Hello, won't you tell me your name?: Investigating Anonymity Abuse in IPFS

Christos Karapapas[1], Iakovos Pittaras[1], George C. Polyzos[1,2], and Constantinos Patsakis[3,4]

[1]Athens University of Economics and Business, Greece
[2]School of Data Science, The Chinese University of Hong Kong, Shenzhen, China
[3]Athena Research Centre, Greece
[4]University of Piraeus
karapapas@aueb.gr, pittaras@aueb.gr, polyzos@acm.org, kpatsak@unipi.gr

Abstract

The InterPlanetary File System (IPFS) offers a decentralized approach to file storage and sharing, promising resilience and efficiency, while also realizing the Web3 paradigm. Simultaneously, the offered anonymity raises significant questions about potential misuse. In this study, we explore methods that malicious actors can exploit IPFS to upload and disseminate harmful content while remaining anonymous. We evaluate the role of pinning services and public gateways, identifying their capabilities and limitations in maintaining content availability. Using scripts, we systematically test the behavior of these services by uploading malicious files. Our analysis reveals that pinning services and public gateways lack mechanisms to assess or restrict the propagation of malicious content. Our findings demonstrate that attackers can exploit the decentralized nature of IPFS and its ecosystem to ensure persistent availability of malicious content while masking their identities. Moreover, we observed instances of this exploitation occurring in practice, further validating the real-world applicability of such attacks.

Keywords: InterPlanetary File System Web3 Security Anonymity

1 Introduction

Web3, often referred to as the *read-write-own* Web, has recently surged in popularity among users and researchers. Although initially presented as a new phase of the World Wide Web, it primarily represents an ideological shift rather than a technological breakthrough. Its main pillars are decentralization, returning data control to users, and the absence of a central authority, treating all users as peers. To achieve this goal, Web3 engulfs technologies such as blockchains, digital currencies, and decentralized identities, all of which have seen rapid growth.

In terms of security, Web3 seeks to mitigate single points of failure, which, in the recent past, have caused substantial disruptions across various technology sectors, including outages of well-known services, leading to widespread paralysis in different technological domains [19]. As with all things in life, Web3 has its dark aspects. The growing interest of users has also drawn the attention of malicious actors toward Web3. Lack of oversight and regulatory authority has led to significant financial losses due to various scams [8, 29]. On the other hand, as the technologies that comprise Web3 are still in their infancy, they suffer from software vulnerabilities, which are exploited by various actors [10]. Another perspective from which Web3 undeniably faces challenges is that of privacy and anonymity, primarily due to its Peer-to-Peer (P2P) nature. While decentralization is a key advantage of Web3, the inherent transparency and traceability of P2P networks often collide with users' privacy expectations [34].

Web3 consists of multiple stacks, each with various protocols that interoperate to deliver user services. These services range from data storage, domain name resolution, and decentralized identities to applications like social media, gaming, and marketplaces. As these protocols and their interconnections evolve, they present potential vulnerabilities that attackers can exploit or leverage. In data storage, there are various protocols, such as InterPlanetary File System (IPFS), Filecoin, Storj, SIA, and others. However, IPFS is widely recognized as one of the most prominent and broadly adopted solutions [11, 38]. Its

open-source nature, content-addressable design, and integration with technologies like Filecoin and public gateways have contributed to its popularity across Web3 applications. Developed by Protocol Labs [32] as an open-source project, it has gained considerable attention in recent years. Notably, companies such as Lockheed Martin have shown interest, even launching an IPFS node into orbit [26]. Furthermore, the growing number of research papers with 'IPFS' in their titles highlights its increasing prominence among researchers, with Semantic Scholar returning more than 800 results for such publications over the past two years. Among the tools that enhance the functionality of IPFS are pinning services, which play a crucial role in maintaining file availability across the network. These services allow users to ensure that specific files remain accessible by hosting them on dedicated nodes, even if the original uploader goes offline. Over time, IPFS has drawn the attention of both malicious actors and security researchers. One study [31] revealed that a notorious botnet exploited its network, while another highlighted that a significant proportion of its nodes are operated by malicious actors [23].

Recent works have shown that malware increasingly leverages benign Internet services to distribute payloads and evade detection. This includes both centralized platforms such as GitHub and Dropbox [40], and large-scale abuse of cloud services like Discord, Mediafire, and Google Drive [5]. Our work extends this threat model to decentralized infrastructures like IPFS, where anonymity, content immutability, and the absence of centralized moderation create an even more permissive environment for abuse. In this paper, we investigate how malicious actors can exploit existing technologies within the IPFS ecosystem to anonymously upload and distribute content. We begin by mapping the current landscape of tools and protocols used to add and access content on IPFS, including pinning services and public gateways. We then design and evaluate practical attacks that leverage these mechanisms to achieve anonymity and persistence within the network. Finally, we explore potential countermeasures to mitigate such exploits. Hence, our main research questions are the following.

RQ1: Do pinning services apply the best know your customer (KYC) practices to allow attribution when malicious content is pinned? (Answered in §4.1)

RQ2: Do pinning services apply any content scanning mechanism to prevent malicious content sharing? (Answered in §4.1)

RQ3: How could an adversary abuse these gaps to share malicious content anonymously? (Answered in §4.1)

RQ4: Is there evidence showing this abuse?(Answered in §4.4)

RQ5: How could an adversary abuse gateways' caching to anonymously share and preserve malicious content online?(Answered in §4.2).

As a result, our research reveals several ways in which IPFS can be abused without providing the necessary tracking mechanisms for perpetrator attribution.

Ethical considerations. While working with live systems, we have taken the necessary measures to ensure that no malware would propagate through the systems and cause any harm. First, any malicious file submitted to IPFS is not executed by any system. However, we acknowledge that once published on IPFS, files may be accessed by third-party systems—including automated scanners or research tools—that might perform dynamic analysis or sandbox execution. To mitigate this risk, all uploaded files either contained benign code flagged as malicious due to simulated behaviors, or were legacy malware samples that no longer pose realistic threats. Secondly, for someone to collect and execute our samples, they must know the CID or monitor all nodes to collect and execute each file. Since the CIDs have not been publicly promoted, the chances of someone collecting all uploaded files and executing them in an unprotected environment are very low. Even in this unlikely scenario, the malicious samples we created just trigger an antivirus without causing actual damage to the system. Finally, the malware we have used from the real world is very well known, and the corresponding URLs have been siphoned, further diminishing the chances of our work impacting any system. We acknowledge that detailing these attack vectors may inadvertently provide insights to malicious actors. However, we believe that openly discussing these issues is necessary to drive improvements in content moderation and security mechanisms within IPFS. Moreover, existing studies have already shown abuse of the IPFS ecosystem. Although no vulnerabilities were directly exploited, we recommend stakeholders in the IPFS ecosystem consider these findings to enhance security measures.

