the breakdown of 368,762 files flagged for takedown reviews and find a large fraction of copyright-protected content ($\approx 87.97\%$ of files), as well as phishing campaigns ($\approx 5.81\%$). We also observe the presence of terrorist content ($\approx 0.06\%$), and potential CSAM ($< 0.01\%$). Importantly, to minimize risk and to report offending content, we have worked closely alongside the Internet Watch Foundation (IWF) [43].

Next, we identify which providers host the denylist content (Section 5) and which consumers retrieve it (Section 6). We find a substantial skew, with over 60% of unique denylist content items hosted by just two peers. This, however, is misleading as we show that content is often replicated at multiple locations. This replication creates challenges for blocking such material. This inspires us to measure the success of the moderation process. We observe limited filtering within the core IPFS network. Thus, we measure what content is blocked by gateways (Section 7). We find that gateway operators have different levels of blocking compliance. For example, out of all the gateways, only the ones operated by Protocol Labs filter the entire badbits list. The other public gateways, including those operated by large CDNs, filter less than 20% of the content. Critically, we also discover that some gateways are vulnerable to content filtering circumvention. For instance, we can circumvent the filtering by using different hash-based naming functions, even if they follow the denylist.

Finally, in Section 8, we discuss potential improvements to the existing content-moderation approaches. We propose strategies that aim to enhance the efficacy of content moderation within the IPFS network. We show that a tighter collaboration between Protocol Labs and traditional content filtering services can increase the ratio of detected phishing content by 227%. Our approach reduces the average time between the appearance of deceptive websites and their filtering by 43% and with minimal implementation effort. Our work fills a missing gap in the growing number of studies analyzing IPFS performance [72], security [63, 67], and resilience [32, 37]. We hope that it is useful for both developers and operators of IPFS infrastructure.

2 Background

The InterPlanetary File System (IPFS) is a set of protocols that facilitate decentralized content-addressable media object storage and retrieval. IPFS is an open-source effort where Protocol Labs is the largest contributor [72]. At its core, IPFS is a content-based addressing scheme that uses Content Identifiers (CIDs). CIDs are hash-based, immutable, self-certifying names that decouple the name of the content from its storage location. Accordingly, any peer can serve the content requested via a CID, and the receiver can verify that the data served matches the identifier requested via hashing. A distributed index maintains records of which providers host a given CID, allowing consumers to discover where to request content from without needing a central index. Here, we provide a brief overview of key parts; see [72] for full details.

DHT. IPFS uses a Kademlia-based [61] Distributed Hash Table (DHT) to index which node hosts data for a certain CID. The DHT indexes *provider records*, which maintain a mapping between a CID and the physical node(s) that can serve it. Peers then use this DHT to publish their provider records, so consumers can discover where to download the content from.

InterPlanetary Network Indexers. Involved parties in the IPFS ecosystem also operate a small set of InterPlanetary Network Indexers (IPNIs or "Indexers") to accelerate its performance. Indexers are high-performance keystore servers that index provider records. Providers can optionally publish their provider records to both the DHT and the Indexers. This can substantially increase performance, as the Indexers offer much faster lookup times for clients. Notably, data will stay in the Indexer unless actively removed by the provider, which will become relevant later. We emphasize that Indexers are intended to complement the DHT — the DHT remains the ultimate ground truth index of IPFS content.

Bitswap. To exchange content, IPFS relies on Bitswap [39], a simple protocol used to retrieve blocks of data. Bitswap is used by nodes to both retrieve and send data. The protocol allows one to ask a peer (using a WANT message) whether it possesses the content for a CID. The same protocol is then used to transfer the data.

Content Publication & Retrieval. When a provider wishes to publish a content item, it first computes its hash-based CID. It then publishes a *provider record* that contains the CID and its own network information. The record, by default, is pushed to the DHT but can also be pushed to one or more Indexers. Following this process, consumers in the network can retrieve the object. For this, a node first performs a 1-hop optimistic broadcast via Bitswap to all its connected neighbors looking for the target CID (typically a few hundred random peers). Searching via Bitswap is fast, but does not provide reliable content resolution, in particular for less popular content. Thus, if this does not yield any results, the node queries the DHT

or Indexers for the provider record. Once it has retrieved the provider record, it can then connect to an available host and retrieve the object. The download is performed using Bitswap.

HTTP Gateways. Accessing IPFS via the DHT and Bitswap comes with the substantial overhead of installing an IPFS node implementation (*e.g.* kubo [4]). To minimize the barriers to uptake, IPFS also supports access via *gateways*, which can translate Hypertext Transfer Protocol (HTTP) GET requests to content retrievals in IPFS. Gateways enable IPFS-agnostic users to seamlessly access IPFS content. When a gateway receives an HTTP GET for a CID, it launches a query into the IPFS network and retrieves the content on behalf of the web client. The gateway then returns the content to the client over the original HTTP channel. Gateways (built upon Nginx) also enable HTTP caching to accelerate performance and minimize load on the IPFS network. Protocol Labs maintains a list of public gateways [51], some of which are operated by Protocol Labs (`dweb.link`) and other large operators like Cloudflare (`cloudflare-ipfs.com`).

Protocol Labs Badbits List. Moderating content (*e.g.* filtering) in IPFS is inherently challenging, due to its decentralized nature. The most obvious location where such moderation could take place is at gateways or Indexers, as these are points of centralization. With the above challenge in mind, in 2021, Protocol Labs created a public list of content items that they deemed "harmful" for various reasons (*e.g.* copyright violation, phishing) and, therefore, should not be accessible. This list is referred to as the *badbits* list [50]. The badbits list is centrally maintained by Protocol Labs and built primarily on takedown requests received via email at `abuse@ipfs.tech` [55]. Any entity or person can issue a takedown request. This request is reviewed by a content moderation team and approved requests result in the content being added to their *badbits* list.

To date, the list contains over 410,000 entries, with each entry being a hex-encoded result of applying SHA256 to the base32-encoded CIDv1. This hashing of the CID allows parties to check whether a given CID is part of the *badbits* list without revealing the CID itself. Note, no additional metadata is included in the list.

Takedown Email Notices. We got access to the takedown request emails received via `abuse@ipfs.tech` between 2023-10-22 and 2024-01-11 and analyzed them according to Appendix A.2. This allows us to understand the nature of takedown requests and how they are acted upon. This was done with a team of former researchers at Protocol Labs, who received permission to share information with us for research purposes. This data was never transferred or stored outside of Protocol Labs. It was used only for the purposes of this study, and we never had direct access to the mailbox. Further, we do not review information about the email sender (Section 9).

3 Methodology

In this section, we explain the methodology and the datasets employed in this work. Figure 1 provides an overview of our data collection architecture.

3.1 Content Collection

We use the Protocol Labs *badbits* list [53] (see Figure 1 ①) as our starting point. We have gathered a time series of list versions since its creation in 2021. Because each entry on the badbits list is hex-encoded hash of the CID (see Section 2), inferring the CID directly is not possible. To overcome this, it is necessary to get a large set of valid CIDs, calculate their SHA256 hashes, and then match them against the badbits list. For this, we collect all candidate CIDs from Bitswap and DHT logs (see next).

Bitswap Logs. We collect Bitswap traces from mid-2021 to January 2024 [31] (see Figure 1 ②) using a modified IPFS node implementation with unlimited connections with peers. Our crawler logs all incoming 1-hop Bitswap broadcast traffic (*i.e.* each searching for a CID) to disk. From this, we observe $\approx$ 300B requests, covering $\approx$ 1B unique CIDs.

DHT Logs. We also set up a modified version of a DHT server [58] to collect the IPFS DHT traffic. We set up 20 virtual peer IDs and write all the incoming DHT requests to disk. We log the timestamp, the sender's peer ID and network address, the type of request, and the target key which could be a peer ID or CID depending on the message type. We also log the proxy DHT server if the original sender uses NAT traversal mechanisms (Section 2). Apart from collecting the traffic, our software acts as a regular DHT server following the IPFS specification. We collect traffic from September 2022 to January 2024 (see Figure 1 ③). From this, we observe 1.3B requests, covering 120M CIDs.

Badbits CIDs. Using the above two data sources, we extract all CIDs from the logs (see Figure 1 ④). We then convert all CIDs into their hash (SHA256). Using these hashes, we reverse map the entries on the badbits list to their associated CID. We recover 411,522 ($\approx$ 99.98%) of the CIDs in the badbits list. We refer to these as *badbits CIDs*.

Denylist CIDs. To measure the spread of a wider body of problematic content, we supplement the badbits list, using other lists. Specifically, we take information from Web2 Anti-Phishing Services (APS) (see Figure 1 ⑤, in ThreatChase Intelligence Platform [36]). This platform compiles lists of phishing URLs by combining data from OpenPhish [17], PhishTank [18], URLhaus [21], and the AntiPhishing Working Group (APWG) [1]. From these phishing URLs, we extract any CIDs present using regular expressions.[1] We identify 16,128 CIDs associated with phishing attacks (see Figure 1,

[1] From manual inspection, we see that all of these CIDs are embedded in URLs that access a particular gateway.

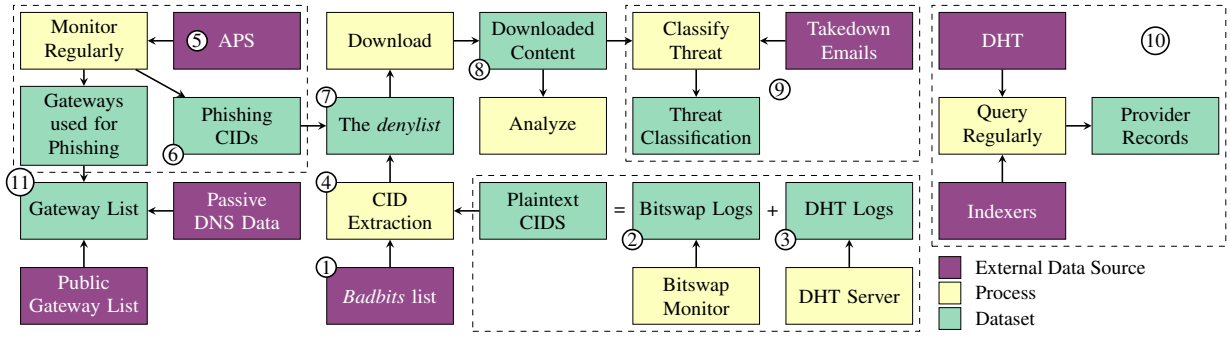

Figure 1: Overview of data collection architecture.

