Sybil Attack Strikes Again: Denying Content Access in IPFS with a Single Computer

Thibault Cholez, Claudia-Lavinia Ignat

Sybil Attack Strikes Again: Denying Content Access in IPFS with a Single Computer

Thibault Cholez
Claudia-Lavinia Ignat
thibault.cholez@loria.fr
claudia.ignat@inria.fr
Université de Lorraine, CNRS, Inria, LORIA
Nancy, France

ABSTRACT

The Distributed Hash Table (DHT) architecture is known to be a very efficient way to implement peer-to-peer (P2P) computer networks. However, the scientific literature also proved that DHT functioning in P2P networks can be easily disrupted by a single entity controlling many peers, known as the Sybil Attack. Various defensive mechanisms are known to prevent such attacks, or at least hinder them. The current study evaluates the resiliency of the InterPlanetary File System (IPFS) P2P network to a legacy Sybil Attack. We show that, surprisingly, IPFS does not implement basic defense mechanisms, allowing the most simple attack from a single computer to easily take the control of any DHT entry. A practical use of this attack is to almost entirely deny access to a given content on the network. Thus we provide some recommendations to quickly remediate this vulnerability.

CCS CONCEPTS

• **Networks** → **Peer-to-peer protocols**; **Denial-of-service attacks**; • **Security and privacy** → **Distributed systems security**.

KEYWORDS

Sybil Attack, IPFS, Distributed Hash Table, Denial of Service, P2P network

ACM Reference Format:
Thibault Cholez and Claudia-Lavinia Ignat. 2024. Sybil Attack Strikes Again: Denying Content Access in IPFS with a Single Computer. In *The 19th International Conference on Availability, Reliability and Security (ARES 2024), July 30-August 2, 2024, Vienna, Austria.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3664476.3664482

1 INTRODUCTION

Since Chord in 2001 [28], the Distributed Hash Table (DHT) architecture is known to be a very efficient way to implement peer-to-peer (P2P) computer networks. DHTs are scalable structures as they only require *O(log N)* steps to retrieve an information, where *N* is the number of peers in the P2P network. DHTs became popular with the introduction of Kademlia in 2002 [19]. Many large scale P2P networks counting millions of concurrent users such as the file sharing systems KAD or Bittorrent's Mainline DHT adopted Kademlia's design.

The most recent iteration of P2P networks, the InterPlanetary File System (IPFS) [1], is also based on the Kademlia architecture. However, Kademlia and other DHTs are sadly known to be very vulnerable to the Sybil Attack, as described by John R. Douceur in 2002 [12]. It consists for an attacker in creating many fake peers (called sybils), that are inserted in the P2P network. By coordinating sybils, the attacker can perform several Denial of Service (DoS) attacks against legitimate peers or against the content indexed in the P2P network that can completely undermine the service.

During the golden decade (2000-2010) of P2P networks' research, many studies investigated Sybil Attacks. Some of them were successfully experimented on large scale P2P networks. At the same time, defensive mechanisms were also proposed to limit the impact of such attacks and make them harder to perform. In this paper, we propose to evaluate the resiliency of IPFS against a legacy Sybil Attack targeting content indexed in the DHT. We show that, at the time of our experiments, IPFS is defenseless against a basic Sybil Attack. This calls for a quick adoption of protections to avoid further compromising IPFS deployments, so we also discuss the mechanisms IPFS developers could quickly implement to mitigate this issue. This is all the more important given that IPFS's P2P layer, which implements the DHT and the mechanisms we evaluate here, is available as a standalone library "libp2p" [7]. The P2P systems relying on the "libp2p" library will by default inherit this vulnerability.

The contributions of our paper are the following:

- we evaluate the time needed to pre-compute sybils' cryptographic identities to target any content on IPFS;
- we describe and evaluate the success ratio of a localized Sybil Attack denying content access on IPFS;
- we propose a selection of the most efficient and easily deployable countermeasures from the literature to circumvent the attack.

The rest of the paper is organized as follows. Section 2 quickly presents the background on IPFS. Then, Section 3 describes the way we perform a Sybil Attack on IPFS and Section 4 presents the results of our experiments. Section 5 discusses the defenses that should be quickly implemented by IPFS developers. Section 6 surveys the previous Sybil Attacks conducted on P2P networks as well as the defense mechanisms that were proposed. Finally, Section 7 concludes the paper and presents our future work.