2 Background

The *InterPlanetary File System* (IPFS) [9] is a decentralized file-sharing system focusing on distributed data storage and quick file distribution. IPFS was created and is maintained by Protocol Labs as an open-source project [32]. Unlike traditional file systems, IPFS uniquely identifies files based on their content, assigning each file a distinct *Content IDentifier* (CID). A key IPFS component is *libp2p*, an open-source library of network protocols that includes KAD-DHT, a scalable variant of Kademlia *Distributed Hash Table* (DHT). The KAD-DHT manages three types of mappings, including *Provider Records*, which indicate who hosts specific content; *Peer Records*, which contain information about a specific peer; and *IPNS records*, which link a static address to dynamic data. IPNS names are essentially pointers (IPNS names) to pointers (IPFS CIDs), whereas IPFS CIDs are immutable (because they are derived from the content) pointers to content, Moreover, IPNS names are self-certifying. *Bitswap*, a key component of IPFS, acts as the data exchange and occasionally as a content discovery protocol, using "want-have" and "have" messages for efficient data transfer. IPFS employs Merkle DAGs, a combination of Merkle Trees and *Directed Acyclic Graph* (DAG), to certify the uniqueness of the exchanged data, ensuring that no duplicates are stored. A recent addition to the IPFS ecosystem is the InterPlanetary Network Indexers (IPNI), a centralized version of the DHT designed to efficiently index provider records. It serves primarily large content providers and complements the existing DHT by focusing solely on provider record management. Additionally, Protocol Labs and other companies offer services that offer public gateways, allowing users to access the content of the IPFS network without maintaining a node.

In IPFS, each peer manages a network of active connections, known as the *swarm*, which typically ranges from 600 connections (the *low water mark*) to 900 (the *high water mark*). When a user requests a file from the IPFS network, the Bitswap protocol is triggered. It sends a message to the user's swarm peers in the format `want-have <root CID>` [13]. Peers in the swarm individually check whether they have the specified CID locally. If a peer possesses the requested content, it responds with a `have` message. If no response is received within 1 second, the process is handed over to the DHT, which operates in two stages. Initially, the process searches for the Provider Record, which contains the Peer ID, which stores the content for the requested CID. Subsequently, it searches for the Peer Record, which shows how the Peer ID is linked to a network address. Once this process is finalized, Bitswap is reactivated to facilitate data exchange with the peer hosting the content [38].

3 Adding a File to IPFS

There are several ways to add a file to IPFS. In this section, we explore different methods and their respective modi operandi. Additionally, we examine the information about the original uploader that can be retrieved for each method and the duration that the files remain online.

IPFS Node For the average user, the primary option for connecting to the IPFS network is the IPFS Desktop application, which supports the most operating systems and includes the functionality of an IPFS node within a user-friendly graphical interface. There is also a command-line version available called Kubo. When a new node connects to the network, if it has a public IP, it is characterized as a DHT Server. Otherwise, e.g., being behind NAT, it defaults to a DHT Client. This is managed by a mechanism called *Autonat*. This distinction ensures that DHT Servers store and provide data, while DHT Clients only request it, optimizing the network's efficiency [38]. When a user wishes to publish a file to IPFS, the process involves splitting the original file into smaller chunks, typically 256 KB. Each chunk is assigned a unique CID and organized into a Merkle DAG added to IPFS. Consequently, two types of records are created in the DHT, with one record stored across a set of 20 specific nodes and the other stored across a different set of 20 nodes. The first type, the Provider Record, indicates who is hosting the file and includes two additional parameters: the republish interval (12 hours by default), which assigns new peers if the original 20 nodes go offline, and the expiration interval (24 hours by default), which verifies that the publisher is still online. The second type is the Peer Record, which maps the peer to its physical address. From the above, it is clear that when a file is added to IPFS, the file itself is not replicated, instead only links pointing to the uploader are created. Upon file addition, the IPFS node automatically pins the file to the original uploader's node, ensuring its availability while the uploader remains online. Replication occurs only if another user requests the file, resulting in it being stored in their cache. Should the original uploader disconnect from the network, the file's availability relies entirely on the cache of interested users. The aforementioned process is depicted in Figure 1. It is also worth mentioning that the Brave Browser natively supports the use of IPFS in conjunction with a local node [15], yet earlier versions provided the ability to add files via Public Gateways.

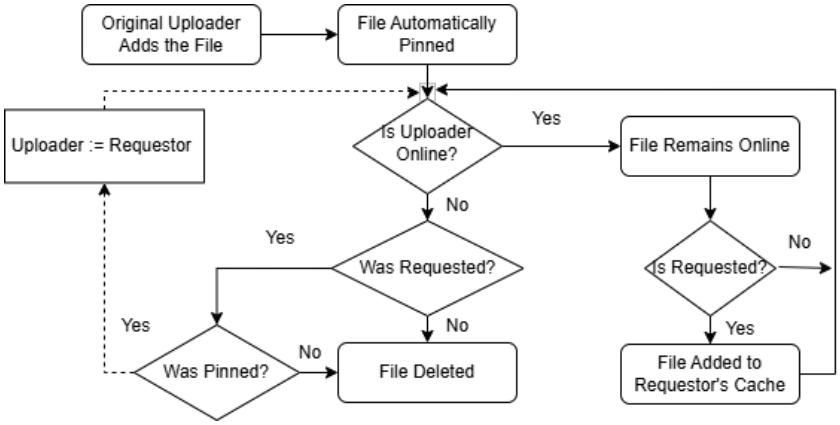

Figure 1: The File Lifecycle In IPFS.

