An Analysis of the InterPlanetary File System and Its Future

David Ward
V00920409
CSC 461 Project Report
davidward@uvic.ca

Abstract

IPFS is a decentralized peer-to-peer file system, intended to be an alternative to client-server Internet architecture. The system is content-based, where files are identified by their content instead of a location on the network. There are hundreds of thousands of active nodes across the world, and it is easy to set one up and join. IPFS is an open standard constructed in a modular fashion from several technologies in order to achieve this goal. Its strengths include strong censorship resistance, lack of single points of failure, and reliable performance in many domains. Some weaknesses include difficulty dealing with high-bandwidth multimedia content like video-on-demand. IPFS shows that a more decentralized Internet is possible, and makes a good case for building one.

1. Introduction

For the past thirty years or so, the Internet has been mostly built on the client-server model. Centralized servers hold content, like web pages, and computers connect to them and request to be served this content. The Internet today is dominated by a small number of very large companies that serve most of the content to users, and this has a variety of risks and negative effects. Server architecture, by its nature, results in a small number of points of failure which would affect a great number of users when something goes wrong. Furthermore, clients are somewhat at the whim of the providers to allow them access to the information, and even to keep the information available in the first place. As the Internet continues to grow, a large-scale alternative solution may be required.

1.1 Motivation

Traditional location-based contend addressing (IP address) has a number of flaws inherent in it's structure [1]. For clarification, IPFS is not meant as a complete replacement for client-server systems, but as an alternate system that can be used for specific reasons. At first glance, IPFS seems like a promising solution to a number of the issues with content-based addressing. However, a more comprehensive examination is required.

1.2 Goals

The goals of this research project is to examine the InterPlanetary File System (IPFS) and give a technical overview, examine some of the research, and make some conclusions about the current state and future of the system. The report will proceed as follows: An overview of IPFS will be given, looking at the conceptual framework and technical components it is made of. Then we will examine the research, looking at performance, distribution, applications to multimedia content transfer, and other factors. We will give an anecdote about how a user runs a node on their system, and give some observations about the process and the information it can reveal about the network. Then we ultimately make some conclusions about IPFS, specifically the main strengths and weaknesses, and finally discuss the future of the system and peer-to-peer networks in general.

1.3 Methods

The main method of research was reading the IPFS documentation, the original white paper, and a selection of research papers about IPFS and related systems. These documents each had different intentions and some differences in audience, so they presented about the topic in different ways and had different focuses for each, giving a comprehensive view of the system. The information was compiled into this report.

2. IPFS System Overview

This section describes IPFS itself. We begin with the reason the system was created and the purpose it is supposed to fulfill, followed by the parts of IPFS and how they work, and then describe some technical strengths and weaknesses, with possible improvements.

2.1 Purpose

In the original white paper, Juan Benet says "The InterPlanetary File System (IPFS) is a peer-to-peer distributed file system that seeks to connect all computing devices with the same system of files" [1]. The goal of IPFS, then, is to provide a way for computers to access and transfer data without the use of any central servers. While the goal is to be widespread, it is not necessarily to replace all client-server systems. However, IPFS provides an alternative structure where data is not centralized, and information can flow freely with almost no possibility of censorship.

2.2 Concepts

The conceptual background for IPFS is somewhat different than for the client-server internet, and these ideas allow for the system to function in the way that it does. The three main concepts that should be understood in order to grasp IPFS are content addressing, peer addressing, and content indexing.

2.2.1 Content Addressing

Content addressing is perhaps the most unique aspect about IPFS, and one of the main reasons for why it works the way it does. In client-server architecture, a file has some name and exists on some server, and let's say the server is associated with a domain name which accessible on the Internet. This file can be can be uniquely identified the combination of the domain name, file path, and file name. In this way, a file is identified by its location. In IPFS however, a file is identified by it's content, the actual data in the file. In [2], we see that when a file is uploaded to IPFS, it is broken into smaller parts called blocks or chunks, usually of some small maximum size like 256kB. The content of the block is then fed through a hash function, and the output is what becomes the name for the file in IPFS, referred to as the Content Identifier (CID). Note that if the content of the file were changed, the hash value would no longer be valid. This content-based addressing gives the benefit that the file can be verified by seeing if the hash of each block matches their respective CID's.