⑥). Our combined dataset, covering both badbits and phishing (*i.e.* Web2-moderated) content, contains 417,912 CIDs. We refer to this combined dataset as the *denylist* (see Figure 1 ⑦).

Content Retrieval. We then attempt to download all the content on the denylist except those reported as CSAM. We configure our IPFS node to not provide the downloaded content to other peers in the IPFS network and delete the content immediately after classifying it (see Figure 1 ⑧, Section 3.2). We successfully download 368,762 content identifiers (CIDs) (≈ 86.60% of all CIDs) in the denylist.

3.2 Content Classification

To gain a more nuanced understanding of the badbits content, we classify the content in the list.

MIME Types. We classify the content based on the MIME type into the following categories: *documents* (*e.g.* pdf, epub), *text* (*e.g.* plaintex, json), *html*, *binary* and *multimedia*.

Text Description. We also generate a text-based description of the material, in cases of text and image content. For documents, (*e.g.* PDFs, epub), we use the open-source GPT Mistral model [47]. This model can extract the language and field/-subject of any given file. For HTML files classified as phishing, we further obtain names or categories of the imitated target website using the *target* field in APS. We submit the remaining unclassified files to Google Bard Large Language Models (LLM) [45] asking it to indicate the target domain. We manually verify the domain detection on 100 files with 100% accuracy.

For images, we use BLIP2 pre-modal pre-training method [57] that combines image encoders and LLMs. We verify our BLIP2 classification on a small sample of 1,000 files manually classified by ProtocolLab/APS with 98% accuracy. We also calculate SHA256 hashes for all the downloaded content and submit them to the IWF [43] for a check against their CSAM database.

From the 368,762 successfully downloaded denylist files, 265,287 (≈ 71.94%) are PDFs, 25,628 (≈ 6.95%) are ePubs, 12,574 (≈ 3.41%) are DjVu files (a type of scanned document), and 7,412 (≈ 2.01%) are HTML files.

Threat Types. We classify all CIDs in the denylist taken from APS as phishing (as those services exclusively list phishing content). To classify the threat type of the remaining content, we first use information from the takedown request emails. We extract the takedown reason from these emails (see Appendix A.2), and classify CIDs into their corresponding category (see Appendix A.2 and A.3): Digital Millennium Copyright Act (DMCA), Phishing, CSAM, Animal Abuse Material (AAM), and Terrorist content. This allows us to label 12,170 of the badbits CIDs.

Partial data about threat types is also hosted in GitHub, containing information about the reason for the inclusion of certain CIDs in the badbits list. We, therefore, use information from GitHub from 2021-02-17 to 2024-01-12, accessed by the same former Protocol Labs researchers (with permission). This enables the classification of 154,465 additional CIDs added to the badbits list before 2023-10-22 (*i.e.* before the email logs are available).

The above data sources allow us to label 45% (166k) of the downloaded files with their threat type. To cover the remaining, we submit them to VirusTotal.[2] This successfully classifies 6,996 (1.9%) files. For the 195k (53%) files that remain unclassified, we employ a local instance of LLAMA2 LLM [71], as detailed in Appendix A.4. We feed it with examples of classified content (classification done by Protocol Labs, VirusTotal, and APS) including the content MIME type and the text description. We then feed it the remaining samples and ask it to perform automatic classification. We classify these files into threat types consistent with those in the badbits list (see Figure 1 ⑨): *DMCA* (*i.e.* copyright-protected content), *phishing*, *malware*, *terrorist content* and *explicit content* (*i.e.* including CSAM and AAM). This gives us labels for 88% of files in the denylist.

3.3 Content Hosting and Consumption

We next gather data on the nodes hosting content in the denylist (*i.e.* providers) and the users requesting it (*i.e.* con-

[2] https://www.virustotal.com

sumers).

IPFS Hosting. We twice-daily collect all provider records from the DHT for each CID in the denylist (see Figure 1 (10)). We do this from 2023-11-01 to 2024-01-13, allowing us to track who hosts the CIDs over time. We crawl the IPFS network to obtain the list of all peer IDs in the DHT and map each CID to the peers responsible for hosting the provider records for that CID, using the techniques from [72]. Then, we directly contact the peers with requests for the provider records of the CIDs. This technique avoids costly DHT queries and limits the impact of the study on the network.

Every three days, we also query the `cid.contact` indexer for provider records of all *badbits* CIDs [54]. We choose this interval to comply with the indexer's policy and its rate limits. During the period of the study, the queried indexer was the only one operational. This data augments the aforementioned DHT data of providers hosting CIDs on the *badbits* list.

Gateway Hosting. We also identify content available via IPFS HTTP gateways. We start with the list of public gateways maintained by Protocol Labs [51] (see Figure 1 (11)). We then expand it with a manual set of additional popular public gateways, and finally add any gateways identified in the URL feeds from the ThreatChase Intelligence Platform.[3] Finally, we add any gateways present in a passive DNS dataset obtained from the Farsight SIE passive DNS channel [40]. This dataset covers all requests made from mid-November 2023 to February 2024. Specifically, from this dataset, we extract any CIDs embedded within the Fully-qualified Domain Name (FQDN). Note that clients can request content from a gateway by embedding the CID as a subdomain [22].

We then implement a gateway checker that verifies the accessibility of all the denylist CIDs from every gateway on the *gateway list*. Our software probed the gateways from 2023-12-27 to 2024-01-31 every three days to avoid rate limits (see Section 7.1).

Consumers. We finally gather data to estimate the demand for each item on the denylist. For this, we extract all the requests for CIDs on the denylist present in the passive DNS, Bitswap and DHT logs. Furthermore, we log the date when each CID is first requested. We observe 7M, 420M, and 63M requests via the DNS, Bitswap and DHT data, respectively. These cover 417,912 unique CIDs.

4 A Badbits Denylist Primer

4.1 Overview of Badbits Compilation

Takedown Request Emails. The takedown email data provides insight into the types of content concerns that third parties flag within IPFS. Figure 2 shows the number of takedown request emails and CIDs received, spanning from 2023-10-22 until 2024-01-11. We break these down into our takedown classification and separately plot the numbers included on the badbits list vs. those discarded by the moderation team. Note, emails, and CIDs marked as "Not Included" could either be discarded items or simply not yet acted upon by the review team.

We observe that over 60% of all email takedown requests are accepted for inclusion in the badbits list. This initially suggests that the quality of takedown notices is relatively high, with fewer examples of miscellaneous submissions. However, Figure 2 also plots the equivalent results based on the number of CIDs (because each email can contain multiple CIDs). In stark contrast, we find that the majority of CID takedown requests are actually *rejected*.

To understand why, we inspect the takedown classification categories, where we see notably different patterns. Closer inspection reveals that these patterns are driven by the tendency for DMCA emails to include up to eight times more CIDs than phishing takedown request emails [19]. Mirroring prior observations [46] in traditional web hosting, it seems that such agencies 'optimistically' send many requests, even though they are not verified. We argue that this strategy creates a significant load, mandating the need for automated tools. Despite the dominance of DMCAs, a minority of emails are also classified as CSAM, terrorist content, or other reasons ($< 1.3\%$ of all emails and $< 1\%$ of all CIDs), confirming that moderation requirements are diverse.

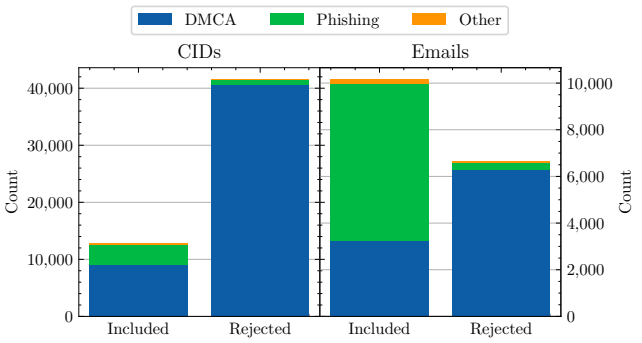

Figure 2: Received takedown emails by category from 2023-10-22 until 2024-01-11. Note that one email may contain multiple CIDs. "Other" includes categories like CSAM, terrorist content, unblock requests, or unspecified reasons.

List Evolution. Figure 3 next shows the growth of the badbits list over time. We plot the absolute number of CIDs received alongside the types. This graph shows that, throughout the existence of the list, the majority of CIDs included are DMCA-related. We also note a recent growth of the list. Almost 90% of the badbits CIDs are added in 2023 alone. We conjecture that this is due to the increasing popularity of the system, which has attracted more users and providers. More

[3] We do this by extracting the domains from URLs that contain a CID and match the correct regular expression.

importantly, it also has attracted the attention of numerous copyright-enforcement agencies [19].

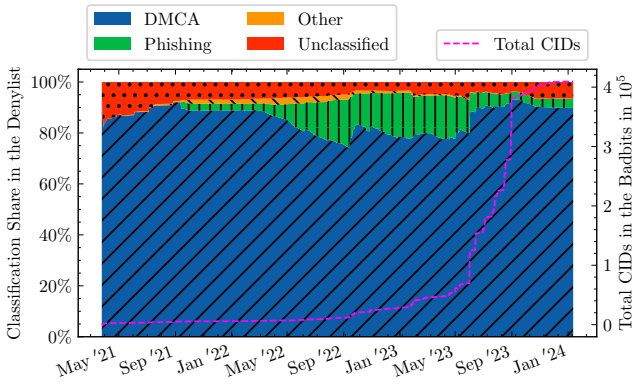

Figure 3: Badbits list evolution by type of threat.

Reaction Time. We are also curious to investigate the time needed to include content on the badbits list. This gives insight into the duration that content can circulate before being identified and moderated. Figure 4 plots the CDF of the number of days from the time we have first seen the content in our Bitswap and DHT logs to the inclusion of its CID on the badbits list. We also include the reaction time after receiving the associated takedown email (from 2023-10-22 until 2024-01-11). For comparison, we use the reaction time of APS in filtering phishing URLs requesting IPFS content through gateways.

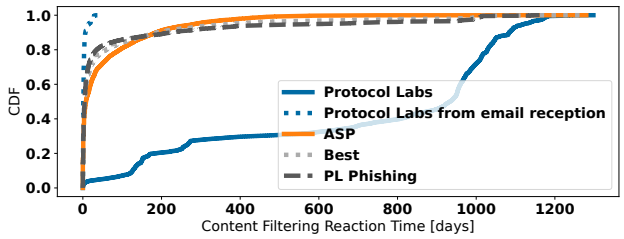

Figure 4: CDF with reaction time distributions for different providers.