Pinning Services IPFS, according to its design principles, does not provide a mechanism to ensure that files added to the network remain online if the original uploader deletes them or disconnects from the network. Files are primarily cached by requesters to ensure their availability to other nodes. The more popular a file is, the higher its chances of staying online for an extended period. Additionally, every IPFS node runs a garbage collector to free up storage space. As a result, cached files are periodically removed, leading some files to disappear from the network over time [11]. To prevent the garbage collector from removing a file, the user must pin it. Pinning can be categorized into two types: local pinning, where the user configures their node to retain the file, though it will fade once the node disconnects from the network; and remote pinning, where an external provider takes the responsibility to ensure that the file remains pinned [18].

A plethora of pinning services is available, with Pinata, Filebase, Fleek, and 4EVERLAND being among the most popular. These platforms offer user-friendly graphical interfaces for adding files to the IPFS network, simplifying the process for the average user. Moreover, they provide free storage space for uploading and pinning files, making them accessible to a wide range of users. Once added, the files can be retrieved through public gateways, which act as HTTP access points to the IPFS network.

Although Web3.Storage and NFT.Storage [28] are not strictly classified as pinning services, their functionality closely resembles traditional pinning solutions, so we include them in this section for completeness. These open-source services, developed by Protocol Labs, are designed to store general and NFT-related data, respectively, in the Web3 era. Both services operate decentralized, leveraging IPFS for content addressing and Filecoin for long-term data preservation rather than offering a pinning service. Web3.Storage is notably free for the community, while NFT.Storage operates under a paid model. NFT.Storage was excluded from further experiments, as it specializes exclusively in NFT metadata storage, which falls outside the scope of our analysis focusing on general-purpose file uploads.

3.1 Public Gateways

Public gateways act as HTTP entry points to the IPFS network, bridging the Web2 and Web3 ecosystems. They process HTTP requests containing CIDs and relay them to an IPFS node, enabling broader access to the network through conventional Web protocols. Although users cannot directly upload files through a gateway, indirect methods enable this functionality, justifying their classification in this section. Furthermore, the HTTP servers underpinning these gateways leverage caching mechanisms, most commonly the Least Recently Used (LRU) strategy which optimizes performance and user experience by evicting the least recently accessed content when the cache reaches its capacity [38]. Based on the above, it is evident that even if the original uploader disconnects from the IPFS network, the file may remain accessible, cached by gateways, with its persistence primarily influenced by its popularity. During the preparation of this study, we identified 10 online gateways [4]. Using the fingerprinting tool WhatWeb [21], we found that nine gateways utilize either Nginx software or Cloudflare proxies, which employ the LRU caching strategy to manage content efficiently.

The fact that public gateways serve as a bridge between the traditional Web and the P2P ecosystem of IPFS makes them very crucial for launching and countering several attacks. For instance, an adversary may host a phishing page on IPFS; however, the content must be rendered from the victim's browser. Thus, the bridge fetches the content from IPFS and brings it to the Web. It must be noted that while

Table 1: Registration Requirements & Free Storage for Pinning Services.

Pinning Service	URL	KYC	Temp Mail Accepted	Free Storage	Registered Country	DMCA Compliant
Pinata	`https://pinata.cloud`	E-mail	✓	1 GB	USA	✓
Filebase	`https://filebase.com`	E-mail	✓	5 GB	USA	✓
Fleek	`https://fleek.co`	E-mail	✓	5 GB	USA	✓
Web3.Storage	`https://web3.storage`	Credit Card	✓	5 GB	USA	-
4EVERLAND	`https://4everland.org`	Crypto Wallet	N/A	5 GB	AUS	✓

there is no official deletion mechanism for IPFS [30], some public gateways follow blocking mechanisms to prevent specific content from reaching the Web [35]. Nevertheless, not all gateways follow the same blocking mechanism and, of course, this does not remove the content from IPFS.

4 Exploiting IPFS for Anonymity: Attack Scenarios

The anonymity offered by IPFS can be exploited by malicious actors. In this section, we analyze how attackers leverage methods discussed in Section 3 to achieve anonymity, presenting and evaluating two distinct attack scenarios. The code is available at `https://github.com/mmlab-aueb/ipfs-anonymity` for reproducing the experiments.

4.1 The Pinning Service Attack

Pinning services ensure that a file remains online. Therefore, it is logical to consider that an attacker could exploit these services to upload a file and guarantee its availability. However, since our focus is on evaluating the level of anonymity, we first examine the information each pinning service requires from users to allow file uploads, i.e., the Know Your Customer (KYC) procedure. We selected Pinata, Filebase, Fleek, Web3.Storage, and 4EVERLAND based on a systematic Internet search. Specifically, we performed Google queries such as "top IPFS pinning services" and "most popular IPFS pinning services," identifying the services most frequently mentioned in developer documentation, technical articles, and community discussions. Academic literature specifically evaluating IPFS pinning services remains limited, further justifying the need to consult current developer ecosystems and real-world service availability. Besides the selected providers, our search also highlighted Infura and Temporal. However, Infura currently restricts access to pre-qualified customers [27], and Temporal appears to have discontinued operations. Thus, our study focuses exclusively on active and publicly available services, realistically representing the infrastructure accessible to potential anonymous attackers.

The Pinata, Fleek, and Filebase services require an email address for user registration. To achieve higher levels of anonymity, we attempted to use a temporary email service. A temporary email is a disposable email address that allows users to receive emails for a short period, often used to maintain anonymity or avoid spam during registration processes. During December 2024 and January 2025, we tested the registration process on Pinata, Fleek, and Filebase using email addresses generated by the service TempMail (`https://temp-mail.org`). Both Pinata and Fleek accepted the first temporary email we generated, allowing us to create accounts successfully. After four attempts with different temporary email addresses, Filebase accepted the registration, suggesting that its filtering against disposable emails may be incomplete. In all three cases, the platforms required us to verify the email address using a one-time password (OTP). 4EVERLAND, on the other hand, does not use email-based registration but instead requires a cryptocurrency wallet. Using Metamask, we successfully created an account on the platform, noting that even for creating the Metamask wallet, no email was needed. Finally, while Web3.storage accepted the temporary registration email, uploading files required linking a payment account, even though the platform also offers a free plan. This suggests that, although temporary emails are allowed, the payment account requirement serves as an additional verification step for users, limiting its suitability for fully anonymous abuse scenarios. Table 1 presents a summary of these findings.

