**InterPlanetary File System**

This paper introduces The InterPlanetary File System (IPFS), which is a peer-to-peer distributed file system that seeks to connect all computing devices with the same system of files. In some ways, IPFS is similar to the Web, but IPFS could be seen as a single BitTorrent swarm, exchanging objects within one Git repository. In other words, IPFS provides a high through-put content-addressed block storage model, with content-addressed hyperlinks. This forms a generalized Merkle DAG, a data structure upon which one can build versioned file systems, blockchains, and even a Permanent Web. IPFS combines a distributed hashtable, an incentive block exchange, and a self-certifying namespace. IPFS has no single point of failure, and nodes do not need to trust each other.

Today, HTTP is the de-facto way to transmit files across the internet. It works pretty well to move small files because the process is cheap. But, HTTP fails to take advantage of new file distribution techniques invented in the past decade. The upgrades are nearly impossible without breaking backward compatibility and because of the huge investment in the current model of HTTP and the web. In the near future, there will be new challenges like:

* Moving datasets in size of petabytes
* High volume real-time media streams
* Versioning and linking of massive datasets
* Preventing accidental disappearance of important files

To understand how IPFS works, it is essential to understand what a Distributed Hash Table (DHT) is

A Hash Table is a data structure that represents an array of key-value pairs. It uses a hash function to compute an index/key. The hash table facilitates very fast lookup of values based on keys.

A Distributed Hash Table (DHT) is a distributed system that provides access to the key-value store. The store is spread over the participating nodes and offers excellent performance and scalability. DHTs are used widely in peer-to-peer system to coordinate and maintain metadata about the system.

A NodeId identifies a node in the IPFS system. A NodeID is nothing but the hash of its public key. Nodes can change their NodeId anytime, but, they are incentivized to remain the same. The nodes store objects (of their interest) in their local storage. These objects represent files and other data structures in IPFS.

All nodes maintain a DHT that is used to find:

* Network address of other peers in the network, and
* Peers who can serve a particular object

This DHT allows IPFS to find peers that can serve an object and be able to reach to the peer over the network.

IPFS uses a different content addressing method to identify content. This approach is different from what exists in the web today, where a server that contains the object addresses the content.

IPFS content address resolves to an IPFS Object which contains a list of IPFS Links and some content data. When adding huge files to IPFS, it is broken into many smaller chunks, and the address resolves to a list of IPFS links that point to the chunks. This content addressing approach confirms that the address will always return the same file.

Another benefit of this approach is that as long as one of the nodes has the content, it can always be accessed from the IPFS network. This solves the dead-link problem that exists in today’s internet. Duplicate files do not take multiple of space because it will always point to the same content hash. Having just one copy in the network is enough for it to be retrievable from the network.

Given the content address, the IPFS network responds with the peers that have the required objects. The object and its links can be received simultaneously from multiple peers.

Our peers can share the object after IPFS returns it from the network. The node can also verify that the content has not been tampered with because hash value maps to content in the IPFS network. Since the content of the file generates the content address (hash), it changes whenever a file is modified.

It is not convenient to share a new content address every time the file changes. To tackle this issue, IPFS has a concept of **IPNS, or the Inter Planetary Name System.**

IPFS provides a generic way to share files and data in a peer to peer fashion optimizing delivery. Some of the use case scenarios where IPFS would be great are:

* Sharing files and huge datasets.
* Serving websites and blogs on IPFS.
* Using as a data source or sink in data processing workflows.
* Using as a Mounted filesystem.

The white-paper discusses a peer-to-peer, version-controlled approach to a filesystem. This filesystem has no single point of failure. Since the data is content-addressed, the nodes in the network don’t need to trust each other. As per this paper, IPFS has the potential to replace HTTP and make the web, distributed.