ARES 2024, July 30-August 2, 2024, Vienna, Austria

ACM ISBN 979-8-4007-1718-5/24/07
https://doi.org/10.1145/3664476.3664482

2 BACKGROUND ON IPFS

Created in 2014, IPFS is an open source P2P content addressed file system. It allows people to share data, either directly or via programs that use IPFS as a library, such as the video sharing platform DTube. In its will to become "the storage layer of the decentralized web", some of IPFS's most useful low-level features have been separated into a generic library called "libp2p" [7]. Its developers keep a specification of their protocol, and multiple implementations exist in different languages, although most are written in Go.

Users can publish a **Document** on the public IPFS network or any autonomous private instance. To identify a document, a *Content Identifier* (Cid) is generated based on the content hash, and can be shared out of band. Each document in IPFS is typically split into many smaller **Objects** that can contain actual *Data*, and a list of *Links* to other objects with their Cid and alias. The result is a Merkle Directed Acyclic Graph, with Cids as nodes. Since each Cid contains the hash of its children nodes, this structure can only be built from the leaves upward, and is immutable. This structure allows a peer to download the entirety of a document knowing only the root Cid, and to easily validate content integrity without a direct authenticated canal to the publisher. Contrary to other anonymous P2P systems, privacy protection is not proposed by IPFS. It is possible to know which peer publishes or requests to access a document, as well as to retrieve it.

Peers are identified by a *PeerID*: each node generates an asymmetric key pair used for authentication, and computes its PeerID as a hash of the public key. A peer tells other nodes that it shares a specific content by publishing a **Provider record** on the DHT that contains a link between a Cid and the PeerID. Other peers can help to distribute the content and avoid link rot due to churn by using a "Pin" function. This makes them additional providers, issuing their own Provider record on the network. Pinning can be done voluntarily by other nodes, but publishing a file on IPFS does not automatically replicate it. The information to actually reach a peer, like its IP address, port number, and protocols used, is stored in a *MultiAddress*. An example of MultiAddress is: /ip4/1.2.3.4/tcp/4001/p2p/<PeerID>. A set of MultiAddresses is associated with a PeerID in a **Peer record** that is stored in the DHT. Nodes might also include the last MultiAddresses they know for a PeerID whenever they answer a request related to this PeerID, based on information freshness.

The metadata is published on the DHT, like Provider records mapping a Cid to a provider's PeerID, or Peer records mapping a PeerID with its recent MultiAddresses. The DHT itself strictly follows the Kademlia design [19]. Distance between identifiers is computed thanks to the XOR metric. The routing table is implemented as a tree of K-Buckets with K=20. The lookup process to find a DHT entry is performed iteratively: a peer queries the closest nodes it knows among its contacts for nodes even closer, and then repeats until the information is found, or no closer peer is available. An information published is replicated on the 20 closest peers found after a DHT lookup. As we will show, this specific mechanism can be exploited by a Sybil Attack by choosing carefully the sybils' PeerID to be the closest to a DHT entry.

IPFS keeps additional connections open to the ones required by the Kademlia routing table. The complete list of open connections for a node is referred to as its Swarm which contains between 600 and 900 connections. This acts as a second level of unstructured overlay. When a node searches for an information, it first queries its Swarm in an opportunistic way before starting a DHT lookup.

3 SYBIL ATTACK DESIGN

3.1 Attack scenario

The attack scenario consists in placing a number of sybils equal or superior to the P2P network replication factor (20 for IPFS), so that they are closer than any other legitimate peer to the Target ID. In this way, an unprotected Kademlia lookup process will naturally converge to find the sybils. In the case of a DHT PUT, the 20 closest nodes, which are all sybils, are chosen to hold the record (a Provider record, a Peer record or an IPNS record). For the rest of the paper we consider that the attacker targets a specific content (either by knowing or computing its hash) from which he wants to deny access in IPFS by preventing the search of the corresponding Provider records.

In the case of a DHT GET lookup, every node encountered during the walk is also asked for the record, not just the closest 20. An attacker can easily perform a DoS attack even though the DHT lookup is in part opportunistic, by intercepting every PUT requests and denying GET requests. This avoids the record being stored in benign nodes. This scenario is illustrated in Figure 1. It is similar to a "topology aware localized sybil attack" [13], with the difference that records are eclipsed instead of nodes. This sybil attack was previously performed on KAD [4] with the difference that PeerID was totally unconstrained in KAD while it is linked to a peer's cryptographic keys in IPFS, which requires an additional pre-computation step.

3.2 Pre-computation of Sybils' PeerID

When compared to other Kademlia based P2P networks, IPFS introduces a difficulty because a PeerID is not simply the result of a random function that is easy to bypass but instead the result of a hash of a key-pair generation. This means that an attacker must generate many cryptographic keys before finding a resulting PeerID that can be used for their attack. Moreover, the more peers in the P2P network, the harder it is to get the closest PeerID because the eligible PeerIDs are more constrained.