To simulate malicious behavior, we developed a Python script packaged into a Windows executable using PyInstaller [1]. It mimicked keylogging, dummy process injection, basic file manipulation, and failed network connections. The file was safe by design, yet flagged by multiple antivirus engines on VirusTotal [2] due to behavioral heuristics. No harmful payload or external communication was included. To ensure unique Content Identifiers (CIDs), we created a distinct version of each script for each pinning service under evaluation. One of the key questions explored in this section is how pinning services handle files clearly marked as malicious, aiming to better replicate the perspective and actions of a potential

attacker. In addition to the simulated malware, we also tested uploading known deprecated malware, specifically the WannaCry ransomware, to the pinning services. The result was identical: the file was successfully uploaded, and its CID was generated. Furthermore, we confirmed its accessibility through the public gateways. Notably, all files, including WannaCry, were immediately accessible, highlighting the absence of mechanisms in public gateways to evaluate the maliciousness of uploaded content. This raises significant concerns about the potential misuse of the IPFS network.

As previously discussed, in IPFS, the physical address of the node hosting a file can be identified. However, when files are hosted by pinning services, attackers are not concerned about their own address being exposed. The only potential exposure point is during the interaction with the pinning service's website for registration and file upload. To mitigate this risk, an attacker could use a public network or leverage the Tor [14] network to enhance their anonymity prior to registering and uploading files to the pinning services. Since many services implement protections that restrict access via Tor, we conducted a series of tests to verify the feasibility of using Tor to access these services. Our tests confirmed that files could be successfully uploaded, and the recorded IP address differed from our actual address, ensuring the attacker's anonymity.

It is important to note that visitors to these files, once uploaded by the attacker, may include either unsuspecting users who were targeted by phishing [35] or malware campaigns, or, in CyberCrime-as-a-Service scenarios [22], collaborators of the attacker, such as affiliates. Even in the latter case, leveraging the Tor network can effectively mitigate the risk of exposing their identities or the nature of their activities.

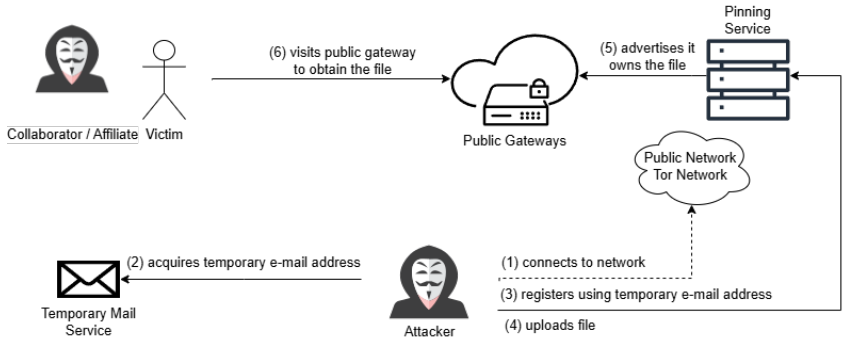

Figure 2: Design of the "Pinning Service Attack".

Figure 2 presents the steps that a malicious actor must follow to execute the "Pinning Service Attack". It allows the attacker to leverage the Tor network for anonymity and anonymously upload files to IPFS. By utilizing pinning services, the attacker ensures that uploaded files remain persistently online.

4.2 The Public Gateway Attack

As mentioned, Public Gateways of IPFS do not provide a direct method for uploading a file to the network. However, their caching might indirectly serve as a pinning service, providing file availability. In this section, we initially examine whether and for how long a file remains cached.

To better understand this phenomenon, we conducted a systematic experiment focusing on caching behavior across multiple gateways. The methodology we adopted is as follows. From the 10 gateways identified in Section 3.1, we selected five based on their strong association with well-known Web companies (e.g., Pinata, Infura), official status within the IPFS ecosystem, reputation, and service quality. Specifically, we chose (a) `ipfs.io` (the official gateway maintained by Protocol Labs), (b) `gateway.pinata.cloud`, (c) `infura-ipfs.io`, (d) `flk-ipfs.xyz` and (e) `4everland.io`. For each selected gateway, we created four different files, resulting in 20 different files. First, we wanted each gateway to have different files to avoid cross-caching scenarios. Second, for each of these, we created four different files corresponding to the four time scenarios we are studying: 1 hour, 6 hours, 12 hours, and 24 hours. We use these intervals to request the respective files from the gateways to understand how popular a file needs to be to remain cached. Subsequently, we used an IPFS node to add the files, ensuring our node ran as a DHT server. Then, to confirm that all the gateways cached all files, we sent up to four requests per file to verify their caching status. The four requests were performed in a negligible amount of time, less than five minutes, and the files became available. After successfully ensuring that all files were cached across the gateways, we disconnected the node from the network, leaving the gateways as the sole source of file hosting. The

latter allows us to isolate the role of gateway caching in maintaining file availability independent of the original node. By doing so, we could analyze how the caching mechanisms of public gateways sustain file accessibility over time.

We automated the process of sending requests to the gateways based on the aforementioned periods and recorded the responses for more than three days. The results indicate that caching duration varies significantly between gateways, with some maintaining availability longer than others, which could be attributed to differences in caching strategies or the relative popularity of each gateway. Figure 3 illustrates the ratio $\frac{\nu}{5}$ per hour, where ν represents the number of gateways caching our files at a given time across the different time scenarios. As depicted, two out of the five gateways removed our files from their cache shortly after 16 hours, while the remaining three continued to retain them online. For ethical reasons, we refrain from disclosing which ones retained or removed the files.

