Blockchain Council™

Fundamentals to IPFS

InterPlanetary File Systems

InterPlanetary File System (IPFS) is a peer-to-peer distributed file system, seeks to connect all computing devices with the same system of files.

IPFS is the combinational technology of the version controlling system and peer to peer network spreadover global namespace. In simple terms, IPFS is a single BitTorrent swarm, exchanging objects within one Git repository.

In contrast to a central storage based file system, IPFS works on a decentralized system in which every user in the network holds a portion of the overall data, thus creating a resilient system for storage and sharing over the globe.

Any user in the network is able to share a file and it will be accessible to everyone in the network by requesting it from a node which possesses it using the **Distributed Hash Table (DHT)**.

IPFS has a high throughput content-addressed block storage model, with content-addressed hyperlinks, forming Merkle-DAG.

Overview of HTTP

HTTP, or HyperText Transfer Protocol, was invented by Tim Berners-Lee in 1991 and is widely known as the backbone of the World Wide Web.

We are aware that HTTP is a request-response based protocol which means whenever a web browser sends a request to an external server, the function of the server in this case is to return the request with a response message.

HTTP faces problems when there is a disruption in the network and when for some reason the client can’t connect with the server. HTTP web is usually full of broken links in the network.

The HTTP with its location-based addressing model encourages centralization. While centralization guarantees trust in selected applications with the data, there’s always looming danger of our data becoming siloed on the web.

How IPFS address the HTTP problems

To address the content, what IPFS does is that it uses a form of representation of the content, and this is accomplished by integrating a hash on a file, which is taken as the address.

Here, the hash is a root object, and it provides its path to other objects as well.

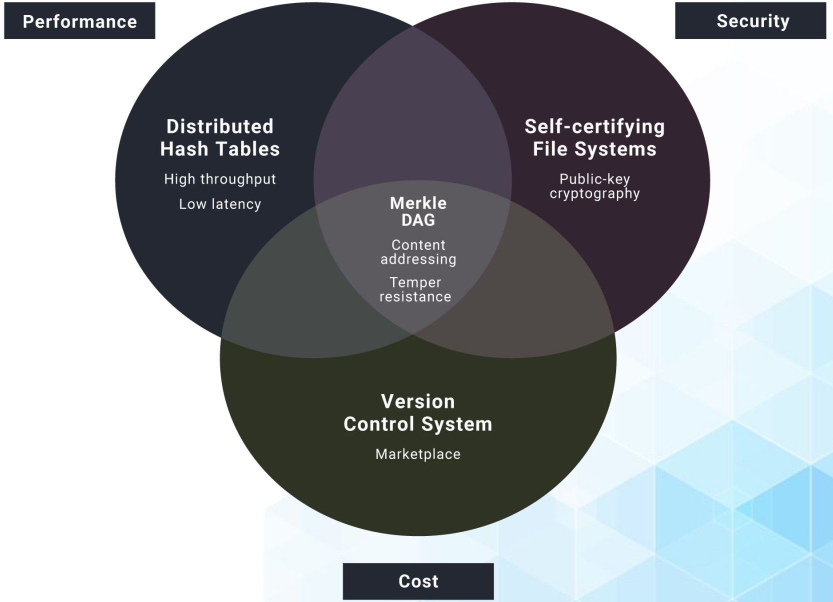
As opposed to requesting a server for data access, you can gain access to this hash, which acts as a starting point of data. This provides physical proximity.

IPFS does this by combining and processing a few similar ideas from other peer-to-peer systems.
A DHT or Distributed Hash Table is used by IPFS for storing data.

After we acquire a hash, we reach the peer who has the content at the hash location, we can then download the content straight from the node which holds the data we want.

IPFS Components

- Distributed Hash Table
- Block Exchanges
- Merkle DAG
- Version Control System
- Self-certifying File System



IPFS Components

Distributed Hash Table:

- The DHT is simply a global hash table.
- The main function of a DHT is to keep track of the data, who has it and where it is.
- In IPFS, the values are the block of data and the keys are the corresponding CIDs. But what DHT means is that the entire table in it is spread at different locations.
- In an IPFS, each computer is the node and shares a part of the larger table.
- For this issue, IPFS employs the use of small tables rather than one big table.
- When a block is to be provided, a node looks for peers with the same peer IDs as that of the CID of the block. The peers do not have the data, but what they do have is a record that tells them to provide the block.
- Also, these nodes store some of the generated metadata with only the part of the public DHT when they are reachable, and not behind a Network Address Translation (NAT).

IPFS Components

Data Distribution:

- Data distribution takes place in IPFS by using a BitTorrent inspired protocol, Bitswap, primarily used for exchanging blocks.
- Bitswap is a message-based protocol having all the messages with want-lists or blocks.
- It works on a Go implementation and a JavaScript implementation.
- The two main tasks of the Bitswap are: one, obtaining blocks requested by the client on the network and two, sending the blocks that it has to other peers who want them. What IPFS does is that it breaks up files into chunks of data called blocks, identified by a content identifier (CID).
- To fetch a file and to find peers that have the file, a node first sends a want-have request to all the peers it is connected to. This request has the CID of the root block of the file (the root block is at the top of the DAG of blocks that make up the file).