We launched a scan of the IPFS network in May 2023 to get some knowledge about its current ID space. To avoid straining the network during our measurement campaign, a passive approach was preferred, which [10] indicates to give similar result to an active one. We placed probe-nodes to gather the PeerIDs and Cids on the IPFS network, using a "hydra" node. Hydra nodes use a special implementation of IPFS, and have many PeerIDs on the network called "heads", with a singular local DHT. We used a hydra node with 200 heads, running over 89 hours (over a 3 day weekend). This led to the collection of 3 500 000 Cids, and 6800 PeerIDs. We used this data to determine a maximum distance of 2^{230} for our sybils' PeerIDs to a Target Cid. This distance is sufficient to take the control over 99.95% of Cids in our dataset. We brute-forced the 20 identifiers we need in about 1h30 on a standard 8 cores desktop computer. This is more than fast enough, especially considering that all the generated but discarded identifiers could be saved in a rainbow table to perform other attacks. This would allow for any

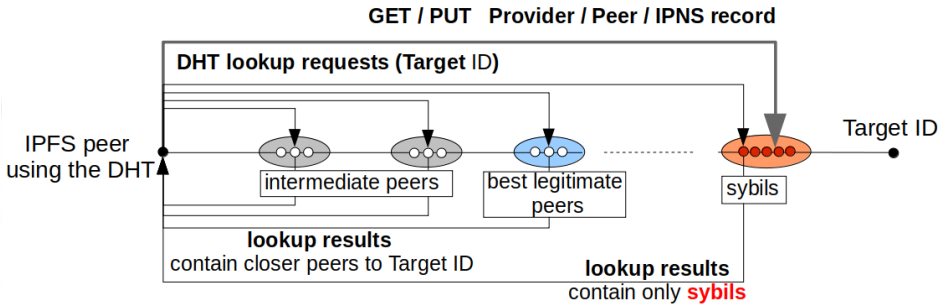

Figure 1: Sybil Attack scenario

Cid to be attacked with only a 90 minutes initial pre-computation time to generate sybils' IDs.

4 EVALUATION

4.1 Sybil client implementation

Our sybils execute the standard IPFS Kubo client, only slightly modified as follows. Overall they behave normally, except when asked for specific Cids that are targets of the attack and given in parameter. For those, they accept to store the Provider record, but never answer it when asked. Of course, we do not target actual IPFS content but only random files we generated on purpose. Sybils also play collectively by only advertising each other when asked for contacts close to the targeted Cid. To help sybils recommend one-another during the lookup process, we removed an IP based filter that prevented sybils to insert one-another in their routing table. Also, they never remove one another even if the connection is cut.

Each modified Kubo client acting as a sybil has its own process run in a shared docker container. Our simple sybil client is sufficient for our objective, but could be improved to eclipse more files with fewer resources. An optimized implementation would likely put multiple sybils per process, share DHT storage like hydra nodes to reduce memory usage, and support fewer protocols to save bandwidth. Using libp2p [7] directly could be a way to reduce overhead.

4.2 Experiment

We made several experiments to validate the effectiveness of our attack. We launched it multiple times against a target file we created and control to ensure that no real file is affected. Each time the sybils are started (at least 20), and left on the network at least 15 minutes to connect. Each sybil uses a public IP address belonging to the same /24 network. Then, the file is provided using a standard Kubo client, only with extra logging. After a couple of minutes, the file is fetched with that same Kubo version. Our nodes do not share memory with prior launches. Provider and Fetcher nodes must be on different machines, to avoid them connecting together via localhost on startup.

The measure of effectiveness is the inability to download the file. If all 20 Provider records are caught by sybils, this is guaranteed. However, some Provider records might be placed on benign peers, due to one of the stop conditions of the lookup algorithm being hit too early in the walk. Another cause of failure might be the peer getting a positive opportunistic answer from its Swarm without needing a successful DHT lookup. As the terminal Kubo interface has an indefinite duration for fetching, it needs to be interrupted manually after some time, even if the file might have been found eventually. We wait for 10 minutes, assuming that most real users would have given up by then, making the attack a success.

When the attack fails, and the file is successfully fetched, we look at the amount of Provider records caught and the time it took to succeed. If most Provider records are caught, the nodes holding the remaining ones might churn out, making the attack succeed afterwards. If the search time is long, the attack can also be successful if users abort their search.