For 50% of CIDs flagged for inclusion, the CID is placed on the badbits list within less than 24 hours. This suggests that the list *is* up-to-date and it rapidly reacts to reported content. However, this process naturally relies on the timely submission of takedown notices by third parties. That said, we witness a significant delay between the time content is first requested within IPFS and the time a takedown email is sent to Protocol Labs. This means that, on average, content is available in IPFS for 713 days before being included on the badbits list. In contrast, the reaction time is significantly shorter for APS, adding gateway URLs onto their list in less than a day for most of the CIDs. This indicates that they receive more timely takedown notices than Protocol Labs.

4.2 Overview of Badbits Content

The above has exposed an active and growing list of material in badbits. We next characterize this content.

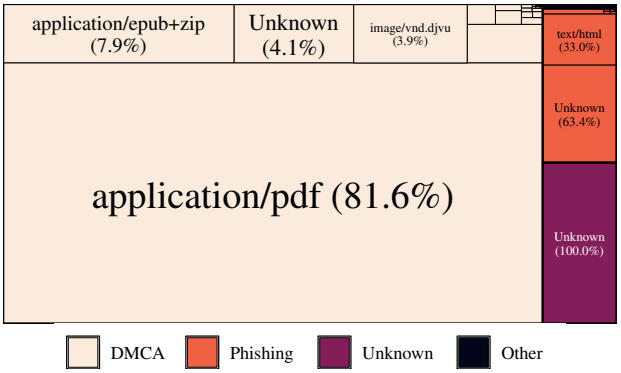

Figure 5: Percentage of content and threat type.

Content Overview. Figure 5 shows the threat and content types associated with denylist content. The "Unknown" file type indicates cases where we did not download the referenced file because of unavailability or it being labeled as CSAM. Reflecting our earlier observations, $\approx$ 87.97% of CIDs are filtered because of copyright violation (*i.e.* DCMA). This threat covers mostly PDF and epub files. The second most common threat is phishing (5.81%) with HTML files and images used as elements of deceptive websites. The remaining threats (*e.g.* malware, terrorist, and explicit content) constitute less than 1% of the CIDs on the denylist.

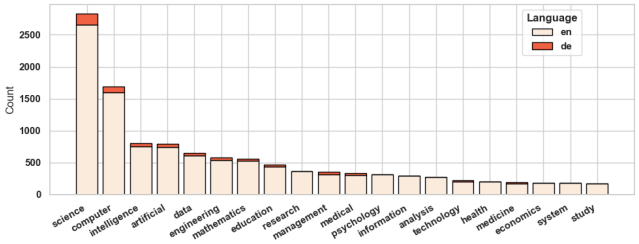

Figure 6: Frequency of subject area keywords for DMCA content.

DMCA. Figure 6 presents the language and subject/area of the DMCA documents. In total, we detect 41 languages. However, the vast majority of documents are in English (84%) and German (12.4%), likely making moderation reviews more straightforward. Interestingly, most of the documents represent academic articles and academic textbooks. We find that this is driven by multiple shadow libraries such as the Free Library, Anna's Archive, and Nexus Search [26, 29, 44] migrating their content to IPFS. This choice perhaps reflects a

perception in such communities that IPFS is an effective tool for hosting censorship-resistant material.

Phishing. Content classified as phishing contains mostly HTML files and images used for creating deceptive websites. In most cases, these appear to be phishing pages that are accessed via gateways using URLs. Specifically, attackers host their pages in IPFS but encourage normal web users to access them by providing HTTP URLs via the gateway services. To explore this further, for the HTML files, we obtain the target domain that the file is trying to imitate. We discover 195 victim services and websites and group them into categories. Figure 10 plots the distribution. 49.4% of these URLs target mail services, 8.8% target file transfer services such as Wetransfer, and 3% hit Microsoft services and websites. To give insight into this content, Figure 7 also presents a word cloud of the phishing image descriptions. Confirming the above victim analysis, we observe logos of multiple companies such as Apple, Yahoo, and Microsoft. Furthermore, the classic elements forming login forms and navigation elements are present (*e.g.* "sign", "envelope", "button"). Beyond the major mail portals, the remaining phishing targets are mainly shipment services such as DHL, financial companies like Visa and Mastercard, and social media networks. This suggests that there is, indeed, an important need for mitigation, as these seem to target high-risk areas, *e.g.* banking.

Terrorist Content. Concerningly, the denylist also contains 255 CIDs marked as terrorist content. This includes documents, videos, images and audio files. To understand the nature of this material, we focus on documents and images, which constitute more than 60% of the content. Figure 8 presents a word cloud of keywords in the content description. Surprisingly, we observe *love* and *poem* being the most repeated words. This results from a collection of literature acting as motivation for terrorist actions [49]. We further spot practical guides on how to blend into society without raising suspicions (*e.g. home*, *marriage*, *job*, *business*). The content relates to multiple groups and geographic locations (*kurdish workers*, *egypt*, *kuwait*). Finally, we observe multiple keywords related to religion and beliefs (*e.g. islam*, *traditional*, *anti-western*). Multiple PDFs contain Twitter screenshots with terrorist propaganda being reflected by the *tweets* keyword.

Explicit Content. Finally, our classification finds a small number of items on the denylist that could be classified as explicit. Figure 9 illustrates the keywords found in the image descriptions classified as explicit content. Multiple intuitive keywords (*e.g.* girl, naked, underwear) indicate the presence of pornography. For added safety, we send the SHA256 hashes of all the images to the IWF [43] that supports online organizations in filtering CSAM. Only three images were present in the IWF database and were classified as *hentai*, a style of Japanese pornographic anime and manga. For the content classified by Protocol Labs as CSAM, we are currently collaborating with the IWF to ethically and legally download and verify those files.

5 A Provider Perspective

Provider Geo-locations. Figure 11 shows the geographical distributions of all 10,668 unique badbits CID hosts that we observe between 2023-11-01 and 2023-11-30. We chose this timeframe because this is the only full month where our DHT and IPNI datasets overlap. Note that we identify unique providers by their peer ID. Figure 11a) shows that we can only geolocate around 30% of providing peers. This is because the provider records either do not contain network addresses and we could not find them in the DHT (20.5%), or the network addresses are private IP ranges (49.7%). The presence of private IPs in the provider records is an interesting phenomenon already reported in previous studies [32]. Potential explanations include misconfiguration (*e.g.* clients using the wrong interface) or outdated software. Only a minority of provider records contain non-private IP addresses that are not found in the Maxmind database ($< 0.1\%$).

Figure 11b) shows again the geographical distributions of all unique badbits CID providers, with each provider now weighted by the number of unique CIDs it hosts. We find 376,201 unique CIDs, with providers in the US and Netherlands sharing most. "Circuit" providers (aka *circuit relay* peers) are special to the IPFS network and exclusively relay control messages to facilitate NAT traversal [65] (*i.e.* they do not exchange traffic of actual content). Their presence hints at a large number of providers operating from residential networks — and therefore requiring NAT traversal. Their geographical location is, therefore, not necessarily representative of the geographical presence of the content providers that exchange actual content traffic. Their presence suggests that some parties may also use circuit relays to mask the hosting of badbits material.

CIDs per Provider. Figure 12 shows the distribution of the number of CIDs hosted per provider for all providers we observe in either the DHT (top) or the Indexer (bottom). The graphs show that, at any point in time, over 99% (DHT) and 70% (Indexer) of the providers host just between 1 and 10 badbits CIDs, respectively. However, about 0.1% (*i.e.* between one and three peers) of providers in the DHT host between 10^4 and 10^5 unique CIDs, which comprises up to 25% of the entire badbits list. In fact, we find single peers in the DHT and Indexers, respectively, that claim to provide more than 19% and 63% of the entire *badbits* list. This suggests that the badbits hosting ecosystem is relatively fragile, and significant volumes of content could be removed with just a small set of targeted takedowns.

Providers per CID. The above fragility may be misleading, as many different nodes could host an individual CID. Thus, Figure 13 shows the distribution of the number of providers

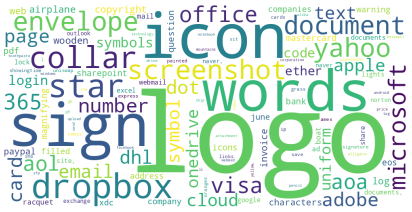

Figure 7: Word cloud of phishing image descriptions.

Figure 8: Word cloud of descriptions of blocked terror-related PDFs.

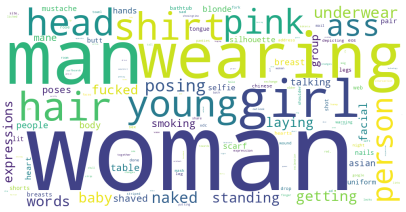

Figure 9: Word cloud of descriptions of explicit images.

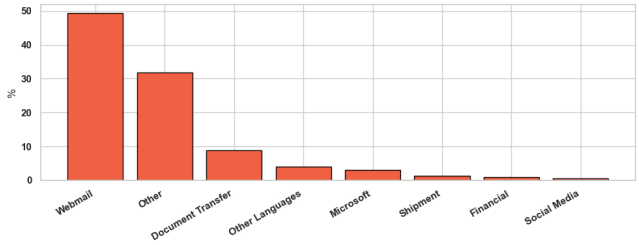

Figure 10: Frequency of services and websites targeted by phishing campaigns.

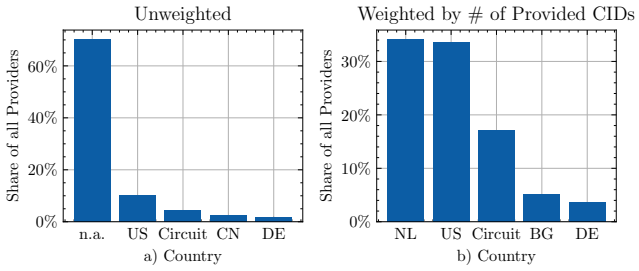

Figure 11: Geolocation distribution of 10,668 unique content providers identified by their peer ID aggregated from 2023-11-01 to 2023-11-30.