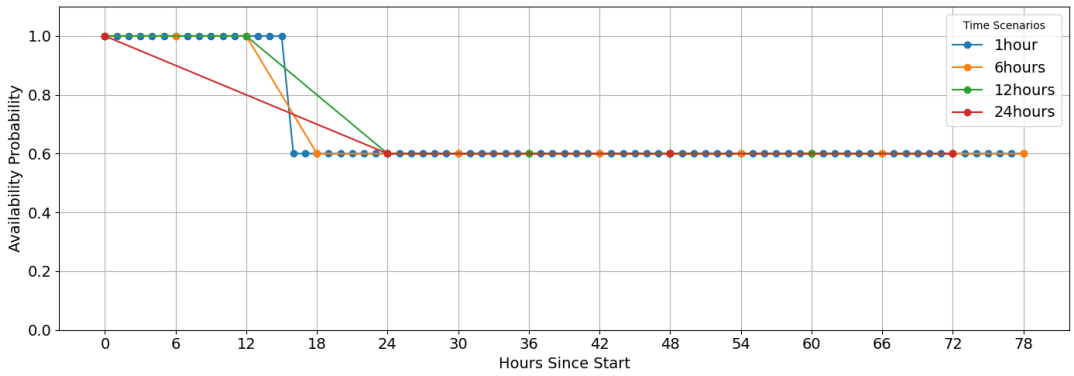

Figure 3: Time-Dependent File Availability Analysis.

In conclusion, we have demonstrated that a malicious actor could potentially exploit Public Gateways to maintain files on the IPFS network anonymously. The process involves first uploading the files to the IPFS network and generating artificial traffic by repeatedly requesting these files. This ensures that the Public Gateways cache the files. Once the files are cached, the actor can sustain their availability by periodically sending requests for the files, preventing them from being removed from the cache due to inactivity. This approach allows the actor to leverage the distributed infrastructure of Public Gateways to maintain file availability while preserving anonymity, eliminating the need for a dedicated pinning service. At this point, it should be noted that during the attack, the attacker only risks revealing their

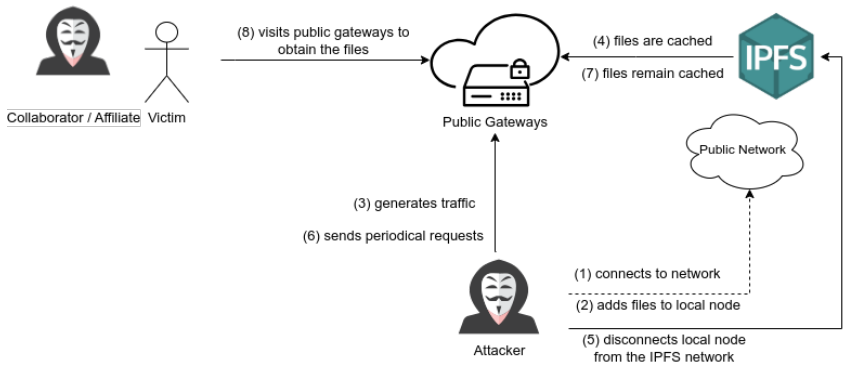

Figure 4: Design of the "Public Gateway Attack".

physical address while uploading the files via the local node. As previously mentioned, this process requires minimal time, significantly reducing the exposure window for the attacker. Additionally, the attacker could perform this step through a public network to further obscure their physical location. The subsequent periodic requests to the public gateways can also be accomplished through a public network or Tor. Additionally, the attacker could utilize a botnet under their control to generate artificial traffic towards the files without revealing their identity. By distributing requests across multiple geographically dispersed nodes, the botnet obscures the origin of the traffic, making it significantly harder to trace back

to the attacker. Note that in the past, the IPFS network has been a victim of such botnet activity [31]. A step-by-step implementation of the attack is illustrated in Figure 4.

4.3 Double Extortion Attack

Typically, ransomware attacks encrypt the victim's files and demand a ransom to be paid to hand over the decryption key. Nevertheless, modern organizations have invested in backup systems that limit the damages of a potential ransomware attack, significantly decreasing the amount of ransom they would be willing to pay. As a countermeasure, ransomware gangs siphon sensitive data to their premises, threatening their victims by leaking the data and creating what is often called a "double extortion".

The siphoning of the data can be performed in multiple ways, however, methods like DNS tunneling, while effective, can be very slow. Therefore, ransomware gangs tend to abuse cloud service providers to upload their "loot". For example, the notorious Conti group used RClone to upload data to multiple cloud storage providers [20]. With IPFS and the poor KYC practices of pinning services, ransomware gangs can have another more robust option. They may harvest sensitive information from the infected hosts and upload them to IPFS through pinning services. Beyond exploiting KYC to gain the necessary storage, ransomware gangs may also exploit whitelisted domains and the lack of content takedown mechanisms. Note that cloud service providers respond to takedown notices, e.g., the victim notifies the cloud service provider that leaked sensitive data are hosted and must be taken down. However, pinning services cannot remove content from the IPFS once it has been uploaded. Although pinning services comply with DMCA policies and can remove a pinned file from their hosted storage, this does not translate into the deletion of the file from the IPFS network. The decentralized nature of IPFS makes this nearly impossible, while the existence of public gateways, many of which do not adhere to the badbits list (as mentioned in 4.4), further complicates takedown efforts. Figure 5 illustrates this abuse scenario.

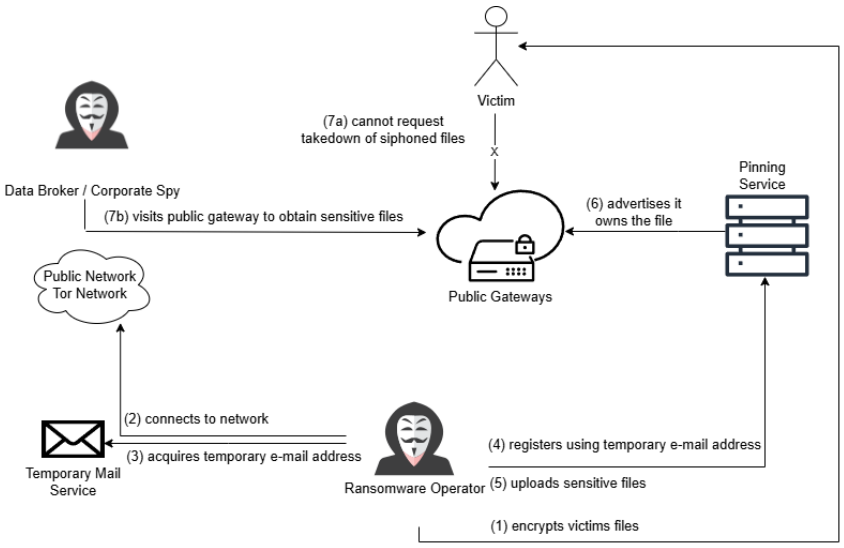

Figure 5: Design of the "Double Extortion Attack".