2.2.2 Peer Addressing

Peer addressing is similar to content addressing, but for computers that join the IPFS network. [2] explains how A public/private key pair is created when the node joins, and the public key is hashed and the result is set as the PeerID. These IDs uniquely identify the computer in the file system, and does not change unless changed manually. Peer addressing also relies on a system called Multiaddressing, where the path to the node is recorded as a sequence of the communication protocols and the corresponding addresses required to traverse that path. For example, a node multiaddress may indicate the IP version and address, transport layer protocol and port, and then the PeerID.

2.2.3 Content Indexing

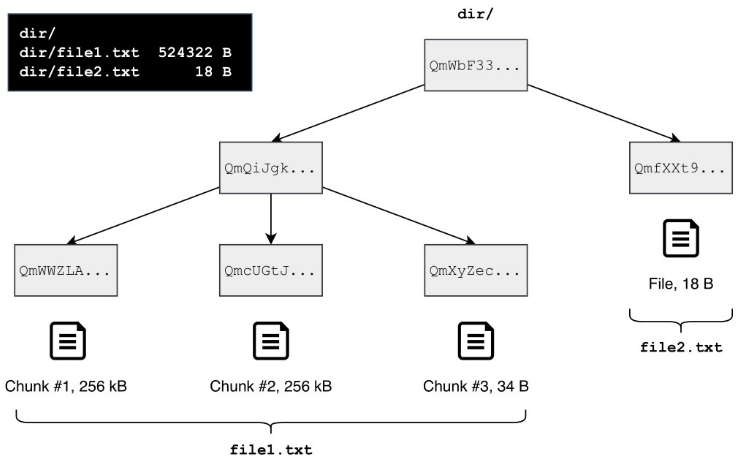
Content indexing refers to the method that IPFS maps content to where it can be found. Since there are no central servers, all blocks, and thus all files, are stored on some node or number of nodes within the network, in their local storage. The mapping of data blocks to storage nodes is done with a Distributed Hash Table (DHT).

2.3 Components

IPFS is constructed from a number of preexisting sub-protocols and components which work together to achieve the goal of a shared and decentralized file system. Understanding them will allow for a more complete grasp of how IPFS works, what it's purpose is, and how it fits in with other network protocols and systems.

2.3.1 Merkle DAG

We have seen that when a file is uploaded to IPFS, it is broken into blocks and the content of each one is hashed to give the block it's CID. But how are the blocks organized into the original file? The solution, as [3] explains, is using a Merkle Directed Acyclic Graph (DAG). It is really more like a tree, where each node is a hash value and the root is the first hash of the file. Data is stored only at the leaves of the graph. See figure 1. Possession of the complete graph is required before the file can be fully obtained.


```
dir/
dir/file1.txt  524322 B
dir/file2.txt      18 B
```

2.3.2 Distributed Hash Table

The Distributed Hash Table is the way IPFS stores peer address and where specific CIDs are stored, and makes this information available to every node so they can find content in the network [1]. The overall design is based on a per-existing DHT called Kademlia [2]. When the CID of a block is obtained, the lookup time to match that value with the peer storing the data is usually constant time, which is the purpose a hash table is used. The table holds the IPv4 and IPv6 address for each peer, along with the connection protocols that are required [4]. When the amount of data in a block is under some very low threshold, the data is stored directly in the DHT, helping to reduce retrieval time [1].

2.3.3 Block Exchange

[1] described the way IPFS nodes exchange data with each other, by way of a novel protocol called BitSwap. Nodes are incentivized to provide data, and punished if they only leech it from others. Nodes have a ‘want’ and ‘have’ list, so other nodes know what blocks they are looking for, and which they can provide. If a node does not have the block that some peer wants, the requested node can search for it among other peers and then provide it. This peer collaboration and barter system is the main way data is disseminated across the network.

2.3.4 Naming System

Due to the fact IPFS does not define objects based on their location, a different naming scheme is required. The system is constructed at a base level to create immutable names, but there are systems in place to allow content identified by a certain hash to be updated without changing the hash value.

2.3.4.1 Immutable Naming

As mentioned, individual blocks of a file are identified by their Content Identifier (CID), which is a hash of the content of that block. The has appears as a long string of characters, and it not very human-readable:

```
QmZ4tDuvesekSs4qM5ZBKpXiZGun7S2CYtEZRB3DYXkjGx
```

A complete hashed object may be a directory with files in it, and the full name would be in the following format (from [1]):

```
/ipfs/<hash-of-object>/<name-path-to-object>
```

Due to the fact the content goes through a hashing algorithm, if the input changes, even slightly, then the output will be different [1]. This makes the objects permanent, and this has several benefits. A peer requesting a file knows if the data they receive is what the CID indicates, because the file can be hashed and the values compared. Versions of a file exist, because alterations to a file create a different identifier, they do not change the original file [1].

