InternetFort

Decentralized P2P powered healthcare management system on IPFS.

White paper v1.00

Dated : 26/07/2018

@hackathon 5.0 [Bikaner]

Contributors

Manish Singh [Independent Researcher @thethird3y3, B.E Pursuing]

Shubham Kuila [CMO @AllReal, B.E Mech Pursuing]

Rushabh Joshi [CFO @AllReal, MCA Pursuing]

Advisor

Aviral Bhatnagar [GuildCapital]

Introduction

IPFS

At its core, IPFS is a versioned file system that can take files and manage them and also store them somewhere and then tracks versions over time. IPFS also accounts for how those files move across the network so it is also a distributed file system.

IPFS has rules as to how data and content move around on the network that are similar in nature to bittorrent. This file system layer offers very interesting properties such as:

- Websites that are completely distributed

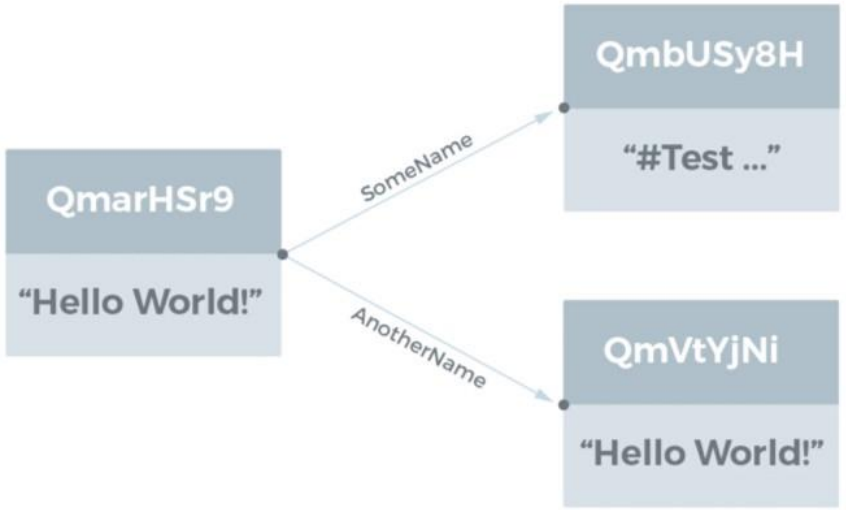
- Websites that have no origin server

- Websites that can run entirely on client side browsers

- Websites that do not have any servers to talk to

How hashes work

Every file in the node are represented by unique hashes which starts "Qm….." this hashes are used to get files from the Nodes. Since every file posses a unique hash. These hash works Like a block structure where every block posses its own hash Identity. IPFS uses port punching method through TCP, similar Technology on what skype works.



Understanding Decentralized Internet

Theory best explain with example

Ramnivas is the patient of Doctor Manish (Where Manish wants to share patient details and reports with Ramnivas and supporting staff)

He puts his PDF file in his working directory

He tells IPFS he wants to add this file, which generates a hash of the file

His file is available on the IPFS network

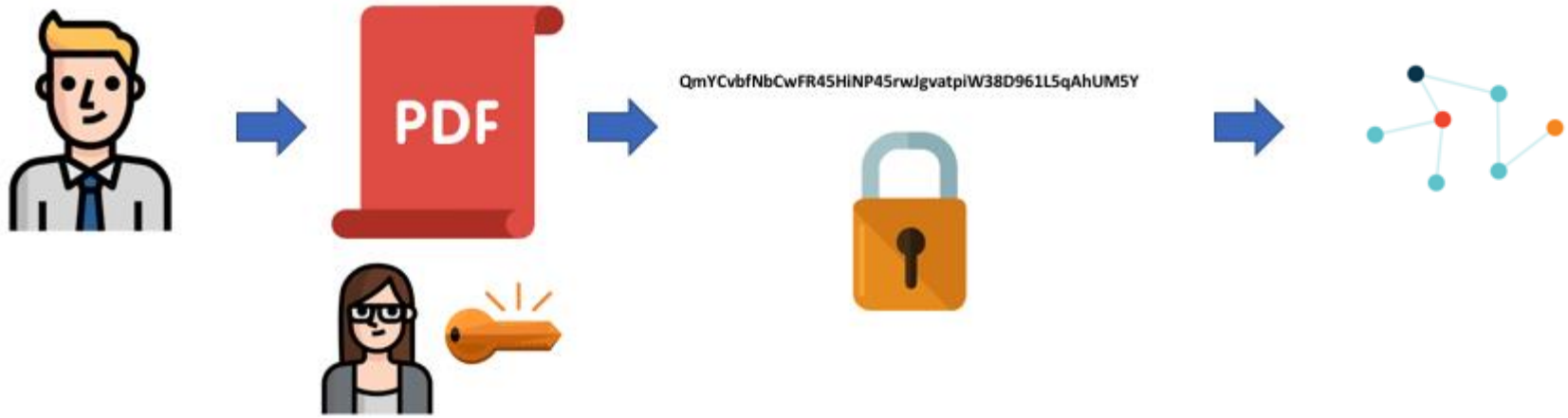
Now suppose Ramnivas wants to share this file with his colleague Lutab through IPFS. He simply tells Lutab the hash from Step 3 above. Then steps 1–4 above just work in reverse for Lutab. All Lutab needs to do is call the hash from IPFS and she gets a copy of the PDF file.