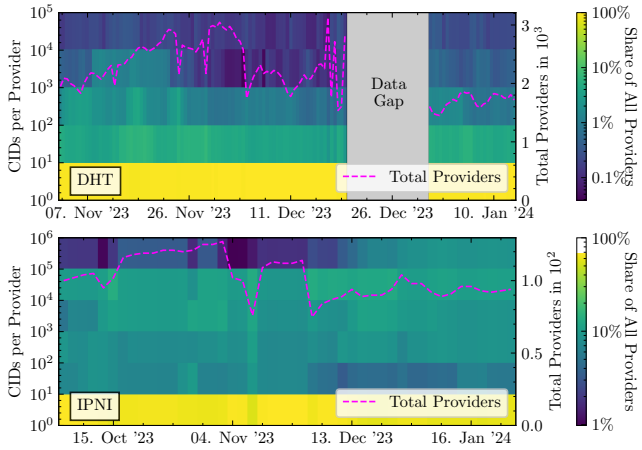

Figure 12: Distribution of the number of unique badbits CIDs hosted per provider and the total number of unique badbits CID providers. CIDs are grouped into bins according to the ticks in the left y-axis. The color indicates the share of all providers that host CIDs in any bin. The right y-axis shows the total number of badbits CID providers that were found in the DHT/IPNI at the time of the measurement.

hosting each CID. Under 0.1% of all provided CIDs were hosted by a single peer over the measurement period. This suggests that the ecosystem is, in fact, far less fragile than the earlier results would suggest: We even find a single CID from the badbits list that is provided by over 91% of all DHT and Indexer peers. Complicating matters, we also witness a growth in replication levels over time. Towards the end of the measurement period, the number of providers per badbits CID seen in the Indexer grows. By the end of our measurement period, the majority of CIDs have five to ten unique providers. We hypothesize that this may be because the Indexer does not automatically remove provider records itself, suggesting that some could be stale (see Section 2). Either way, we argue that this makes takedowns in IPFS more challenging.

CID Churn. Considering the presence of potential takedown attempts, we next measure the underlying churn of DHT providers on the badbits CIDs, depicted in Figure 14. Here, we measure churn by considering CIDs that started and stopped being provided within our total measurement window from 2023-11-01 and 2024-01-12. This means, from the 298,951 CIDs that were provided in the DHT (this excludes CIDs solely indexed in the Indexer), we only consider 60.4% (180,523). The remaining ones were either already present in the network when we started our measurements or were still provided after we stopped.

The graph shows that half of the CIDs added to the badbits are not retrievable after roughly two days. Although this could be because of takedown attempts, we suspect this is through natural churn. The graph also shows that CIDs classified as phishing have a lower churn rate and hence stay longer in the network. Looking at the Indexer, we find that indexed provider records exhibit virtually no churn as only 11 out of 304,495 badbits CIDs are removed during the measurement window from 2023-10-06 and 2024-01-28. We hypothesize that this is

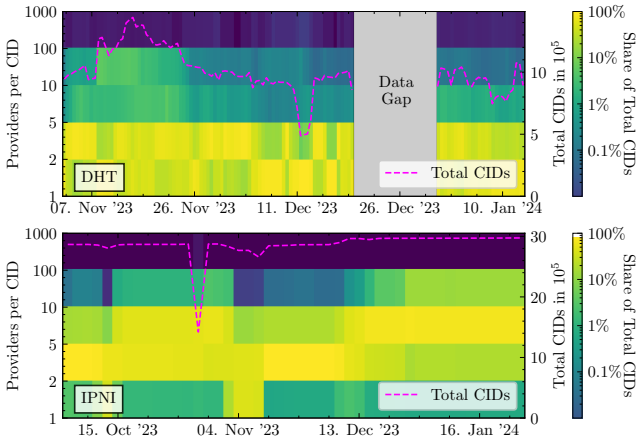

Figure 13: Distribution of the number of providers hosting a single CID and the total number of hosted badbits CIDs. Providers are grouped into bins according to the ticks on the left y-axis. The color indicates the share of all CIDs that providers host in any bucket host. The right y-axis shows the total number of badbits CIDs that were found in the DHT/IPNI at the time of the measurement.

again because Indexer records are not automatically removed.

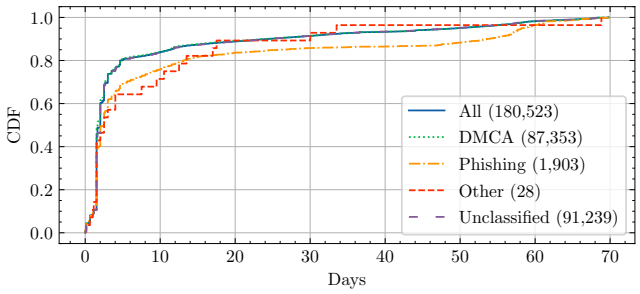

Figure 14: Provider record churn in the DHT. The numbers in parenthesis show the total number of CID uptime measurements for each curve. The measurement resolution is 12h.

6 A Consumer Perspective

We now turn our attention to *consumers*. We explore who requests denylist content through IPFS protocols and gateways, and estimate the popularity of such material.

Injection of new CID Requests. We first analyze when the denylist CIDs were first requested in the network. Figure 15 shows the number of unique, new denylist CIDs requested via Bitswap per month. The labeled dates indicate notable events in the IPFS user community, likely contributing to the reported numbers. At ①, the Free Library project, a shadow library similar to Libgen or Sci-Hub, announced that it was operating on IPFS [44, 75]. At ②, the Nexus project, another shadow library popular using Telegram bots to share content, announced the release of their latest index and dataset on IPFS [23]. At ③, Anna's Archive, an aggregation and indexing project for multiple shadow libraries and knowledge collections, announced they moved the contents of Z-Library, a shadow library recently taken down, to IPFS [28, 29]. Finally, at ④, the Nexus project announced a web frontend on IPFS/IPNS called the Standard Template Construct [26]. The project is particularly vocal about their use of IPFS to combat censorship, with frequent posts about updates and new material accessible via their interfaces as well as encouragements to install IPFS on user machines [24, 25, 27]. Note, the initial spike in 2021 is caused by the start of our data collection, *i.e.* all CIDss are new. Importantly, these bursty patterns create significant challenges for the moderation team, which must handle intense volumes of takedown notices in relatively short periods. Thus, needing to capacity manage for the peak (rather than the average) likely increases cost and complexity.

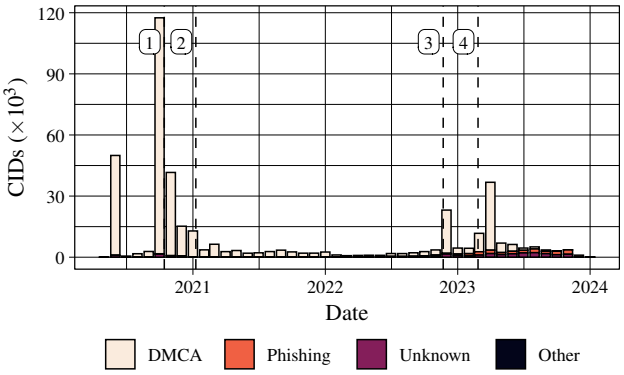

Figure 15: Number of denylisted CIDs seen via Bitswap per month.

Content Popularity. We next investigate the popularity of denylist CIDs. Specifically, we compare the number of denylist CID requests seen against the overall number of requests in our Bitswap traces. Over the entire period, we see an average of 14.23×10^3 denylist vs. 42.00×10^6 other unique CIDs per day. This suggests that only a very small ($\approx 0.034\%$) part of IPFS traffic requests denylist content. Closer inspection of Figure 15 shows that this almost exclusively consists of copyright-protected content. This is in sharp contrast with the HTTP gateway traffic that we analyze in the next section.

Figure 16 plots the number of requests for denylist content over time. For this plot, we first count the number of unique denylist CIDs, as well as the number of unique (CID, PeerID) tuples requesting denylisted CIDs, per month. Counting requests via unique (CID, PeerID) tuples avoids overcounting due to repeated requests, which happen when content is unavailable [31]. We observe a growing number of denylist CID requests, suggesting that demand for such content is expanding. Between January 2021 and November 2023, the number of requests increased 8-fold to almost 2×10^6 requests per

month. Further, we observe a sharp increase in the number of requests for CID classified as phishing starting from November 2022. In January 2024, the phishing queries constituted ≈ 50% of all requests for denylist content. This date correlates with the wider deployment of gateways, which facilitate access to IPFS content (see Section 2) and potentially help spreading IPFS-hosted phishing websites by making them available via HTTP. We suspect that these request patterns could also be exploited by the moderation team to help allocate resources to checking particular types of material.

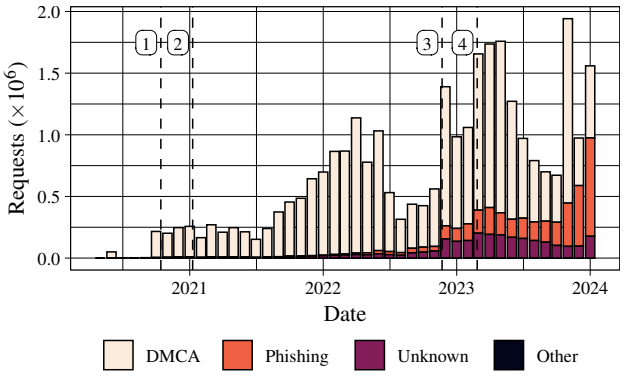

Figure 16: Number of unique peerID/CID combinations requesting denylisted CIDs via Bitswap, by threat type.

Geographical Request Flow. Now that we have a profile of both providers and consumers, we conclude by studying the flow of content between these two parties. Figure 17 shows the request flow from the request initiator to the denylist content provider's Maxmind location for all traced Bitswap requests and unique content providers (DHT and Indexer) from 2023-11-01 until 2023-11-30. Note that we count requests for CIDs hosted by several providers multiple times in the diagram (as we do not know the ultimate provider that the client selects). In total, we observe 5,597,343 requests and 10,668 unique content providers (as identified by their peer ID).

The Sankey diagram reveals a number of interesting trends. We see that the largest share of both requests and providers comes from the US. We see that some countries contribute more proportionate "demand" than "supply". For example, Germany requests 7x more unique badbits items than it serves itself. In contrast, the Netherlands only requests 50% of unique badbits CIDs that they provide.

7 A Gateway Perspective

In this section, we look at the HTTP gateways, as these are a rationale location where content filtering may take place. We investigate to what extent they block denylist content, and test potential filter circumvention techniques.

