swarm

A Guide to Migrating Your Data

From IPFS to Swarm

Who

- Ramesh Pallikara

Who

- Ramesh Pallikara
- DevRel @ Swarm

Who

- Ramesh Pallikara
- DevRel @ Swarm
- Fullstack Web Dev for > 20 years

Who

- Ramesh Pallikara
- DevRel @ Swarm
- Fullstack Web Dev for > 20 years
- Working with IPFS & Swarm for ~3 years

What is IPFS

- peer-to-peer
    - content delivery network
        - with built-in caching and replication

What is IPFS

- peer-to-peer
    - content delivery network
        - with built-in caching and replication
    - file sharing protocol
        - built on this distributed file system called IPFS

What makes IPFS special

- built around the innovation of **content addressing**
  - store, retrieve, and locate data based on the fingerprint of its actual content
  - rather than its name or location

What makes IPFS special

- an IPFS hash a.k.a the CID (Content IDentifier)
  - is immutable
  - will **always** return the exact same content

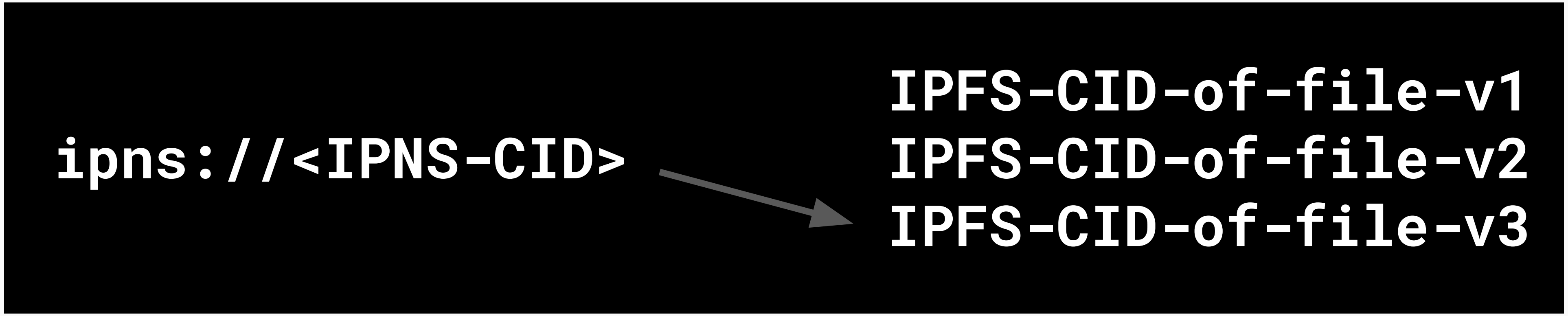
```
ipfs://<CID>
```

IPFS vs IPNS

- For mutable pointers
  - to immutable content
  - or to other mutable pointers
  - **IPNS** records are used

IPFS vs IPNS

- an IPNS record is a static CID
    - that can point to different IPFS CIDs over time
        - for eg: to the latest version of a particular file
- Mutable pointers to immutable content
    - or to other mutable pointers



```
ipns://<IPNS-CID>  →  IPFS-CID-of-file-v1
                      IPFS-CID-of-file-v2
                      IPFS-CID-of-file-v3
```

IPFS vs IPNS

- each IPNS name corresponds to a key pair

IPFS vs IPNS

- each IPNS name corresponds to a key pair
- IPNS name is a CID with a multihash of the public key

IPFS vs IPNS

- each IPNS name corresponds to a key pair
- IPNS name is a CID with a multihash of the public key
- The IPNS record contains the public key and signature
    - allowing anyone to verify that the record was signed by the private key holder