4.3 Results

We tested two versions of the Kubo client for the peer using the DHT. For Kubo 19.2, we first launched a series of tests using 27 sybils because we had 27 public IP addresses at our disposal. We caught every Provider record in 8 out of 11 attempts. The 3 remaining attempts each caught 19 Provider records out of 20, with the fetch succeeding for 2 of them. All of the attack failures happened within an hour of starting the sybils, with subsequent tests succeeding.

For Kubo 20.0, we also launched a series of tests using 27 sybils. We caught every Provider record in 10 out of 12 attempts. The fetch still failed for the 2 remaining attempts, that caught 17 and 19 Provider records out of 20 respectively. Those 2 "near misses" were the first tests in the series.

Convinced that IP filtering was in place according to [21], we still tried to launch the attack using a single IP address and 20 sybils. Surprisingly the attack still worked, and even better than with multiple IP addresses. We ran two series of tests, comprised of 11 and 12 launches. Every one of the 20 Provider records was caught, each time, resulting in the attack success. All the results are summarized in Table 1.

In conclusion, the tested attack works very well. It requires some time for the sybils to be well integrated into the network, but ultimately it only requires a single IP address. Our results show that IPFS lookup process is unprotected and can easily be abused by a simple Sybil Attack, raising important concerns about the reliability of IPFS. We disclosed our findings to Protocol Labs following the

Kubo vers.	Nb sybils	Nb IP@	Nb attack success	Nb attack failure	Nb Records intercepted in case of failure
19.2	27	27	9/11	2/11	19 and 19/20
20	27	27	10/12	2/12	17 and 19/20
20	20	1	11/11	0/11	-
20	20	1	12/12	0/12	-

Table 1: Results of Sybil Attack experiments on IPFS

advertised procedure[1]. The next section discusses how to remediate this weakness.

5 PROPOSAL OF POSSIBLE DEFENSES

We only consider defenses protecting the lookup algorithm that is at the heart of the attack we described and do not consider routing table protections that are already partly discussed in [6] and that are more sensitive because they can alter peers' connectivity. Because the lookup algorithm is driven by the peer querying the DHT, the sole update of a client will make it instantly benefit from an added protection without needing to wait from other clients' update. To allow a quick adoption, we select solutions that are easy to implement and do not break backward compatibility.

While some IP restrictions are enforced on the routing table, none appears to be present in the lookup process. This makes attacks significantly easier than they need to be. So a first layer of defense should enforce constraints about the diversity of IP addresses that are considered during a DHT lookup. This diversity is natural for an open and widely deployed P2P network but would force attackers to distribute their attacks at the IP network level, for instance by leveraging a botnet. Typically, for IPv4, a single candidate per /24 subnetwork should be contacted during a given lookup process and should be allowed in the candidate list of closest contacts. Of course, this threshold must be adapted for IPv6. Since /56 are commonly given to end-users in place of a single IPv4 address, a /32 threshold should be set.

A second layer of defense is to strengthen the lookup algorithm against adversarial routing by actually implementing the recommendations of S/Kademlia. Indeed, our experiments and investigation of the source code showed that none of the S/Kademlia improvements have been implemented, despite being announced by IPFS developers [1]. Currently, all the gathered candidates during a DHT lookup toward the TargetID are shared in a common structure called "searchPeers". An attacker only needs to recommend sybils at one of the steps to poison the entire search. Instead, S/Kademlia [20] recommends that each of the parallel lookups is disjoint (never stepping on the same peer and not reusing data). If fully disjoint walks introduce too much overhead, a hybrid approach would be to split the 10 existing parallel lookups into three disjoint walks with three shared lookups each. The divergent lookup process described in [13] could also be considered and should be even more robust but it is less easy to recommend because of its significant overhead and the protocol changes required. Also, we found that one of the "Lookup termination" conditions is to have already queried the three closest candidates. It is too restrictive and can make the lookup end too early once sybils are found, missing other legitimate candidates. Even outside of an attack, as closer nodes are prioritized for queries, it can lead to good candidates not being queried. This condition should only be active after a minimum delay or number of steps.

Finally, a third layer of defense is to consider solutions taking into account the statistical distribution of PeerIDs. While the attack we described only requires a handful of sybils, they require precise identifiers around the TargetID. This alters the uniform distribution expected from PeerIDs (because resulting from a hash function) and this can be detected by a statistical test applied on the list of the closest candidates retrieved after a lookup [5]. The more sybils and the closer to the TargetID, the easier their detection and filtering. Such statistical test was working in KAD with a replication factor of 10 and should work even better in IPFS on a 20-peers sample. The implementation of this mechanism in the source code of another Kademlia instance [17] can help its integration. The only constraint is that it requires a rough estimation of the number of peers in the network. This can be achieved by issuing DHT lookup toward random identifiers or by inspecting the deepest buckets in a peer's routing table. An alternative to this defense is to tweak "write" walks to eagerly issue PUT requests to very close peers that are empirically close enough to the TargetID, so that at least some records are placed on benign nodes. As read operations are already opportunistic and the information received can be validated, this should be enough to avoid the attack. To avoid obvious sybils, a "too-close" filter could also be implemented for free, but it is only a partial defense as it just reduces the interval where PeerIDs can be brute-forced.