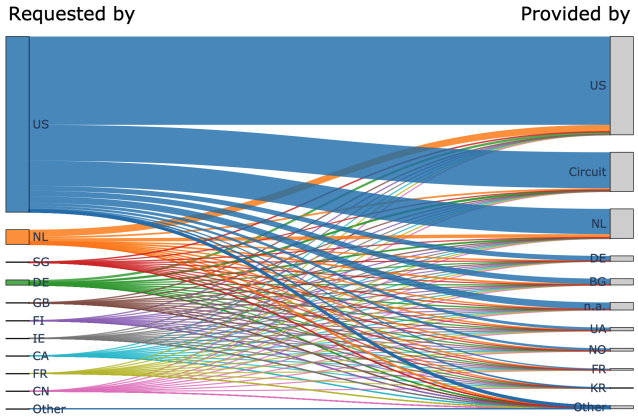

Figure 17: Geographical request flow (2023-11-01 to 2023-11-30). Same CID requests are considered multiple times.

7.1 Access via Gateways

We manually breakdown the 431 gateways in our dataset into four categories: 1. operated by Protocol Labs, 2. operated by a large CDN, 3. public gateways listed on the `ipfs.tech` gateway tracker [51] (excluding the ones in 1 and 2), 4. ephemeral gateways, 5. others. The *ephemeral* and *others* categories consist of unlisted gateways discovered from the ThreatChase Intelligence Platform and the passive DNS dataset. In this set, we identified 105 ephemeral gateways with hostnames belonging to a subdomain of a Web3.0 hosting service named Infura that provides IPFS Gateway-as-a-Service (GaaS) to its clients. The service enables a quick deployment of a custom, HTTP gateway. All the remaining gateways are classified as *others*.

We test the availability of content via these gateways by sending daily requests during the entire month of January 2024. To avoid overloading, we use a daily sample of 10,000 CIDs: 5,000 from badbits and 5,000 blocked by Web2 entities, *i.e.* ones from denylist but are not in badbits. Instead of requesting and downloading the content, an HTTP HEAD request is sent (as opposed to a GET) to which the gateway responds with only the HTTP header containing, among other things, an HTTP status code.

A gateway responds with HTTP status codes 410(Gone) and 451(UnavailableForLegalReasons) for filtered content, according to IPFS standards for public HTTP gateways [52]. We therefore classify responses containing those two codes as *Blocked*. In contrast, we assume the content is *Available* if 200(OK) is returned. We categorize requests that receive other status codes, such as 500(InternalServerError), 504(GatewayTimeout), etc. and those that do not produce a response within 10 seconds as *Failure*. This is mostly caused by the content no longer being available in IPFS.

Figure 18 presents the results of our HTTP requests. We observe significant inconsistent use of the badbits list across operators. The gateways operated by Protocol Labs block

nearly 100% of badbits, but this is not the case for other operators. The list is implemented, to some extent, by the public gateways that are officially listed by Protocol Labs, but not by the others. In particular, the CDN-operated gateways block only around 18% of the requests for badbits CIDs, similar to the gateways in the *others* category. We also observe low adoption of the CIDs moderated by Web2 providers. The gateways operated by Protocol Labs block none of them, while the CDN-operated gateways block up to 7%. At the same time, all the public and other gateways block some Web2-blocked CIDs (average 18%). We suspect that this is caused by the threat of the entire domain being blocked by mechanisms such as Google Safe Browsing (GSB). Such cases were reported in the past [68].

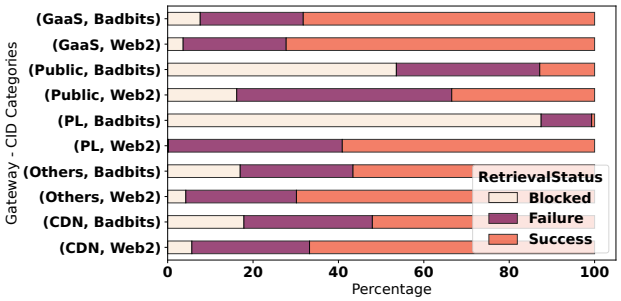

Figure 18: HTTP requests to gateways using a sample of CIDs from badbits and Web2 denylists.

We also use the passive DNS dataset to determine the CIDs requested from each gateway. A CID c can be requested from some gateways either as a part of the FQDN (*e.g.* `http://c.gw.tld/`) or as a part of the path (*e.g.* `http://gw.tld/ipfs/c`). The DNS dataset allows us to observe CIDs expressed in FQDN.[4] It is also a method that we noticed in more than 80% of URLs collected from APS.

Figure 19 shows the percentage of different types of requests for the most popular gateways measured by the overall number of requests. We observe that the gateways have a varied percentage of denylist requests, with one measured gateway showing 99% of requests attributed to phishing. This is in contrast to the DHT results with only $\approx 0.034\%$ denylist traffic share. This suggests that the gateways are frequently used in phishing campaigns to disseminate malicious content to victims without the need to install IPFS software.

7.2 Circumventing Gateway Filters

We conclude this section by testing several potential circumvention attacks on gateways. Specifically, there are multiple ways of requesting content via a gateway — thus, we check if all of these correctly implement filtering.

[4] this is the current recommended way of requesting CIDs from gateways

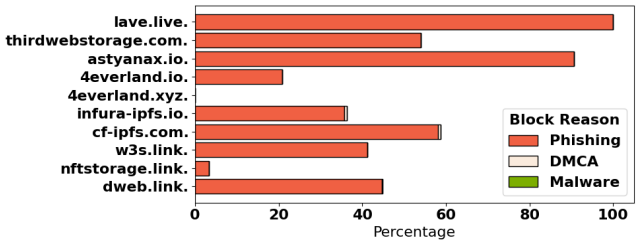

Figure 19: Percentage make-up of types of requests for the top-10 popular gateways. The Y-axis is sorted by the overall number of requests (the bottom is the most popular).

CID Versions. In our first experiment, we test whether gateways correctly hash the CIDs to the correct badbits list format before checking for a match. For that, we request blocked content using the canonical, version 1 CID, as well as the equivalent version 0 CID [2], which is not directly included in the badbits list. We randomly sample 100 CIDs out of those that are in the top 10% blocked by gateways. We request these, as well as their version 0 counterpart, on 2024-02-04, and note whether previously blocked content becomes available if addressed differently. We find no cases of this occurring and thus conclude that all evaluated gateways correctly normalize requested CIDs before checking against the badbits list.

Relative Paths. Next, we investigate how gateways handle requests for blocked data internally. IPFS allows the storage and retrieval of directories. We examine whether it is possible to trick gateways into providing blocked content by creating a wrapping directory around blocked objects. For a blocked CID c, a request would then take the form of `http://gw.tld/ipfs/`d`/foo`, where d is the CID of the directory and `foo` is the *name* of a directory entry referencing c. Upon receiving such a request, a gateway would first fetch d from the network, translate `foo` to the linked CID c, and then proceed with fetching c as usual. Assuming the parent collection CID is not blocked, we thus check whether the gateway correctly checks these internal fetch operations, instead of just the external, user-initiated ones.

We again sample 100 CIDs. We then create a directory collection wrapping these CIDs and provide it using a non-NATed node. Importantly, we *do not* provide any of the malicious content ourselves. Additionally, since the generated CID for this block is only known to us, the chance of it being used to spread malicious content is negligible.

We then attempt to retrieve the blocked content using a path relative to the CID of the generated parent. We find that only one gateway is vulnerable to this type of evasion. This gateway officially blocked 68 CIDs, yet we were able to circumvent the blocking for 67 of them ($\approx 98.5\%$). The failed evasion was not due to the content being blocked, but unavailable. We suspect the one vulnerable gateway is running outdated or custom software, whereas all other gateways use

a correct implementation. Notably, we also ran a preliminary version of this experiment[5] at an earlier date (2023-12-19) and found four additional vulnerable gateways operated by Protocol Labs. We followed responsible disclosure and they have now been fixed.

Alternative Hash Functions. The flexibility of CIDs allows users to generate multiple CIDs for the same content using different hash functions for the embedded digest. It is generally not possible to translate between these without access to the content itself, which would need to be re-hashed using the different functions. At the time of writing, the kubo [4] implementation of IPFS supports 72 hash functions.

We conjecture that attackers could circumvent blocks by generating multiple CIDs using different hash functions. We, therefore, test whether any of the gateways perform advanced, content-based blocking techniques, such as real-time content analysis or generation of alternative CIDs hashes for the requested content. We again sample 100 CIDs, selected from the set of phishing CIDs.[6] For these, we download the referenced content and re-add it to the network using a different CID (using a different hashing function). We then perform the same experiment as outlined earlier, and remove the content from our node immediately afterwards.

This circumvention attack is extremely effective. Out of the 57 gateways blocking any of the requested content, we are able to access the blocked content on 56. For these, evasion was possible for the majority (97.4%) of usually blocked CIDs. The missing evasions stem from unrelated issues, usually because the content is not returned completely within the timeout. The one gateway on which evasion was not possible, blocked *all* attempted evasions. Curiously, this was the one gateway that was vulnerable to our previous relative paths circumvention. We again suspect that they might be running custom software that does not support the alternative hash function. We have contacted them for clarification.

8 Discussion & Recommendations

We now discuss the limitations of the current content moderation approach and propose new improvements.

8.1 Improving the badbits List

We first discuss the quality of the badbits list itself. As shown in Figure 4, Protocol Labs usually processes all takedown requests within a day, suggesting effective decision-making. That said, traditional Web2 services such as APS and VirusTotal seem to get informed faster and thus be able to react quicker. We conjecture that this is caused by specializing in specific types of threats (*e.g.* phishing) and having larger user bases reporting data items. Therefore, we posit that working with established Web2 partners could shorten the time required to identify content. Automatic inclusion of content items into the *badbits* list based on trust could expedite the process. As a preliminary experiment, to simulate the process, we extract CIDs in URLs blocked by APS and compare them against the best detection time between Protocol Labs and APS. If these CIDs were automatically imported, such an enhanced denylist would classify 227% more CIDs and reduce the average reaction time by 43% compared to the current badbits.