Security

If only hashes are called from IPFS, these hashes can be brute forced to obtain all the files from nodes. Since these are files are unprotected this creates problems for IPFS.

Hence we introduce Asymmetric Encryption, since the files are encrypted from client side only client holds the key to files. This avoids the problem of MITM, even if the file is obtained the file cannot be accessed since the files are encrypted with key. Only client holds the key to file, if patient or staff wants to access the reports only doctor can provide the keys.

Understanding Decentralized Internet



Rijndael MixColumns

In the MixColumns step, the four bytes of each column of the state are combined using an invertible linear transformation. The MixColumns function takes four bytes as input and outputs four bytes, where each input byte affects all four output bytes. Together with ShiftRows, MixColumns provides diffusion in the cipher

$$\begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix} \qquad 0 \leq j \leq 3$$

Introduction to blocks

Unique blocks and structure

Every block in IPFS consist of 256byte of data, when data is larger than 256 byte the data is shreeded into blocks of data.

Every blocks is assigned with unique hash, however slight manipulation of data in the block can change hash value of entire block leading to other parent blocks.

This can be explained by **Merkle Tree**

If children hashes are affected, parent hashes are automatically affected. This kills duplication of data or data manipulation among the peers and nodes.

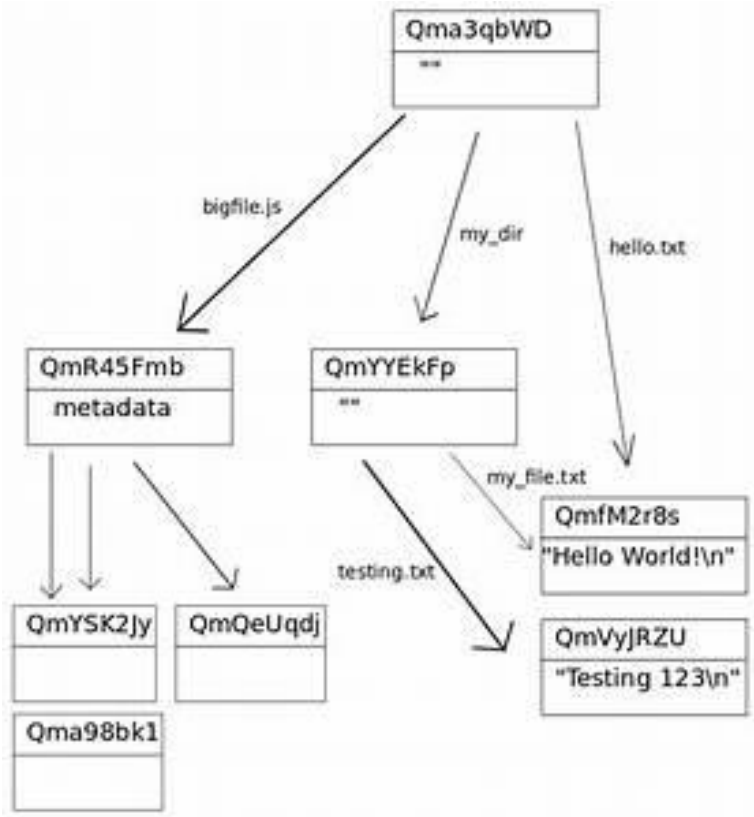
Bitcoin works on concept of merkle Tree.

Merkle Tree

Hash trees can be used to verify any kind of data stored, handled and transferred in and between computers. They can help ensure that data blocks received from other peers in a peer-to-peer network are received undamaged and unaltered, and even to check that the other peers do not lie and send fake blocks.

Hash trees are used in hash-based cryptography. Hash trees are also used in the IPFS, Btrfs and ZFS file systems(to counter data degradation[5]); Dat protocol; Apache Wave protocol Git and Mercurial distributed revision control systems; the Tahoe-LAFS backup system; Zeronet; the Bitcoin and Ethereum peer-to-peer networks the Certificate Transparency framework; and a number of NoSQL systems like Apache Cassandra, Riak and Dynamo.Suggestions have been made to use hash trees in trusted computing systems.