6 RELATED WORK

6.1 Previous Sybil Attack experiments against DHTs

Since the original description of the Sybil Attack by John R. Douceur in 2002 [12], many research papers have refined the attack and tested it on largely deployed P2P networks. We can identify three main strategies used by attackers using sybils against a DHT.

First, they can try to fill a peer's routing table with sybils to remove its connections to legitimate peers and ultimately to the P2P network. This attack was first described by Castro et al. [2] and is sometimes referred as an "*Eclipse attack*" [24] because the targeted peer can be eclipsed from the network. At a large scale, a well prepared attack can lead to the partition of the P2P network. This attack was successfully experimented on the KAD file sharing P2P network [31] which is also based on the Kademlia DHT. Second, they can exploit the DHT lookup process to make it loop indefinitely among sybils that keep announcing each-other until the lookup

[1]Reporting a Vulnerability guidelines: https://github.com/ipfs/kubo/blob/master/SECURITY.md

timeout is triggered. This attack was described in [14] and was able to prevent any request to end in KAD. Third, attackers can choose sybils' IDs to be the closest to a targeted DHT entry in order to capture all the related publication or search requests. Three attacks were launched on the KAD network. With a single machine, [27] directly inserted a massive number of sybils (2^{16}) in peers routing table to take the control over a portion of the DHT ($1/256^{th}$). The two others are more subtle and localized. [16] inserted only a few sybils (2-3) around DHT entries to poison them with fake results when responding, while [4] took the control over the targeted DHT entries by inserting 20 sybils around them. A more recent study [18] considered the security of the DHT implemented in Ethereum. They described two attacks to eclipse a peer from the network by monopolizing the incoming and outgoing connections of the victim, what can then lead to double-spending. They also proposed countermeasures that have been implemented in Geth client (from v1.8.0.). Since Ethereum does not use the DHT to store actual data but only to connect peers, this attack targets the way peers connect, not the lookup mechanism we consider here.

6.2 Defense mechanisms against the Sybil Attack

Given the critical consequences of Sybil Attacks on DHTs and the low cost for an attacker, many defense mechanisms have also been proposed by researchers [15, 26, 30]. Some of them have been successfully implemented in widely deployed P2P networks. A first range of solutions simply relied on a central certification authority granting user access to the network. We will not present them because it is not applicable in the case of an open P2P network.

Another family of solutions leveraged cryptography. S/Kademlia [20] and [2] limited PeerID choice by introducing the resolution of a cryptograhic puzzle similar to the proof-of-work that will be popularized by Bitcoin. In [22] a certification tree is proposed and the ID of a joining peer must be signed by a given number of peers among the same sub-tree. Another way to limit the attack proposed in [11] is to assess a PeerID based on user's IP address and port number. For each IP prefix a group of peers in the DHT is responsible to keep track of the peers and allow new peers to join according to thresholds.

Other solutions try to make DHT inner structures more resilient. [2] proposed to add a second routing table in Pastry as a backup which is less optimized than the main table but more robust to Sybil Attacks. [23] proposed to inspect the connectivity of peers and made the hypothesis that sybils will be more connected. [9] proposed to make public the bootstrap relationship between peers and to send the full list of contacts during a lookup process, in order for a peer to fully master the process and detect suspicious peers. [8] proposed a few rules to prevent sybils to fill a routing table: a periodic reset of the table and of PeerIDs but also a limitation of the number of updates in the table per hour. S/Kademlia [20] proposed to make DHT lookups disjoint, by separating the information gathered by each, and forbidding two walks to step twice on the same node. [13] also designed a more robust "divergent" lookup algorithm but that introduces significant overhead. Another solution described in [5] is to compare the PeerID distribution of the best peers found after a lookup with the uniform distribution that must be seen when peers generate their ID thanks to a random or hash function. Any sybil that specifically chooses its ID alters the distribution around a target and can be filtered as statistical outlier.