4.4 Real-Life Evidence of Malicious Exploitation

While previous work such as [35] investigated the presence of malicious or illegal content across the IPFS network, our approach specifically targets pinning services, i.e., entities that intentionally maintain long-term availability of hosted content. By focusing on CIDs advertised by major pinning providers, our analysis offers a more precise view into deliberate, persistent misuse of the IPFS ecosystem and links it directly to infrastructures that facilitate anonymity and permanence.

We utilized `ipni-cli` [3] to monitor CIDs advertised by Pinata, Filebase, and Fleek pinning services on the `cid.contact` indexer for 24 hours. For all providers, we repeatedly executed the following command:

```
ipni ads get --ai=<provider addr> --head
```

This command retrieves information about the latest advertisement from the specified provider, including the number of CIDs it contains. Once we obtained this information, we proceeded to extract the actual CIDs using:

```
ipni random <provider addr>
```

With the parameter `n`, this command returns `m` CIDs from a random selection of the most recent `n` advertisements. By setting `n=1`, we ensured that the selection always targeted the most recent advertisement. Since the previous command had already provided us with the exact number of CIDs, we could request all of them at once. This approach enabled us to systematically retrieve all hashes from every advertisement recorded since the beginning of the experiment. By continuously executing these queries and storing the results, we effectively built a historical record of all advertisements and their associated CIDs from each provider. During the 24-hour interval, we collected **(i)** 1,124,780 CIDs from Pinata, **(ii)** 718,578 from Filebase, and **(iii)** 339,684 from Fleek. For each of these, we standardized the format of the CIDs to match the entries in the `Bad Bits Denylist` [33], ensuring compatibility for an accurate comparison. The Bad Bits Denylist is a list maintained by Protocol Labs, updated upon email recommendations to filter undesirable files, such as malware, phishing content, or copyright-infringing materials. Note that the list is enforced on the public gateways operated by Protocol Labs but is advisory for all other nodes within the IPFS network. By matching the monitored CIDs against the entries in the denylist, we discovered that within 24 hours, the pinning services advertised five CIDs included in the Bad Bits Denylist. It is worth mentioning that one of these CIDs was advertised by all three services, while two were common to two services. We consider the presence of these blocked CIDs –and even more so their simultaneous advertisement on the same day by multiple pinning services– a strong indication of malicious actors' organized exploitation of the anonymity provided by pinning services. Finally, we managed to retrieve three of them, discovering that one was a JavaScript file involved in a Bank of America phishing scam, the second was a login phishing webpage targeting a Korean webmail service, and the third was an image, likely used for malicious purposes.

5 Related Work

Research has shown that malware increasingly abuses centralized Web and cloud platforms for infrastructure, persistence, and evasion. Yao et al. [40] propose Marsea, a concolic execution engine that detects malware interaction with benign Web applications such as GitHub and Dropbox, revealing how these services are repurposed for malicious use. At a broader scale, Allegretta et al. [5] analyze threat intelligence from 36 vendors and identify over 22,000 abused benign domains, including services like Discord and Google Drive, used to distribute malware. These works demonstrate that even trusted, centrally managed services are vulnerable to abuse. In this work, we show that decentralized infrastructures like IPFS introduce new and arguably more permissive abuse surfaces, due to their inherent anonymity, lack of content moderation, and resistance to takedown.

In recent years, Web3 has emerged as a new paradigm for the Internet, prioritizing user anonymity and privacy. These features are especially significant as concerns about user privacy and tracking escalate. However, numerous studies indicate that these features are often compromised. Kshetri [25] highlights several vulnerabilities within Web3 and the metaverse, particularly the extensive data collection and exposure of personal and sensitive data due to numerous security breaches on Web3. Furthermore, the author points out that anonymity can be compromised via the traceability of blockchain transactions on Web3 platforms, potentially linking personal identities and actions to public transaction records.

On the other hand, other studies focus on how anonymity and privacy are compromised on Web3. Wang et al. [39] explore how Web3 social platforms, such as friend.tech [17], impact user privacy and anonymity. In particular, they identified that the integration between Web3 and legacy Web2 platforms could significantly undermine Web3 anonymity and lead to privacy leakage. This occurs because user actions on Web2 platforms can be associated with accounts on Web3 platforms since these actions are immutably written on blockchains. Then, the recorded actions can be linked and traced back to the users. To address these problems, the authors argue that a balanced approach between transparency and privacy in Web3 is needed. Additionally, Torres et al. [37] focus on how wallets and Decentralized Applications (DApps) manage user data. The authors conclude that current privacy measures are insufficient, highlighting that Web3 applications, particularly wallets, often expose sensitive user data, such as wallet addresses. This exposure directly contradicts the foundational privacy promises of Web3 by compromising user anonymity and privacy.

A central element of Web3 and a core focus of our study are distributed file systems, with IPFS being the most prominent. Previous research has demonstrated that IPFS can be exploited by malicious actors in various domains. For example, studies have shown its use in Malware as a Service systems [22], while others have reported the presence of phishing files or copyright violations within the IPFS network [35]. Moreover, IPFS also has some privacy violations. In particular, Balduf et al. [7] showcase a privacy attack on the IPFS network by leveraging the Bitswap protocol and introducing a set of attack vectors. The authors state that every IPFS node is susceptible to each of the introduced attacks, and moreover, they succeed in exploiting it by deploying a number of nodes with extended connectivity to passively monitor the Bitswap channel and demonstrate their attack methodology by discovering the PeerId of the public IPFS HTTP gateways.

In addition to attackers, security analysts can leverage Bitswap's privacy shortcomings. Son et al. [36] propose *IF-DSS*, a digital forensics investigation framework for Decentralized Storage Services (DSSs). They analyze the most critical DSSs from the point of view of digital forensics and apply the proposed framework to IPFS. To collect appropriate and sufficient data, they separate them into those that exist on the local side as well as remotely. Finally, they suggest tackling the dissemination of illegal material in three steps: (i) Content filtering, i.e., blacklisting of the inappropriate content, (ii) stop content sharing, i.e., turn the node from server to client, and finally, (iii) shutting down the node.