We argue that the above process could be a mutually beneficial arrangement. We highlight this using our experiences with VirusTotal. When we initially submitted the SHA256 hashes of all the downloaded denylist files to VirusTotal, 1.2% of them were marked as malicious, and 58% of the files were not present in VirusTotal's knowledgebase. This represents a significant gap in VirusTotal's knowledge base. Indeed, when we checked VirusTotal again two weeks later, they discovered some of the files badbits had already seen. An additional 3618 files were newly labeled as malicious, including phishing (2421 files) and trojans (857 files), bringing the total to 2.2%. Thus, we posit that inputs from the badbits list team could help expand VirusTotal's knowledgebase faster, identifying content worth checking. A good candidate for such verification would be by sharing content requested through gateways. We found that this content has a much higher probability of being harmful (Figure 19). We recognize that operators might be unwilling to provide real-time feeds of content requested through their gateways due to privacy concerns. However, the inclusion of CIDs in the FQDN [3] means such data can be seen through passive DNS monitoring anyway.

Another limitation of the current *badbits* list is that it does not contain any taxonomy of content type or the reason for its inclusion. We hypothesize that gateway operators have different views on which types of threats should be filtered. Indeed, we see that some gateways do not follow the full badbits list (Figure 18). Thus, a concrete recommendation is that the badbits list is augmented with information about the nature of the content to better encourage uptake (*e.g.* allowing gateways to only filter certain types of material). The list should also have separate feeds for different jurisdictions, especially regarding the DMCA content.

8.2 Improving Web2 URL Filtering

Many Web2 filter lists already include URLs for the IPFS gateways, *e.g.* APS and GSB. We argue that their approach, however, stems from a misunderstanding of how IPFS gateways work. For example, we find that most APS blocked IPFS URLs use the gateway URL (*i.e.* gateway + CID combination). However, a CID can simply be requested via another gateway. Thus, once access to a CID is blocked through a gateway, malicious actors could just change the gateway domain to

[5] see Appendix A.1 for details in the difference in methodology
[6] see Section 9 for ethical discussions about this experiment

another one. Gateways-as-a-Service providers simplify the process of deploying short-lived gateways.

We also repeat the same checks for GSB. To test this, we create URLs from every combination of denylist CIDs and gateways in our dataset and submit them to GSB to check. Surprisingly, we find that the GSB blocks *entire* gateways, rather than specific content URLs provided by them. We observe that GSB blocks 56 gateways entirely, yet it allows the downloading of *all* denylist content from the remaining gateways. In practice, this is extremely ineffective, particularly as 95% of blocked gateways are already offline. The remaining 5% does not implement any filtering, suggesting that they could have been serving harmful content (helping to explain why GSB blocks them).

Clearly, both of the above approaches to compiling URL filter lists are ineffective for blocking IPFS gateway material. Instead, we argue that they should maintain a list of blocked CIDs — not gateway URLs. These CIDs can then be used to generate a full set of URLs by combining them with all known gateway domains. We are currently engaging with OpenPhish, the Anti-Phishing Working Group, and Phishtank to advocate for this change.

8.3 Improving IPFS Node Filtering

To date, the badbits list is mainly used by the gateways to implement filtering. However, it still remains possible for nodes to request badbits CIDs via the native IPFS DHT. As of IPFS v0.24.0, there is support for blocklists that filter their local files. However, this has not seen widespread uptake. To improve efficacy, we argue that more comprehensive versions of moderation should be introduced into the native IPFS network. Unfortunately, this raises many philosophical, ethical and technical questions. Our final recommendation is that the IPFS developers explore the development of decentralized moderation schemes that can answer such questions.

9 Ethical Considerations

Our study raises certain ethical queries, which we discuss below. We sought ethical guidance from multiple institutions. We got our methodology, datasets and the submitted version of the manuscript reviewed by Protocol Labs. We further coordinated our work with the IWF and got ethical approval from the IRB of our host institution.

Our overarching goal is to derive general, non-personalized insights about the spread of problematic content on IPFS. We use collected data only for purposes that are permissible with respect to Art. 6(4)GDPR. We operate within the bounds of European data protection law, specifically Art. 5(1)(e)GDPR. We host a user-facing website with our data management policy.[7] We are following data minimization practices: 1) we store the sensitive data for up to 24 months or until we do not need it anymore to complete our research, whichever comes sooner; 2) we are constantly reflecting on our data storage practices; 3) if we conclude that some data fields or whole data sets are no longer needed for our intended purpose, we delete them accordingly. All the acquired content was immediately deleted after analysis.

We emphasize that the badbits dataset is publicly available, and there is no intrinsic way for us to map CIDs to private metadata (*e.g.* the creator of the file). With permission, the takedown request information came from researchers formerly at Protocol Labs. They only accessed the data while still working at Protocol Labs and focused on only the CIDs and the reason for the takedown request contained within the email, which was parsed via script (as shown in Exhibit A.2). Information about the email complainants was not reviewed or used as part of this study.

Due to the sensitivity of the content, we also avoid manual inspection of any material. To circumvent this, we took extensive steps to formulate automatic pipelines capable of classifying the content objects. This does involve downloading each object once. We configure our IPFS node to not provide the downloaded content to other peers. We never inspect the material ourselves and delete the content immediately after classifying it. Importantly, at no point did we download any content recorded as part of a CSAM takedown request. Instead, we shared this information with the IWF and followed their best practice advice at all times.

The files submitted to VirusTotal are publicly available. We only upload those contained on lists curated by Protocol Labs or APS. VirusTotal T&S indicate that the data is stored and shared with authorized parties to "raise global IT security levels". We followed best practices as followed by numerous previous studies that used VirusTotal [56, 59, 69]. Our contribution enabled VirusTotal to detect an additional 3618 malicious files (phishing and trojans) and protect multiple users from online harm.

We note that some of our measurements (the Bitswap, DHT and gateway queries) introduce additional load onto IPFS. As part of this, we rate-limited our queries to reduce the load and, overall, believe that the study had a very minimal impact on operations. This was confirmed by the former Protocol Labs employees themselves, giving us confidence in its safety. Unavoidably, these particular query datasets involved collecting IP addresses. We do not attempt to map these back to personal identities, as such analysis is not within the scope of this study. Furthermore, after performing the geolocation and AS mappings, we make no further use of any IP addresses. Following this step, we perform IP address anonymization using salted hashing The salt is discarded after anonymization to prevent reverting the hash. Finally, we perform no per-user analysis, and focus only on overall system analysis, aggregating behaviors together.

When requesting data from gateways, we used HTTP

[7] https://monitoring.ipfs.trudi.group/privacy_policy.html

HEAD requests without actually downloading the files. This minimizes the overhead and does not expose gateway operators to legal risks. The gateway filtering evasion experiments raise certain ethical concerns because we try to circumvent the blocking of malicious content, and because we ourselves had to provide some data to the network to perform these experiments. For the CID version experiment, we do not provide any data and only check whether gateways correctly implement the specification. For the directory entry evasion experiment, we publish the wrapping directory block, but no malicious content. This experiment also only checks conformance with the suggested blocking methods. The last experiment, for which we provide content using a different hashing function (and thus an entirely new CID) raises limited ethical concerns. The newly-generated CIDs are opaque and known only to us. We stop providing the content after we concluded the experiment. It is thus extremely unlikely the content could spread due to this experiment.

10 Related Work

IPFS. In recent years, numerous studies focus on IPFS. Trautwein et al. [72] present the design of IPFS and evaluate its deployment and performance. Balduf et al. [31] conduct a study monitoring data requests in IPFS. They also examined the viability of running IPFS in restrictive networks, potentially evading censorship [33]. Wu et al. [77] explore the feasibility of using IPFS for decentralized video streaming. Sridhar et al. [67] investigate possible content censorship issues in IPFS. Balduf et al. [32] present a study on the decentralization of IPFS and its implications. They similarly highlighted the importance and high usage of public gateways [30], in particular for NFT-related content. Daniel and Tschorsch [37] propose a method for passively measuring IPFS churn and network size. Prünster et al. [66] discuss the potential eclipse attack on IPFS networks. Trautwein et al. [73] explore techniques for accelerating record storage in IPFS. Wei et al. [76] examine the role of centralization in IPFS and its impact on system dynamics. In contrast to these studies, we focus on content moderation activities in IPFS, and the availability of problematic material. To the best of our knowledge, no prior works have studied this. Our findings can contribute to the design, performance, and security aspects of IPFS.

Moderation in Decentralized Systems. Closest to our work is research on malicious content and moderation in P2P networks, particularly BitTorrent. Cuevas et al. investigate content publishing motivations while proposing TorrentGuard to combat scam and malware distribution [34, 35]. Farahbakhsh et al. examine anti-piracy actions' impact on BitTorrent publishers [41]. Fetscherin analyze movie piracy on KaZaA, highlighting challenges in decentralized file-sharing [42]. Zhou et al. and Kalafut et al. address threats from P2P worms and malware, emphasizing defense strategies [48, 78]. Zuo *et al.* [80, 81] look at moderation in the memo.cash platform. Das *et al.* [38] analyze the NFT ecosystem for security issues. While these studies concentrate on P2P networks, our research delves into similar challenges within IPFS. This is particularly significant considering its evolving role in the Decentralized Web as a storage layer.

11 Conclusion

This paper has studied content moderation in IPFS, one of the largest platforms in the growing "Decentralized Web". We identified, characterized and measured the presence of problematic content in IPFS. We analyzed 368,762 files flagged for inclusion on the Badbits denylist and found copyright material, phishing scams, and even terrorist material.

We also examined whether IPFS gateways filter this content. Gateways play a critical role by facilitating access to IPFS content via HTTP. Although the gateways operated by Protocol Labs blocked all content with accepted takedown requests, there was limited adherence by other operators. For example, gateways operated by CDNs blocked just 19% of such content. We also show that most gateways are vulnerable to simple attacks, *e.g.* using alternative hash functions for naming. We showed that this material was highly available within the IPFS network as there is limited (or no) filtering on IPFS nodes.

We addressed this problem with proposals to improve the speed and quality of the content moderation process. Our solutions increased in 227% the detection of phishing material and reduced the average time to filter such content by 43%. We hope that this work will help developers and operators in addressing the very challenging problem of content moderation in a decentralized ecosystem.

References

[1] Anti-phishing working group: Unifying the global response to cybercrime. https://apwg.org.

[2] Content identifiers. https://docs.ipfs.tech/concepts/content-addressing/. Accessed: 2024-01-30.

[3] Ipfs gateway resolution style. https://docs.ipfs.tech/concepts/ipfs-gateway/#gateway-request-lifecycle. Accessed: 2024-02-09.

[4] An IPFS implementation in Go. https://github.com/ipfs/kubo, 2016. Accessed: 2024-02-08.