In practice, KAD implemented several simple checks that make the life of an attacker harder [3]. First, a peer responsiveness is checked before being added in the routing table thanks to a PING-PONG exchange. Second, peer's information cannot be updated without the proper key. Third, at most one peer per IP address and 10 peers per /24 subnetwork can be at a given time in the routing table, and they cannot be in the same K-bucket (i.e. too close in the DHT ID space). In addition, gtk-gnutella implemented the filtering of sybils in lookup results based on the distribution of PeerIds [17].

6.3 IPFS DHT security studies

In IPFS, the routing table pruning system was abused in [21] to insert sybils and eclipse the node. Counter-measures against the attack described in [21] have been introduced in versions 0.5, 0.6 and 0.7 [6]. Peers are also authenticated in IPFS [29] to prevent spoofing of routing table's contacts.

The closest related work is by far a study by Sridhar et al. [25] that was performed in parallel to our work. They consider the same attack scenario and detect it with the aforementioned solution considering PeerIDs' distribution [5] but changed the counter-measure applied in case of detection. When an attack is detected, instead of trying to avoid sybil nodes according to their contribution to the distribution deviation, they rather estimate the size of the DHT region the client must cover to be sure to find the right amount of uniformly distributed legitimate peers. Then, the client contacts every peer in the zone, including sybils. We reproduced their experiments and their solution works very well against the passive attacker model they consider who simply does not answer to search requests. However, slightly changing the attacker model to answer fake Provider records, instead of not answering at all, defeats their solution. In this case, sybil nodes are still contacted and can abort the lookup prematurely, before any legitimate peer holding an actual Provider Record is contacted. Indeed, in the file "go-libp2p-kad-dht/routing.go" of the IPFS client source code implementing the Kademlia DHT[2] can be found the function illustrated in Listing 1 which aborts the search if the number of Provider Records (psSize), be they fake or not, found by the DHT lookup is 10 or more (default value of count). Moreover, the latest version of IPFS (0.28.0) released at the time of writing does not implement this protection yet and is still vulnerable to the basic Sybil Attack we described.

Listing 1: IPFS stop condition on Provider Records returned

```
lookupRes,err:= dht.runLookupWithFollowup(
 ctx, string(key),
 func(ctx context.Context, p peer.ID)
  ([]*peer.AddrInfo, error) {
        [...]
 },
 func(*qpeerset.QueryPeerset) bool {
        return !findAll && psSize() >= count
 },
)
```

[2]https://github.com/libp2p/go-libp2p-kad-dht/

Yet, we can still point a few differences. For instance, our attack is a bit more efficient because sybils always advertise each other so that when one is found, all are found. So we only need 20 of them when [25] used 45. We additionally highlight the lack of S/Kademlia security features [20] to prevent sybils on the path of a DHT lookup to prematurely abort the lookup with fake answers. Also, we showed that it is not necessary to distribute the attack at the IP network level because the IP address diversity filter is currently only enforced at the routing table level but not in the lookup process. In the absence of preventive rules on the contacted IP addresses during a DHT lookup, only one computer is sufficient and is even more efficient than a distributed attack. As we suggested, the attack is harder to perform by applying restrictions regarding IP addresses on the lookup process.

7 CONCLUSION

The goal of this paper was to evaluate the resiliency of IPFS lookup process against a localized Sybil Attack. We described our attack scenario which includes a pre-computation of sybils' PeerIDs and how we created our sybils by only modifying a few lines of code in a Kubo client. The results of our last series of experiments successfully denying content access in IPFS by simply launching 20 sybils on a single IP address is without appeal: IPFS lookup process is defenseless. We finally discuss simple yet efficient defenses that should be implemented by IPFS developers from what was proposed in the scientific literature and what was successfully implemented in similar Kademlia-based DHTs.

In our future work, we plan to conduct a practical survey of existing defense mechanisms against the Sybil Attack that are applicable to an open P2P network. We will implement them, compare them in a comprehensive manner and combine them to highlight synergies so that we can provide the P2P community with a didactic reference study to make future DHT implementations resilient to the Sybil Attack. We will also consider new active attacker models against IPFS.

ACKNOWLEDGMENTS

This work was partly supported by the France 2030 ANR Project ANR-23-PECL-0009 TRUSTINCloudS and the "Alvearium" Inria and Hive partnership.

We thank the postgraduate student who evaluated the impact of a Sybil Attack on IPFS and obtained the experimental results. They preferred to remain anonymous but can claim their contribution to this work with this sha256 of their public key:

bd171d35522149cb20f6e36ee5b9fa2151922409e5d5e37156fb35cf8f3287c5.