The initial Bitcoin implementation of Merkle trees by Satoshi Nakamoto applies the compression step of the hash function to an excessive degree, which is mitigated by using Fast Merkle Trees.



Door to ever lasting data

TCP Hole Punching & P2P

We assume here that port prediction has already taken place through one of the method outlined above, and that each peer knows the remote peer endpoint. Both peers make a POSIX connect call to the other peer endpoint. TCP simultaneous open will happen as follows:

Peer A sends a SYN to Peer B

Peer B sends a SYN to Peer A

When NAT-a receives the outgoing SYN from Peer A, it creates a mapping in its state machine.

When NAT-b receives the outgoing SYN from Peer B, it creates a mapping in its state machine.

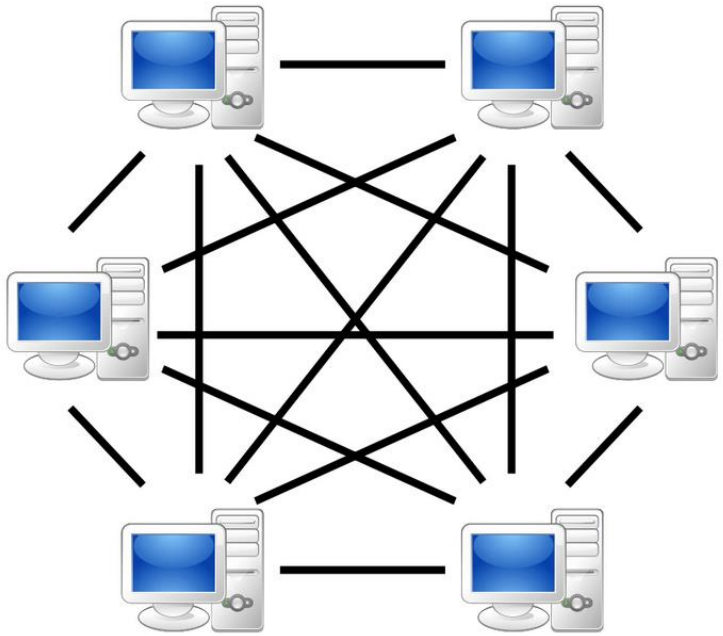
Both SYN cross somewhere along the network path, then:

SYN from Peer A reaches NAT-b, SYN from Peer B reaches NAT-a

Depending on the timing of these events (where in the network the SYN cross),

at least one of the NAT will let the incoming SYN through, and map it to the internal destination peer

Upon receipt of the SYN, the peer sends a SYN+ACK back and the connection is established.



TCP hole punching concept is applied in applications like Skype and other VoIP applications, TCP hole punching avoids the problem of open ports with leaves many physical devices open to attacks over open ports

There is no rote node among the peers in network, the data once on a node can be obtained by any node through unique hash. This increases the rate of transfer among the files but increase the network usage in the directory.

Unless and until data is not deleted from every note files cannot be permanently deleted from IPFS.

IPNS

IPNS

IPFS hashes represent immutable data, which means they cannot be changed without the hash being different. This is a good thing because it encourages data persistence, but we still need a way to find the latest IPFS hash representing your site. IPFS accomplishes this using a special feature called IPNS.

IPNS allows you to use a private key to sign a reference to the IPFS hash representing the latest version of your site using a public key hash (pubkeyhash for short). If you've used Bitcoin before, you're familiar with this - a Bitcoin address is also a pubkeyhash. With our Neocities IPFS node, I signed the image of Penelope (our site mascot) and you can load it using our IPNS pubkeyhash for that node: QmTodvhq9CUS9hH8rirt4YmihxJKZ5tYez8PtDmpWrVMKP.

IPNS isn't done yet, so if that link doesn't work, don't fret. Just know that I will be able to change what that pubkeyhash points to, but the pubkeyhash will always remain the same. When it's done, it will solve the site updating problem.

Now we just need to make the location of these sites human-readable, and we've got all the pieces we need.

Source

Wiki

https://en.wikipedia.org/wiki/Merkle_tree

https://en.wikipedia.org/wiki/InterPlanetary_File_System

https://en.wikipedia.org/wiki/Peer-to-peer

https://en.wikipedia.org/wiki/TCP_hole_punching

IPFS

https://ipfs.io/ipfs/QmNhFJjGcMPqpuYfxL62VVB9528NXqDNMFXiqN5bgFYiZ1/its-time-for-the-permanent-web.html

AES

https://github.com/Urban82/Aes256