2.3.4.2 Mutable Objects and the Self-Certified File System

Immutable names have uses, but it is very often that we want to update a file but still be able to easily access it and call it by the same name. To solve this problem, [1] describes the system built into IPFS called the InterPlanetary Name Space (IPNS). Recall that each node has a unique PeerID which is a hash of it's public key. This value never changes (unless it is altered manually) and is able to be found on the DHT. Every user gets a mutable name space at the path:

```
/ipns/<PeerID>
```

The user can publish objects to this path, and then sign them with their private key, so that the retrieving node can verify the source is legitimate [2]. The permanent PeerID can now used to reference and object without fear that the link will fail if the object is modified.

3. Research

A number of research papers [2, E] have done some in-depth analysis on IPFS, and their findings give a very comprehensive view of the system. The main investigation was into deployment scale, performance, and privacy, which are critical aspects of a decentralized system. Both papers agree that research an analysis of distributed systems is difficult because the network is very large, there is constant node churn, and there may be parts of the network which are not easily or frequently reachable. To mitigate this, the research in both papers was done over a period of at least several months.

3.1 Quantifying IPFS

In [2], the authors looked at the performance of IPFS when it came to uploading and downloading files, the worldwide distribution of nodes, and some observations about node churn.

3.1.1 Performance

The main observable points of IPFS performance are content publication (making files available for other nodes) and retrieval (downloading files from the network), and this is what was tested in [2]. Publication takes 33.8 – 138.1 seconds for the 50th to 95th percentiles respectively. This delay does not depend on file size because it is only the content record that is being published. The actual data of the file remains on the first node until it is requested by another. Content retrieval was measured at 2.9 – 4.7 seconds in the 50th to 95th percentiles respectively. This speed is slightly slower than a typical web request on a client-server system, which is usually around 2.5 seconds on a desktop [5]. Researchers observed a 100% retrieval rate of files

3.1.2 Distribution

The researchers in [2] build a web crawler which ran from a server in Germany, and queried IPFS nodes about the nodes which they had connected with. This crawler ran for over three months. This resulted in determining a rough number of how many nodes are actually in the network. 198,964 PeerIDs were found, which mapped to 464,303 IP addresses. Note that with node churn, the number changes over time, but serves as a useful data point. The crawler was able to connect to a little over half of the nodes at least once, while just under half were never reachable. The distribution of the peers was over 152 countries, with more concentrated in the United States, China, France, Taiwan, and South Korea.

3.1.3 Node Churn

Node churn refers to action of nodes in a peer-to-peer network who stay connected for a period of time and then leave. This is inevitable in a peer-to-peer system because no node is required to remain part of the system, unlike servers. The problem with this as it relates to IPFS is that file data is stored directly in the memory of the nodes in the network, so if a node leaves, the information they have is no longer accessible. It is true that they can rejoin the network and may still have the data, but that is not guaranteed. During research, [2] found that 87% of peers were connected for less than 8 hours, and only 2.5% remained in the network for more than 24 hours.

3.2 Multimedia

IPFS can distribute any kind of file, because all files are broken down into chunks the same way. The system runs above the transport layer, so the movement of files could happen of over UDP, TCP, or others, depending on what is appropriate for the situation [?]. IPFS does not have it’s own transport protocol. However, we have seen that the core of IPFS is built to handle immutable files, and while there is support for mutable file paths, we may note it would not be sufficient for certain applications, because all blocks of a file (i.e. all the CIDs) are still required before the entire object can be reconstructed. The question remains, is there a way to deliver multimedia content over IPFS with reasonable performance and reliability?

3.2.1 Information-Centric Networking

This was the question asked in [6], where they wanted to know the feasibility of delivering video on demand (VoD) over a peer-to-peer network instead of the usual content distribution network (CDN). We can immediately note some similarities between IPFS and CDN, in that they both have a set of computers which are not a main server, but are instead machines which cache content for users to request. However, CDNs are not are set up by some provider and are still a type of server, and thus do not meet the requirements for decentralization that we are aiming for. They found that IPFS alone was not able to deliver VoD sufficiently, because it is only present at the application layer, and only able to do unicast transmission.