REFERENCES

[1] Juan Benet. 2014. IPFS - Content Addressed, Versioned, P2P File System. *CoRR* abs/1407.3561 (2014), 11. arXiv:1407.3561 http://arxiv.org/abs/1407.3561

[2] Miguel Castro, Peter Druschel, Ayalvadi Ganesh, Antony Rowstron, and Dan S. Wallach. 2003. Secure Routing for Structured Peer-to-Peer Overlay Networks. *SIGOPS Oper. Syst. Rev.* 36, SI (Dec. 2003), 299–314. https://doi.org/10.1145/844128.844156

[3] Thibault Cholez, Isabelle Chrisment, and Olivier Festor. 2009. Evaluation of Sybil Attacks Protection Schemes in KAD. In *3rd International Conference on Autonomous Infrastructure, Management and Security - AIMS 2009 (LNCS, Vol. 5637)*. Springer, Enschede, Netherlands, 70–82. https://doi.org/10.1007/978-3-642-02627-0_6

[4] Thibault Cholez, Isabelle Chrisment, and Olivier Festor. 2010. Monitoring and Controlling Content Access in KAD. In *International Conference on Communications - ICC 2010*. IEEE, Capetown, South Africa, 1–6. https://doi.org/10.1109/ICC.2010.5502179

[5] Thibault Cholez, Isabelle Chrisment, Olivier Festor, and Guillaume Doyen. 2012. Detection and mitigation of localized attacks in a widely deployed P2P network. *Peer-to-Peer Networking and Applications* 6, 2 (May 2012), 155–174. https://doi.org/10.1007/s12083-012-0137-7

[6] IPFS community. 2023. Hardening the IPFS public DHT against eclipse attacks. https://blog.ipfs.tech/2020-10-30-dht-hardening/. Accessed: 20-09-2023.

[7] IPFS community. 2023. Libp2p IPFS Docs. https://docs.ipfs.tech/concepts/libp2p/. Accessed: 20-09-2023.

[8] Tyson Condie, Varun Kacholia, Sriram Sank, Joseph M. Hellerstein, and Petros Maniatis. 2006. Induced Churn as Shelter from Routing-Table Poisoning. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2006, San Diego, California, USA*. The Internet Society, 1–16. https://www.ndss-symposium.org/ndss2006/induced-churn-shelter-routing-table-poisoning/

[9] George Danezis, Chris Lesniewski-Laas, M. Frans Kaashoek, and Ross J. Anderson. 2005. Sybil-Resistant DHT Routing. In *ESORICS (Lecture Notes in Computer Science, Vol. 3679)*. Springer, 305–318. https://doi.org/10.1007/11555827_18

[10] E. Daniel and F. Tschorsch. 2022. Passively Measuring IPFS Churn and Network Size. In *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*. IEEE Computer Society, Los Alamitos, CA, USA, 60–65. https://doi.org/10.1109/ICDCSW56584.2022.00020

[11] Jochen Dinger and Hannes Hartenstein. 2006. Defending the Sybil attack in P2P networks: taxonomy, challenges, and a proposal for self-registration. In *First International Conference on Availability, Reliability and Security (ARES 2006)*. IEEE Computer Society, Los Alamitos, CA, USA, 756–763. https://doi.org/10.1109/ARES.2006.45

[12] John R. Douceur. 2002. The Sybil Attack. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems (IPTPS '01)*. Springer-Verlag, Berlin, Heidelberg, 251–260. https://doi.org/10.1007/3-540-45748-8_24

[13] Daniel Germanus, Stefanie Roos, Thorsten Strufe, and Neeraj Suri. 2014. Mitigating Eclipse attacks in Peer-To-Peer networks. In *IEEE Conference on Communications and Network Security, CNS 2014*. IEEE, 400–408. https://doi.org/10.1109/CNS.2014.6997509

[14] Michael Kohnen, Mike Leske, and Erwin P. Rathgeb. 2009. Conducting and Optimizing Eclipse Attacks in the Kad Peer-to-Peer Network. In *NETWORKING 2009 (LNCS)*. Springer, Berlin, Heidelberg, 104–116. https://doi.org/10.1007/978-3-642-01399-7_9

[15] Brian Neil Levine, Clay Shields, and N. Boris Margolin. 2006. *A Survey of Solutions to the Sybil Attack*. Tech report 2006-052. University of Massachusetts Amherst, Amherst, MA. http://prisms.cs.umass.edu/brian/pubs/levine.sybil.tr.2006.pdf

[16] Thomas Locher, David Mysicka, Stefan Schmid, and Roger Wattenhofer. 2010. Poisoning the Kad Network. In *Distributed Computing and Networking*. Springer, Berlin, Heidelberg, 195–206. https://doi.org/10.1007/978-3-642-11322-2_22

[17] Raphael Manfredi. 2020. gtk-gnutella source code. https://github.com/gtk-gnutella/gtk-gnutella/blob/d5eef26211bbbd664a1be928155c36538060e0c0/src/dht/lookup.c. Accessed: 20-09-2023.