[5] Discussify. https://github.com/ipfs-shipyard/discussify-browser-extension, 2018. Accessed: 2023-05-26.

[6] DTube. https://d.tube, 2018. Accessed: 2024-01-30.

[7] Ethlance. https://github.com/district0x/ethlance, 2018. Access: 2023-22-01.

[8] Gala games. https://app.gala.games/games, 2020. Accessed: 2023-05-26.

[9] IPFS ecosystem directory. https://ecosystem.ipfs.tech/, July 2020. Accessed: 2023-06-03.

[10] Matters News. https://matters.news, 2020. Accessed: 2023-05-26.

[11] dClimate. https://www.dclimate.net, 2021. Accessed: 2023-05-26.

[12] Deece. https://github.com/navinkeizer/Deece, 2021. Accessed: 2023-05-26.

[13] Audius. https://audius.org, 2022. Accessed: 2023-05-26.

[14] Cisco: Interplanetary file system seeing 'widespread' abuse by hackers. https://therecord.media/cisco-interplanetary-file-system-seeing-widespread-abuse-by-hackers, 2022.

[15] Space. https://github.com/ipfs-shipyard/space, November 2022. Accessed: 2023-05-30.

[16] Splinterlands. https://splinterlands.com/, 2022. Accessed: 2023-05-26.

[17] OpenPhish: Phishing Intelligence. https://openphish.com/, 2023. Accessed: 2024-02-02.

[18] PhishTank: Join the fight against phishing. https://phishtank.org/, 2023. Accessed: 2024-02-02.

[19] Publishers carpet-bomb ipfs gateway operators with dmca notices. https://torrentfreak.com/publishers-carpet-bomb-ipfs-gateway-operators-with-dmca-notices-230625/, 2023.

[20] Threat actors abuse interplanetary file system protocol to spread malware. https://duo.com/decipher/threat-actors-abuse-interplanetary-file-system-protocol-to-spread-malware, 2023.

[21] URLhaus. https://urlhaus.abuse.ch, 2023. Accessed: 2024-02-02.

[22] Content Identifiers (CIDs). https://docs.ipfs.tech/concepts/content-addressing/, 2024.

[23] Telegram admins of the Nexus Channel. Update note on nexus index «hermes». https://t.me/nexus_search/31, January 2021. Accessed: 2024-02-02.

[24] Telegram admins of the Nexus Channel. Distributing all shadow libraries. https://t.me/nexus_search/168, October 2023. Accessed: 2024-02-06.

[25] Telegram admins of the Nexus Channel. Our own ipfs infrastructure. https://t.me/nexus_search/177, November 2023. Accessed: 2024-02-06.

[26] Telegram admins of the Nexus Channel. Standard template construct - copyright endgame. https://t.me/nexus_search/103, February 2023. Accessed: 2024-02-02.

[27] Telegram admins of the Nexus Channel. Weekly digest. https://t.me/nexus_search/171, October 2023. Accessed: 2024-02-06.

[28] annas-blog.org authors. Help seed Z-Library on IPFS. https://annas-blog.org/help-seed-zlibrary-on-ipfs.html, November 2022. Accessed: 2024-02-02.

[29] annas-blog.org authors. Putting 5,998,794 books on IPFS. https://annas-blog.org/putting-5,998,794-books-on-ipfs.html, November 2022. Accessed: 2024-02-02.

[30] Leonhard Balduf, Martin Florian, and Björn Scheuermann. Dude, where's my NFT: Distributed infrastructures for digital art. In *Proceedings of the 3rd International Workshop on Distributed Infrastructure for the Common Good*, DICG '22, page 1–6, New York, NY, USA, 2022. Association for Computing Machinery.

[31] Leonhard Balduf, Sebastian Henningsen, Martin Florian, Sebastian Rust, and Björn Scheuermann. Monitoring data requests in decentralized data storage systems: A case study of IPFS. In *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*, pages 658–668, 2022.

[32] Leonhard Balduf, Maciej Korczyński, Onur Ascigil, Navin V Keizer, George Pavlou, Björn Scheuermann, and Michał Król. The cloud strikes back: Investigating the decentralization of IPFS. In *Proceedings of the 2023 ACM on Internet Measurement Conference*, pages 391–405, 2023.

[33] Leonhard Balduf, Sebastian Rust, and Björn Scheuermann. I'm InterPlanetary, get me out of here! accessing IPFS from restrictive environments. In *Proceedings of the 4th International Workshop on Distributed Infrastructure for the Common Good*, pages 13–18, 2023.

[34] Ruben Cuevas, Michal Kryczka, Angel Cuevas, Sebastian Kaune, Carmen Guerrero, and Reza Rejaie. Is content publishing in bittorrent altruistic or profit-driven? In *Proceedings of the 6th International COnference*, pages 1–12, 2010.

[35] Rubén Cuevas, Michal Kryczka, Roberto González, Angel Cuevas, and Arturo Azcorra. Torrentguard: Stopping scam and malware distribution in the bittorrent ecosystem. *Computer Networks*, 59:77–90, 2014.

[36] KOR Labs Cybersecurity. ThreatChase open platform for protection against phishing. https://threatchase.korlabs.io/, 2023.

[37] Erik Daniel and Florian Tschorsch. Passively measuring ipfs churn and network size. In *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pages 60–65. IEEE, 2022.

[38] Dipanjan Das, Priyanka Bose, Nicola Ruaro, Christopher Kruegel, and Giovanni Vigna. Understanding security issues in the nft ecosystem. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, page 667–681, New York, NY, USA, 2022. Association for Computing Machinery.

[39] Alfonso De la Rocha, David Dias, and Yiannis Psaras. Accelerating content routing with bitswap: A multi-path file transfer protocol in ipfs and filecoin, 2021.

[40] DomainTools. Farsight dnsdb. https://www.domaintools.com/products/farsight-dnsdb/.

[41] Reza Farahbakhsh, Angel Cuevas, Rubén Cuevas, Reza Rejaie, Michal Kryczka, Roberto Gonzalez, and Noel Crespi. Investigating the reaction of bittorrent content publishers to antipiracy actions. In *IEEE P2P 2013 Proceedings*, pages 1–10. IEEE, 2013.

[42] Marc Fetscherin. Movie piracy on peer-to-peer networks—the case of kazaa. *Telematics and Informatics*, 22(1-2):57–70, 2005.

[43] Internet Watch Foundation. Eliminating child sexual abuse online. https://iwf.org.uk, February 2024. Accessed: 2024-02-02.

[44] freeread.org authors. IPFS Free Library. https://freeread.org/ipfs/, 2020. Accessed: 2024-02-02.

[45] Google. Bard - chat based ai tool from google. https://bard.google.com, February 2024. Accessed: 2024-02-02.

[46] Damilola Ibosiola, Ignacio Castro, Gianluca Stringhini, Steve Uhlig, and Gareth Tyson. Who watches the watchmen: Exploring complaints on the web. In *The World Wide Web Conference*, pages 729–738, 2019.

[47] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[48] Andrew Kalafut, Abhinav Acharya, and Minaxi Gupta. A study of malware in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, page 327–332, New York, NY, USA, 2006. Association for Computing Machinery.

[49] Elisabeth Kendall and Ahmad Khan. Jihadist propaganda and its exploitation of the arab poetic tradition. *Reclaiming Islamic tradition: Modern interpretations of the classical heritage*, pages 223–246, 2016.

[50] Protocol Labs. Bad bits denylist. https://badbits.dwebops.pub/. Accessed: 2024-02-06.

[51] Protocol Labs. IPFS Public Gateway Checker. https://ipfs.github.io/public-gateway-checker/. Accessed: 2024-02-02.

[52] Protocol Labs. Path gateway specification. https://specs.ipfs.tech/http-gateways/path-gateway/. Accessed: 2024-02-02.

[53] Protocol Labs. Bad Bits. https://badbits.dwebops.pub/, February 2024. Accessed: 2024-02-02.

[54] Protocol Labs. CID Contact. https://cid.contact/, February 2024. Accessed: 2024-02-02.

[55] Protocol Labs. Report copyright infringement on the ipfs gateway service. https://docs.google.com/forms/d/e/1FAIpQLSekmfikDJ5mIS5YugnAinTfiuyJ4BgkqkkX18DldpirC3dUQ/viewform, November 2024. Accessed: 2024-02-02.

[56] Stevens Le Blond, Adina Uritesc, Cédric Gilbert, Zheng Leong Chua, Prateek Saxena, and Engin Kirda. A look at targeted attacks through the lense of an {NGO}. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 543–558, 2014.

[57] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[58] libp2p. Hydra Booster, 2017.

[59] Yun Lin, Ruofan Liu, Dinil Mon Divakaran, Jun Yang Ng, Qing Zhou Chan, Yiwen Lu, Yuxuan Si, Fan Zhang, and Jin Song Dong. Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3793–3810, 2021.

[60] RTrade Technologies Ltd. Temporal. https://temporal.cloud, 2020. Accessed: 2023-08-01.

[61] Petar Maymounkov and David Mazieres. Kademlia: A peer-to-peer information system based on the XOR metric. In *International Workshop on Peer-to-Peer Systems*, IPTPS. Springer, 2002.

[62] Almonit Organization. Almonit. https://almonit.eth.link/, 2021. Accessed: 2023-06-02.

[63] Constantinos Patsakis and Fran Casino. Hydras and ipfs: a decentralised playground for malware. *International Journal of Information Security*, 18:787–799, 2019.

[64] Ian Preston. Peergos. https://peergos.org. Accessed: 2023-07-01.

[65] Protocol Labs. Hole punching in libp2p - Overcoming Firewalls. https://blog.ipfs.tech/2022-01-20-libp2p-hole-punching/, March 2022. Accessed: 2023-09-12.

[66] Bernd Prünster, Alexander Marsalek, and Thomas Zefferer. Total eclipse of the heart – disrupting the InterPlanetary file system. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 3735–3752, Boston, MA, August 2022. USENIX Association.

[67] Srivatsan Sridhar, Onur Ascigil, Navin Keizer, François Genon, Sébastien Pierre, Yiannis Psaras, Etienne Rivière, and Michał Król. Content censorship in the interplanetary file system. 2024.

[68] NFT Storage. Gateways and gatekeepers: How Google's mistakes can still affect web3. https://blog.nft.storage/posts/2022-04-29-gateways-and-gatekeepers. Accessed: 2024-02-02.