On the other hand, some works try to enhance IPFS privacy. Katsantas et al. [24] focus on hiding the identity of content on IPFS by using only hash functions. The authors aim to prevent intermediaries from detecting the retrieved contents without relying on trusted third parties. Furthermore, Daniel et al. [12] point out that as IPFS follows the ICN paradigm, a client requests content directly rather than visiting an address. Thus, Bitswap queries all the client's neighbors for content, resulting in the client's interest leaking. Aiming to reduce interest leakage, the authors propose three privacy-enhanced standards for content discovery. By using these protocols, on the one hand, the level of privacy of the client is improved, but that of the provider is reduced. More specifically, they propose a solution using bloom filters and `Bloom-Swap`, a solution using bloom filters in which the provider sends its inventory to the client, and he, in turn, checks locally whether the requested content is a Bloom Filter member to ask the block directly. `PSI-Swap`, which uses Private Set Intersection (PSI), reduces and improves privacy levels on the provider's side as well. Finally, the `BEPSI-Swap`, which combines the two previous ones, improves the efficiency of PSI-Swap, at the cost of making PSI probabilistic. The authors then implement a proof of concept of the proposed protocols and study them from the security and efficiency perspectives.

6 Countermeasures & Conclusions

The decentralized nature of the technologies we study, combined with the fact that the majority of the software is open-source, makes enforcing rules for implementing countermeasures challenging. From the perspective of pinning services, KYC practices must become stricter. Measures such as filtering temporary emails, implementing blockchain-based identity systems, e.g., cryptocurrency wallets with benign transaction history, applying stricter criteria for users operating through Tor networks, enabling content scanning mechanisms, and adhering to a centralized deny list like Bad Bits should be enforced. Public gateways act as bridges for Web2 users to access the Web3 ecosystem. For the average user, requiring a blockchain-based identity would deter them from utilizing these gateways. However, all gateways could be required to comply with the Bad Bits, a policy currently enforced only on gateways managed by Protocol Labs. Moreover, even if a CID is listed on the Bad Bits Denylist, a malicious actor can circumvent it by simply choosing an alternative chunking size when adding the file to IPFS (**RQ5**). This approach generates a different CID that is not associated with the blacklisted one [35], making content filtering on gateways significantly more challenging. In this study, we examined the vulnerabilities of IPFS pinning services and public gateways, highlighting how malicious actors can exploit their anonymity features or lack of proper KYC policies to share undesirable content. By implementing and testing two distinct attack methodologies, we demonstrated not only their feasibility (**RQ3**) but also observed instances of malicious activity occurring within the IPFS ecosystem (**RQ4**). Our findings reveal critical issues, including the lack of robust KYC practices in pinning services (**RQ1**), insufficient content filtering mechanisms (**RQ2**), and the challenges posed by the decentralized and open-source nature of the IPFS ecosystem. These gaps enable attackers to take advantage of the anonymity features of the system while avoiding accountability. Since current KYC practices in pinning services can be easily bypassed, the use of stricter measures, of even the consideration of blockchain-based identity verification methods, such as zero-knowledge proofs (ZKPs), e.g., zkLogin [6], would allow users to verify their legitimacy without

exposing their full identity.

It should be stressed that the decentralized nature of IPFS raises significant legal and regulatory challenges, particularly in the enforcement of content moderation and compliance with existing digital laws. While platforms operating in centralized environments are bound by regulations such as the Digital Services Act (DSA) [16], decentralized systems like IPFS lack clear accountability structures. This creates a regulatory gap that malicious actors can exploit to distribute illicit content while avoiding legal repercussions. One of the main concerns is jurisdictional ambiguity. Since IPFS content is hosted on a distributed network of peers, it is often unclear which jurisdiction has the authority to enforce takedown requests or prosecute offenders. This is especially true on platforms like IPFS, where there is no deletion mechanism and data ownership is not always known. Pinning services, many of which operate in different countries with varying legal requirements, further complicate the enforcement process.

However, this sparks the debate surrounding IPFS security and other such platforms regarding the trade-off between privacy and censorship resistance. While decentralization offers increased resilience against state-sponsored censorship, it also enables unmoderated content proliferation, including, but not limited to, extremist propaganda, child sexual abuse material, and malware distribution. The ability of malicious actors to exploit anonymity for illegal activities creates a dilemma where content moderation mechanisms must be introduced without undermining the fundamental principles of decentralized storage. Strengthening the security of IPFS and the surrounding ecosystem is essential not only to prevent its misuse but also to promote its adoption as a reliable and privacy-preserving tool for decentralized file sharing, which is fundamental to the Web3 paradigm. Thus, future research could focus on the development of automated tools to detect malicious CIDs in a decentralized and scalable way. Another approach would be decentralized content moderation, where community-driven flagging mechanisms allow for voluntary filtering rather than direct deletion. Likewise, user-driven reputation systems for pinning services and nodes could help differentiate legitimate operators from malicious ones. By assigning trust scores to nodes based on their activity and compliance with community standards, users could make informed choices about which nodes to trust for content retrieval and caching.

Acknowledgements

This work was supported in part by the European Commission under the Horizon Europe Programme, as part of the project SafeHorizon (Grant Agreement no. 101168562). The content of this article does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the authors.

References

[1] `https://pyinstaller.org/`.

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

[3] ipni-cli. `https://github.com/ipni/ipni-cli`.

[4] Public gateway checker. `https://ipfs.github.io/public-gateway-checker/`.

[5] Mauro Allegretta, Giuseppe Siracusano, Roberto González, Marco Gramaglia, and Juan Caballero. Web of shadows: Investigating malware abuse of internet services. *Computers & Security*, 149:104182, 2025.

[6] Foteini Baldimtsi, Konstantinos Kryptos Chalkias, Yan Ji, Jonas Lindstrøm, Deepak Maram, Ben Riva, Arnab Roy, Mahdi Sedaghat, and Joy Wang. zklogin: Privacy-preserving blockchain authentication with existing credentials. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, page 3182–3196, New York, NY, USA, 2024. ACM.

[7] 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. IEEE, 2022.

[8] Massimo Bartoletti, Stefano Lande, Andrea Loddo, Livio Pompianu, and Sergio Serusi. Cryptocurrency scams: analysis and perspectives. *IEEE Access*, 9:148353–148373, 2021.