[18] Yuval Marcus, Ethan Heilman, and Sharon Goldberg. 2018. Low-Resource Eclipse Attacks on Ethereum's Peer-to-Peer Network. Cryptology ePrint Archive, Paper 2018/236. https://eprint.iacr.org/2018/236

[19] Petar Maymounkov and David Mazières. 2002. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems (IPTPS '01)*. Springer-Verlag, Berlin, Heidelberg, 53–65. https://doi.org/10.1007/3-540-45748-8_5

[20] Sebastian Mies and Ingmar Baumgart. 2007. S/Kademlia: A practicable approach towards secure key-based routing. In *Parallel and Distributed Systems, International Conference on*, Vol. 2. IEEE Computer Society, Los Alamitos, CA, USA, 1–8. https://doi.org/10.1109/ICPADS.2007.4447808

[21] Bernd Prünster, Alexander Marsalek, and Thomas Zefferer. 2022. Total Eclipse of the Heart – Disrupting the InterPlanetary File System. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 3735–3752. https://www.usenix.org/conference/usenixsecurity22/presentation/prunster

[22] Hosam Rowaihy, William Enck, Patrick McDaniel, and Tom La Porta. 2007. Limiting Sybil Attacks in Structured P2P Networks. In *26th IEEE International Conference on Computer Communications (INFOCOM)* (Anchorage, Alaska). IEEE, 2596–2600. https://doi.org/10.1109/INFCOM.2007.328

[23] Atul Singh, Miguel Castro, Peter Druschel, and Antony Rowstron. 2004. Defending against eclipse attacks on overlay networks. In *EW 11: Proceedings of the 11th workshop on ACM SIGOPS European workshop* (Leuven, Belgium). ACM, New York, NY, USA, 21. https://doi.org/10.1145/1133572.1133613

[24] Atul Singh, Tsuen-Wan Ngan, Peter Druschel, and Dan S. Wallach. 2006. Eclipse Attacks on Overlay Networks: Threats and Defenses. In *Proceedings of IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*. IEEE, 1–12. https://doi.org/10.1109/INFOCOM.2006.231

[25] Srivatsan Sridhar, Onur Ascigil, Navin Keizer, François Genon, Sébastien Pierre, Yiannis Psaras, Etienne Rivière, and Michał Król. 2024. Content Censorship in the InterPlanetary File System. In *31st Annual Network and Distributed System Security Symposium, NDSS 2024*. The Internet Society, 1–17. https://doi.org/10.14722/ndss.2024.23153

[26] Mudhakar Srivatsa and Ling Liu. 2004. Vulnerabilities and Security Threats in Structured Overlay Networks: A Quantitative Analysis. *Computer Security Applications Conference, Annual* 0 (2004), 252–261. https://doi.org/10.1109/CSAC.2004.50

[27] Moritz Steiner, Taoufik En-Najjary, and Ernst W. Biersack. 2007. Exploiting KAD: Possible Uses and Misuses. *SIGCOMM Comput. Commun. Rev.* 37, 5 (Oct. 2007), 65–70. https://doi.org/10.1145/1290168.1290176

[28] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. 2001. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. *SIGCOMM Comput. Commun. Rev.* 31, 4 (aug 2001), 149–160. https://doi.org/10.1145/964723.383071

[29] Dennis Trautwein, Aravindh Raman, Gareth Tyson, Ignacio Castro, Will Scott, Moritz Schubotz, Bela Gipp, and Yiannis Psaras. 2022. Design and Evaluation of IPFS: A Storage Layer for the Decentralized Web. In *Proceedings of the ACM SIGCOMM 2022 Conference* (Amsterdam, Netherlands). ACM, New York, NY, USA, 739–752. https://doi.org/10.1145/3544216.3544232

[30] Guido Urdaneta, Guillaume Pierre, and Maarten Van Steen. 2011. A Survey of DHT Security Techniques. *ACM Comput. Surv.* 43, 2, Article 8 (feb 2011), 49 pages. https://doi.org/10.1145/1883612.1883615

[31] Peng Wang, James Tyra, Eric Chan-Tin, Tyson Malchow, Denis Foo Kune, Nicholas Hopper, and Yongdae Kim. 2008. Attacking the Kad Network. In *Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks (SecureComm '08)*. ACM, New York, NY, USA, 1–10. https://doi.org/10.1145/1460877.1460907