Based on this, they decided to test the compatibility of IPFS with Named-Data Networking (NDN) architecture. NDN is a complete replacement for IP which runs at the network layer, intended to address the problems of the client-server model and location-based networking. [6] investigates the hypothesis that the combination of these two technologies, IPFS and NDN, will be sufficient to replace CDNs and provide decentralized video streaming.

3.2.2 Results

To answer this query, [6] constructed a network emulation system with used IPFS and NDN to simulate a group of nodes requesting video-on-demand from each other. The paper goes into great detail about what configurations gave the best results, but we are only interested in broad strokes, and knowing the feasibility of multimedia content over IPFS (potentially combined with other systems). The main finding was that combining IPFS with NDN, a network protocol that is not IP, does allow for adequate delivery of VoD without any CDNs.

3.3 User’s View of a Node

The following is an informal description and analysis of a user running an IPFS node locally, using the IPFS Desktop application (https://docs.ipfs.tech/install/ipfs-desktop/). This program is one of a ways a node can be easily created and connected to the network. When the program starts for the first time, this is what was referred to earlier as a node “joining IPFS”. It is at that point a PeerID is created and the node can communicate with others. The application displays information such as number of peers discovered, amount of data that the node is hosting, incoming and outgoing network traffic, and more. The node was run on a Windows 10 laptop with a 2.8GHz Intel processor, 16GB of Ram, and 89GB of free SSD space. The location of the machine was Victoria BC, and no VPN was used. These observations in general are not unique and can be easily replicated by running IPFS desktop, though the details will vary.

3.3.1 Observations

IPFS Desktop shows data about the network when the node is running. The user can see, among other things, how many peers they have discovered, how much data they are hosting, which countries the peers are in, and incoming and outgoing data rates.

3.3.1.1 First Session

The first session began at 00:15 PST on December 10th and ran for 20 minutes before being halted. 4MiB of data were hosted on the machine for most of the duration. Hosted data are the blocks that the node stores on the machine to they can be retrieved by others. The number of discovered peers fluctuated between 400 and 700, with locations from the United States, Canada, Germany, Spain, France, China, unknown locations, and some others. The latency for these peers was between 20 and 1000ms, with most below 500ms, and was not generally dependant on location. All of the peers used IPv4, with a mixture of TCP and UDP, for their connection protocols. The session was halted, but when this happens, the hosted data is not removed from memory unless it is deleted manually.

3.3.1.2 Second Session

The second session was started at 00:44 PST on December 10th and ran for about 40 minutes. After 9 minutes about 700 peers had been discovered and 8MiB of data were hosted. The incoming network traffic was 50-300KiB/s and the outgoing network traffic was 20-100KiB/s, with some extreme peaks, which was most likely the storage or retrieval of blocks. Peer locations were about the same, with some also appearing from Japan and Eastern Europe. Peer latency distributions and connection protocols were the same as the previous session. After roughly thirty minutes, the incoming and outgoing traffic calmed down to about 20KiB/s incoming and 40KiB/s outgoing, with very few peaks.

3.3.1.3 Third Session

The third session began at 13:27 PST on December 10th and ran for two hours. Note that the amount of hosted data does not reset when the session is ended, so it began at hosting 8MiB. By the end of the third session, the node was hosting 17MiB. The number of peers discovered fluctuated greatly over the time, going from under 200 to over 800 in sometimes less than one minute. The cause of this fluctuation is probably at least partly caused by node churn. The locations of peers were similar to the previous sessions, but there was a wider distribution including countries like Finland, South Korea, Australia, Turkey, Russia, Argentina, and so on. It appeared as though the distribution of node locations increased slowly as the session was longer.

3.4 Analysis

Looking at the task manager, by the end of the third session IPFS desktop used 3-10% of the CPU, about 460MiB of memory, and 2-6Mbps of network traffic. However, based on what was seen, these values, specifically the memory usage, will likely continue to increase slowly if the node is left to run for a long period or time. But of course, there is a maximum amount of memory that the application will use, for any given machine. In any case, the values are not particularly high, and so an IPFS node could run comfortably on a machine with less power than what was tested. This good because for IPFS to work well, it needs many nodes, and that would not be possible if expensive hardware was always required to run a node.