[9] Juan Benet. IPFS-content Addressed, Versioned, P2P File System. *arXiv preprint arXiv:1407.3561*, 2014.

[10] Catherine Carpentier-Desjardins, Masarah Paquet-Clouston, Stefan Kitzler, and Bernhard Haslhofer. Mapping the defi crime landscape: An evidence-based picture. *arXiv preprint arXiv:2310.04356*, 2023.

[11] Erik Daniel and Florian Tschorsch. IPFS and friends: A qualitative comparison of next generation peer-to-peer data networks. *IEEE Communications Surveys & Tutorials*, 24(1):31–52, 2022.

[12] Erik Daniel and Florian Tschorsch. Privacy-enhanced content discovery for Bitswap. In *2023 IFIP Networking Conference*, pages 1–9, 2023.

[13] 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.

[14] Roger Dingledine, Nick Mathewson, Paul F Syverson, et al. Tor: The second-generation onion router. In *USENIX security symposium*, volume 4, pages 303–320, 2004.

[15] Trinh Viet Doan, Yiannis Psaras, Jörg Ott, and Vaibhav Bajpai. Toward decentralized cloud storage with ipfs: opportunities, challenges, and future considerations. *IEEE Internet Computing*, 26(6):7–15, 2022.

[16] European Union. Regulation (EU) 2022/2065 of the European Parliament and of the Council of 19 October 2022 on a Single Market For Digital Services and amending Directive 2000/31/EC (Digital Services Act). `https://eur-lex.europa.eu/eli/reg/2022/2065/oj/eng`, 2022.

[17] friend.tech/. `https://www.friend.tech/`.

[18] Barbara Guidi, Andrea Michienzi, and Laura Ricci. Data persistence in decentralized social applications: The IPFS approach. In *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–4. IEEE, 2021.

[19] Grant Hatchimonji. `www.techtarget.com/whatis/feature/8-largest-IT-outages-in-history`, 2024.

[20] Michael Heller. A conti ransomware attack day-by-day. `https://news.sophos.com/en-us/2021/02/16/conti-ransomware-attack-day-by-day/`, 2021.

[21] Andrew Horton. Whatweb. `https://github.com/urbanadventurer/whatweb`.

[22] Christos Karapapas, Iakovos Pittaras, Nikos Fotiou, and George C Polyzos. Ransomware as a service using smart contracts and IPFS. In *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–5. IEEE, 2020.

[23] Christos Karapapas, George C. Polyzos, and Constantinos Patsakis. What's Inside a Node? Malicious IPFS Nodes Under the Magnifying Glass. In Norbert Meyer and Anna Grocholewska-Czuryło, editors, *ICT Systems Security and Privacy Protection*, pages 149–162, Cham, 2024. Springer Nature Switzerland.

[24] Thomas Katsantas, Yannis Thomas, Christos Karapapas, and George Xylomenos. Enhancing IPFS privacy through triple hashing. In *2024 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6, 2024.

[25] Nir Kshetri. Privacy violations, security breaches and other threats of Web3 and the metaverse. 32nd European Regional ITS Conference, Madrid 2023: Realising the digital decade in the European Union – Easier said than done? 277993, International Telecommunications Society (ITS), 2023.

[26] Lockheed Martin. `https://www.lockheedmartin.com/en-us/news/features/2024/smartsat-equipped-satellite-uploads-new-mission-on-orbit.html`, 2024.

[27] MetaMask. `https://docs.metamask.io/services/reference/ipfs/`.

[28] NFT.Storage. `https://nft.storage/`.

[29] Constantinos Patsakis, Fran Casino, Nikolaos Lykousas, and Vasilios Katos. Unravelling ariadne's thread: Exploring the threats of decentralised dns. *IEEE Access*, 8:118559–118571, 2020.

[30] Eugenia Politou, Efthimios Alepis, Constantinos Patsakis, Fran Casino, and Mamoun Alazab. Delegated content erasure in IPFS. *Future Generation Computer Systems*, 112:956–964, 2020.

[31] Silvia Pripoae. Looking Into the Eye of the Interplanetary Storm, 2020.

[32] Protocol Labs. `https://www.protocol.ai/`.

[33] Protocol Labs. Bad Bits Denylist. `https://badbits.dwebops.pub/`.

[34] Dan Sheridan, James Harris, Frank Wear, Jerry Cowell Jr, Easton Wong, and Abbas Yazdinejad. Web3 challenges and opportunities for the market. *arXiv preprint arXiv:2209.02446*, 2022.

[35] Saidu Sokoto, Leonhard Balduf, Dennis Trautwein, Yiluo Wei, Gareth Tyson, Ignacio Castro, Onur Ascigil, George Pavlou, Maciej Korczynski, Björn Scheuermann, and Michal Król. Guardians of the galaxy: Content moderation in the interplanetary file system. In Davide Balzarotti and Wenyuan Xu, editors, *33rd USENIX Security Symposium, USENIX Security 2024, Philadelphia, PA, USA, August 14-16, 2024*. USENIX Association, 2024.

[36] Jihun Son, Gyubin Kim, Hyunwoo Jung, Jewan Bang, and Jungheum Park. IF-DSS: A forensic investigation framework for decentralized storage services. *Forensic Science International: Digital Investigation*, 46:301611, 2023.

[37] Christof Ferreira Torres, Fiona Willi, and Shweta Shinde. Is your wallet snitching on you? an analysis on the privacy implications of web3. In *Proceedings of the 32nd USENIX Conference on Security Symposium*, SEC '23, USA, 2023. USENIX Association.

[38] 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.

[39] Bin Wang, Tianjian Liu, Wenqi Wang, Yuan Weng, Chao Li, Guangquan Xu, Meng Shen, Sencun Zhu, and Wei Wang. The Illusion of Anonymity: Uncovering the Impact of User Actions on Privacy in Web3 Social Ecosystems, 2024.

[40] Mingxuan Yao, Jonathan Fuller, Ranjita Pai Kasturi, Saumya Agarwal, Amit Kumar Sikder, and Brendan Saltaformaggio. Hiding in plain sight: An empirical study of web application abuse in malware. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 6115–6132, 2023.