[69] Octavian Suciu, Radu Marginean, Yigitcan Kaya, Hal Daume III, and Tudor Dumitras. When does machine learning {FAIL}? generalized transferability for evasion and poisoning attacks. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1299–1316, 2018.

[70] Berty Technologies. Berty. https://berty.tech, 2018. Accessed: 2023-05-26.

[71] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[72] Dennis Trautwein, Aravindh Raman, Gareth Tyson, Ignacio Castro, Will Scott, Moritz Schubotz, Bela Gipp, and Yiannis Psaras. Design and evaluation of ipfs: a storage layer for the decentralized web. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pages 739–752, 2022.

[73] Dennis Trautwein, Yiluo Wei, Ioannis Psaras, Moritz Schubotz, Ignacio Castro, Bela Gipp, and Gareth Tyson. Ipfs in the

fast lane: Accelerating record storage with optimistic provide. In *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*. IEEE, May 2024.

[74] trudi group. IPFS Network Size Estimates. https://grafana.monitoring.ipfs.trudi.group/, 2022. Accessed: 2024-02-01.

[75] Reddit user shrine. p2p Free Library: Help build humanity's free library on IPFS with Sci-Hub and Library Genesis. https://reddit.com/r/DataHoarder/comments/jb1hkn/p2p_free_library_help_build_humanitys_free/, October 2020. Accessed: 2024-02-02.

[76] Yiluo Wei, Dennis Trautwein, Yiannis Psaras, Ignacio Castro, Will Scott, Aravindh Raman, and Gareth Tyson. The eternal tussle: Exploring the role of centralization in ipfs. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*. USENIX Association, April 2024.

[77] Zhengyu Wu, ChengHao Ryan Yang, Santiago Vargas, and Aruna Balasubramanian. Is ipfs ready for decentralized video streaming? In *Proceedings of the ACM Web Conference 2023*, WWW '23, page 3002–3010, New York, NY, USA, 2023. Association for Computing Machinery.

[78] Lidong Zhou, Lintao Zhang, Frank McSherry, Nicole Immorlica, Manuel Costa, and Steve Chien. A first look at peer-to-peer worms: Threats and defenses. In Miguel Castro and Robbert van Renesse, editors, *Peer-to-Peer Systems IV*, pages 24–35, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[79] ZorrillosDev. Watchit. https://watchit.movie/, 2022. Accessed: 2023-05-23.

[80] Wenrui Zuo, Aravindh Raman, Raul J MondragÓN, and Gareth Tyson. A first look at user-controlled moderation on web3 social media: The case of memo. cash. In *Proceedings of the 3rd International Workshop on Open Challenges in Online Social Networks*, pages 29–37, 2023.

[81] Wenrui Zuo, Aravindh Raman, Raul J Mondragón, and Gareth Tyson. Set in stone: Analysis of an immutable web3 social media platform. In *Proceedings of the ACM Web Conference 2023*, pages 1865–1874, 2023.

A Additional Methodology

A.1 Initial Gateway Evasion Methodology

For the exploratory directory entry gateway evasion experiment performed in December 2023, we did not create our own directory block wrapper around often-blocked content. Instead, we searched for occurrences of blocked CIDs in *existing* directories. This limits both the directories used as well as the blocked content used for testing, since we needed to find these directory entries.

A.2 Email Classification

The code snippet in Listing 1 shows the methodology to classify the takedown emails.

```
for email in emails:

    sender_domain = email.sender_domain()
    lower_subject = email.subject.lower()
    lower_body = email.body.lower()

    if "Copyright Enforcement Notice" in email.subject:
        email.takedown_type = TakedownType.DMCA
    elif "Notice of DMCA removal from Google Search" in email.body:
        email.takedown_type = TakedownType.DMCA
    elif "Notice of Claimed Infringement" in email.subject:
        email.takedown_type = TakedownType.DMCA
    elif "copyright infringing material" in lower_body:
        email.takedown_type = TakedownType.DMCA
    elif "DMCA Takedown Notice" in email.subject:
        email.takedown_type = TakedownType.DMCA
    elif "alerts@terrorismanalytics.org" in email.from_addresses:
        email.takedown_type = TakedownType.TERRORIST
    elif "Islamic state propaganda" in email.body:
        email.takedown_type = TakedownType.TERRORIST
    elif "child sexual abuse material" in lower_subject:
        email.takedown_type = TakedownType.CSAM
    elif "child sexual abuse material" in lower_body:
        email.takedown_type = TakedownType.CSAM
    elif "cruel animal abuse content" in lower_subject:
        email.takedown_type = TakedownType.AAM
    elif "cruel animal abuse material" in lower_subject:
        email.takedown_type = TakedownType.AAM
    elif "cruel animal abuse material" in lower_body:
        email.takedown_type = TakedownType.AAM
    elif "phising" in lower_subject:  # typo
        email.takedown_type = TakedownType.PHISHING
    elif "phishing" in lower_subject:
        email.takedown_type = TakedownType.PHISHING
    elif "phish" in lower_subject:
        email.takedown_type = TakedownType.PHISHING
    elif "phishing" in lower_body:
        email.takedown_type = TakedownType.PHISHING
    elif "spamcop.net" == sender_domain:
        email.takedown_type = TakedownType.PHISHING
    elif "truelite.it" in email.reply_to_domains:
        email.takedown_type = TakedownType.PHISHING
    elif "scam email" in lower_body:
        email.takedown_type = TakedownType.PHISHING
    elif "restore" in lower_subject:
        email.takedown_type = TakedownType.UNBLOCK
    elif "unblock" in lower_body:
        email.takedown_type = TakedownType.UNBLOCK
    elif "unblock" in lower_subject:
        email.takedown_type = TakedownType.UNBLOCK
    elif "hosting a fake site that contains unauthorized resources" in
       lower_body:
        email.takedown_type = TakedownType.PHISHING
    else:
        unblock_strings = [
            "unable to download pdf",
            "the content is not illegal",
            "the content that i requested has been blocked",
            "contact has been blocked in error",
            "unable to download books",
            "error block",
            "ircorrectly blocked",
            "incorrectly blocked",
            "blocked in error",
            "please unblock"
        ]
        for str in unblock_strings:
            if str in email.subject.lower() or str in email.body.lower():
                email.takedown_type = TakedownType.UNBLOCK
                break
```

Listing 1: Takedown Email Classification

A.3 GitHub Information Classification

The query in Listing 2 shows how we identified any data points that we could not match towards one of our predefined categories. It also indicates which keywords correspond to which takedown type.

```
select *
from github_information gh
where not ( -- dmca
       gh.title like '%DMCA%'
    or lower(gh.title) like '[dmca]%'
    or lower(gh.title) like '%dmca removal%'
    or lower(gh.title) like '%copyright%'
    or lower(gh.title) like '%infringing material%'
    or lower(gh.title) like '%infringing activity%'
    or lower(gh.title) like '%claimed infring%'
    or lower(gh.title) like '%rights infringement notice%'
    or lower(gh.title) like '%movie%'
    or lower(gh.body) like '%digital millennium copyright act%'
    or lower(gh.body) like '%infringing activity%'
    or lower(gh.body) like '%infringing content%'
    or lower(gh.body) like '%copyright infringement notice%'
    or lower(gh.body) like '%copyright removal request%'
)
and not ( -- terrorist
       lower(gh.title) like '%terrorism%'
    or lower(gh.title) like '%terrorist%'
    or lower(gh.title) like '%daesh%'
    or lower(gh.title) like '%islamic state propaganda%'
    or lower(gh.title) like '%isis west africa%'
    or lower(gh.title) like '%isis propaganda%'
)
```

```
and not ( -- csam
        lower(gh.title) like '%child sexual abuse%'
    or lower(gh.title) like '%csam%'
    or lower(gh.body) like '%child sexual abuse%'
)
and not ( -- malware
        lower(gh.title) like '%malware%'
)
and not ( -- phishing
        lower(gh.title) like '%phish%'
    or lower(gh.title) like '%social engineering%'
    or lower(gh.body) like '%phish%'
    or lower(gh.body) like '%fishing%'
    or lower(gh.title) like '%wetransfer%'
    or lower(gh.title) like '%criminal scam website%'
    or lower(gh.body) like '%domain abuse%'
    or lower(gh.title) like '%domain abuse%'
    )
and not ( -- not possible
    gh.title ~ '\[takedown\] \d+ requests: ''Takedown request \d+'''
)
and not (
    gh.body = ''
);
```

Listing 2: Takedown GitHub Information

A.4 GPT Query Process

To generate summaries and extract information using the GPT model, there are three main components: the *pre-prompt*, the *prompt*, and the *instruction*. Each component serves a specific purpose in guiding the GPT model's response.

Pre-prompt. The *pre-prompt* sets the context and provides any necessary instructions or information before querying the GPT model. It primes the model with relevant details to ensure accurate and coherent responses. For text extraction we use the following:

```
pre_prompt_extraction0 = "In order to extract information, pay attention to the
following file\n"
```

For summaries, we do not provide a pre-prompt. For queries, we use:

```
pre_prompt_query0 = "Pay attention and remember the information below, which
       will help to answer the question or imperative after the context ends.\n"
```

Prompt. The *prompt* is the actual query posed to the GPT model. It directs the model on what information to generate or extract based on the given context. For text extraction, we use:

```
prompt_extraction0 = "Using only the information in the document/file sources
       above, extract: \n"
```

For summaries, we use:

```
prompt_summary0 = "Using only the information in the document/file/ sources
       above, write a condensed and concise summary of key results in a single
       sentence."
```

For queries, we use:

```
    prompt_query0 = "According to only the information in the document/file
       sources provided within the context above, "
```

Instruction. The *instruction* specifies the task or objective to be accomplished by the GPT model. It provides further guidance to ensure that the model's response aligns with our research goals. For PDF summaries, we use:

```
    instruction= "Give a one-line summary of this document"
```

However, we have different prompts for each content type because we observed that mentioning the file type in the prompt enhances the quality of the response we get.

For the subject area we use:

```
    instruction= "Give the subject area of the file or its field please, in
       nothing more than 3 words"
```

Integration. When querying the GPT model, these components are integrated to form a coherent request, structured as follows:

```
"{pre_prompt_query}
{content}
{prompt_query}{instruction}"
```

Note, the ordering might be different depending on whether we are summarizing, extracting or querying the content.

By utilizing the *pre-prompt*, *prompt*, and *instruction* effectively, we facilitate the querying process and enhance the quality of generated summaries, extracted information and labelling.