4. Discussion

We have given a technical overview of IPFS, presented several aspects of the research, and given an anecdote about joining the network on a local computer. The high usage of the system over many countries, along with the large amounts of nodes involved is evidence that many people have a use for IPFS, or that it is filling gaps in the services that the Internet had previously been able to provide. However, we should look more closely at the strengths and weaknesses of the system, and what it's potential future is.

4.1 Strengths

The primary benefit of IPFS is that files can be stored and transferred without the existence of any central servers. This means there is no single point of failure, and no entity can completely deny access to particular information. Data also cannot be intentionally deleted, since it may exist on any number of nodes which may or may not be online. These factors, which are inherent in the design of IPFS, mean is it more-or-less resistant to censorship, be that from governments, corporations, or other entities. We saw from the example of running a node that it IPFS does not require a specialized computer, and it is easy for anyone to join and leave at any time. This is important because a peer-to-peer network is worth very little if no one wants to go to the trouble of becoming a node. IPFS is an open standard, and many implementations (instances of IPFS written in a specific programming language) have been created, all of which are open source. It serves somewhat as a test to see what the Internet needs in its current state and in the future, so it makes sense that it is open to be altered and improved in any ways that are needed.

4.2 Weaknesses

Though IPFS is widespread and useful, it has many aspects that could be improved, or do not function optimally. The first is that performance is lower than what most users have when dealing with server-based systems, but this is not surprising. The speed of peer-to-peer systems will likely improve purely as computing power increases and hardware gets better, but speed is something that will generally need to be sacrificed to obtain a decentralized system. IPFS also suffers when it comes to distributing video-on-demand, something which requires other technologies in order to make work. There were also some concerning finds by [2] in their analysis of the distribution of nodes. 24% of PeerIDs were located in China, which is a bit worrisome considering the CCP's history with Internet censorship, like the Great Firewall [7]. However, its unlikely that this will greatly affect IPFS, even if all those nodes were suddenly not accessible. Additionally, a peer cannot make another peer delete data, so information within the network cannot be censored in that way.

4.3 Future

IPFS is considered by some [3] to be in the second generation of peer-to-peer systems, and it is likely we will see continued improvement as time goes on. The Internet has come to a point where it has surpassed what the original designers and engineers thought would happen, and thus the underlying architecture is not totally suited to how we use it today, and in fact has some major gaps which we have described. It is not possible to predict exactly what will happen, but it's likely that IPFS will continue to be used for some time, especially considering how popular it is as the moment, and will almost certainly have more improvements made on it. However, even if it fades out, it can serve as a large-scale test of what people need from a peer-to-peer system, and where some of the problems lie in building that.

5. Conclusion

The InterPlanetary File System is a widespread peer-to-peer network currently hosting a large set of distributed files across the world. Its design was intended to improve on and fix the client-server architecture that has been the mainstay of the Internet for decades. Like any system, IPFS has strengths and weaknesses, but presents a bold and robust picture for what a future decentralized Internet could look like.

References

[1] J. Benet, "IPFS - Content Addressed, Versioned, P2P File System," arXiv.org, 2014. https://arxiv.org/abs/1407.3561

[2] D. Trautwein et al., "Design and evaluation of IPFS," Proceedings of the ACM SIGCOMM 2022 Conference, Aug. 2022, doi: https://doi.org/10.1145/3544216.3544232.

[3] E. S. Daniel and Florian Tschorsch, "IPFS and Friends: A Qualitative Comparison of Next Generation Peer-to-Peer Data Networks," arXiv (Cornell University), Feb. 2021, doi: https://doi.org/10.48550/arxiv.2102.12737.

[4] "Toward Decentralized Cloud Storage With IPFS: Opportunities, Challenges, and Future Considerations | IEEE Journals & Magazine | IEEE Xplore," ieeexplore.ieee.org. https://ieeexplore.ieee.org/document/9903549.

[5] J. Beckman, "Key Website Page Load Time Statistics (2023 Updated Data)," The Tech Report, Oct. 01, 2023. https://techreport.com/statistics/website-load-time-statistics-data/

[6] Onur Ascigil et al., "Towards Peer-to-Peer Content Retrieval Markets: Enhancing IPFS with ICN," Sep. 2019, doi: https://doi.org/10.1145/3357150.3357403.

[7] D. Wang and G. Mark, "Internet Censorship in China: Examining User Awareness and Attitudes," ACM TRANSACTIONS ON COMPUTER-HUMAN INTERACTION, vol. 22, no. 6, pp. 1–22, Dec. 2015, doi: https://doi.org/10.1145/2818997.