IPFS Components

Merkle DAG:

- Merkle DAG is a DAG having nodes with an identifier and this identifier is the result of hashing the node's contents using a cryptographic hash function like SHA256.
- The parents in the Merkle DAG are added after children because the children's identifiers must be computed in advance to be able to link them.
- The nodes in a Merkle DAG are the root of the DAG itself, this subgraph structure is contained in the parent DAG.
- In Merkle DAGs, every node carries a payload, but there are no balance requirements similar to merkle trees. The branches re-converge or a node can have several parents in DAGs.
- Content addressing in DAGs is referred to as identifying a data object by the value of its hash. Therefore, the node identifier is named as a content identifier or CID.

IPFS Components

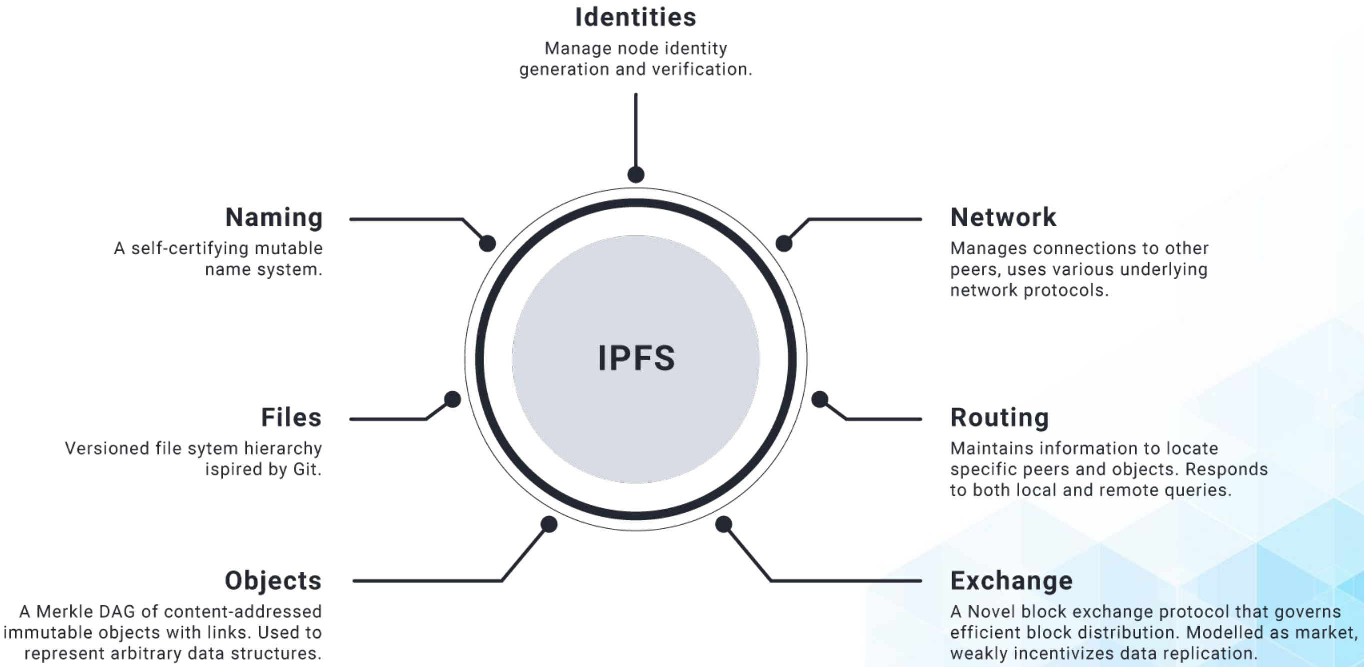
Version Control System:

- The files here are stored and versioned using a Merkle DAG and it allows users to independently duplicate and edit multiple versions of a file, store these versions and later merge edits with the original file.
- As long as objects correspond to the original data, and any new versions are accessible, the entire file history can be retrieved.
- As data blocks are stored locally across the network and can be cached indefinitely, this concludes IPFS objects being stored permanently.

Self-certifying File System:

- A Mutable File System (MFS) is a tool built into IPFS that lets you treat files like you would a normal name-based file system, where you can add, remove, move, and edit MFS files and have all the work of updating links and hashes taken care of.
- Unix FS is the data format used to represent files and all their links and metadata in IPFS and is loosely based on how files work in Unix.

IPFS Protocols



Identities

Manage node identity generation and verification.

Naming

A self-certifying mutable name system.

Network

Manages connections to other peers, uses various underlying network protocols.

Files

Versioned file sytem hierarchy ispired by Git.

Routing

Maintains information to locate specific peers and objects. Responds to both local and remote queries.

Objects

A Merkle DAG of content-addressed immutable objects with links. Used to represent arbitrary data structures.

Exchange

A Novel block exchange protocol that governs efficient block distribution. Modelled as market, weakly incentivizes data replication.

How IPFS does work?

Content is accessible through peers, located anywhere in the world, that might relay information, store it, or do both.

IPFS knows how to find what you ask for via its content address, rather than its location.

Three fundamental principles to understanding IPFS:

- Unique identification via content addressing
- Content linking via directed acyclic graphs (DAGs)
- Content discovery via distributed hash tables (DHTs)

Blockchain Council™