IPFS vs IPNS

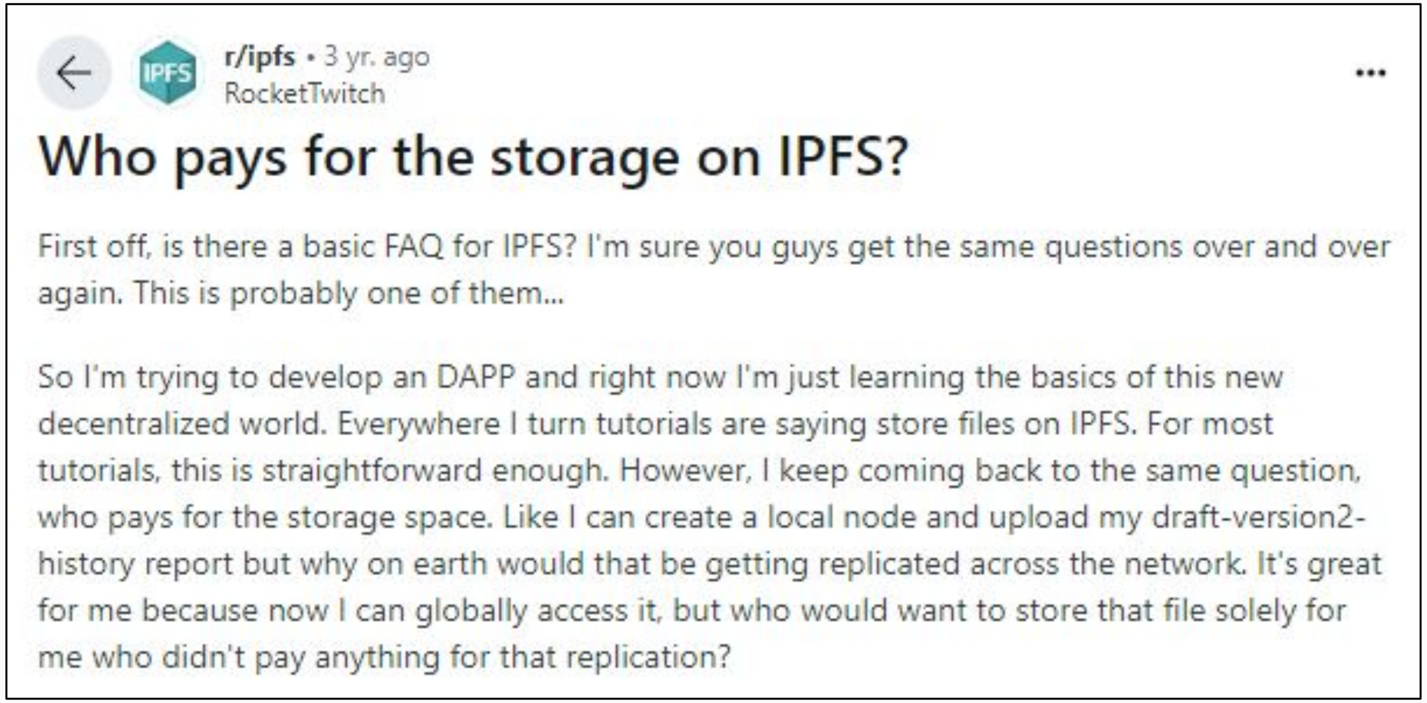
- each IPNS name corresponds to a key pair
- IPNS name is a CID with a multihash of the public key
- The IPNS record contains the public key and signature
    - allowing anyone to verify that the record was signed by the private key holder
- self certifying
    - fast and easy to confirm a record is authentic

swarm

Common Misconceptions about IPFS

Common Misconceptions about IPFS

#1 Storing on IPFS is **free**



r/ipfs • 3 yr. ago
RocketTwitch

Who pays for the storage on IPFS?

First off, is there a basic FAQ for IPFS? I'm sure you guys get the same questions over and over again. This is probably one of them...

So I'm trying to develop an DAPP and right now I'm just learning the basics of this new decentralized world. Everywhere I turn tutorials are saying store files on IPFS. For most tutorials, this is straightforward enough. However, I keep coming back to the same question, who pays for the storage space. Like I can create a local node and upload my draft-version2-history report but why on earth would that be getting replicated across the network. It's great for me because now I can globally access it, but who would want to store that file solely for me who didn't pay anything for that replication?

Common Misconceptions about IPFS

Storing on IPFS is **not really** free.

Common Misconceptions about IPFS

Storing on IPFS is **not really** free.

Unless

- you run an IPFS node yourself and

swarm

Common Misconceptions about IPFS

Storing on IPFS is **not really** free.

Unless

- you run an IPFS node yourself and
- you keep it running and

Common Misconceptions about IPFS

Storing on IPFS is **not really** free.

Unless

- you run an IPFS node yourself and
- you keep it running and
- you pin your files to it to guarantee its availability in the network

Common Misconceptions about IPFS

Storing on IPFS is **not really** free.

Unless

- you run an IPFS node yourself
- you keep it running and
- you pin your files to it to guarantee its availability in the network

Or use an IPFS pinning service within its **free tier**

Common Misconceptions about IPFS

Storing on IPFS is **not really** free.

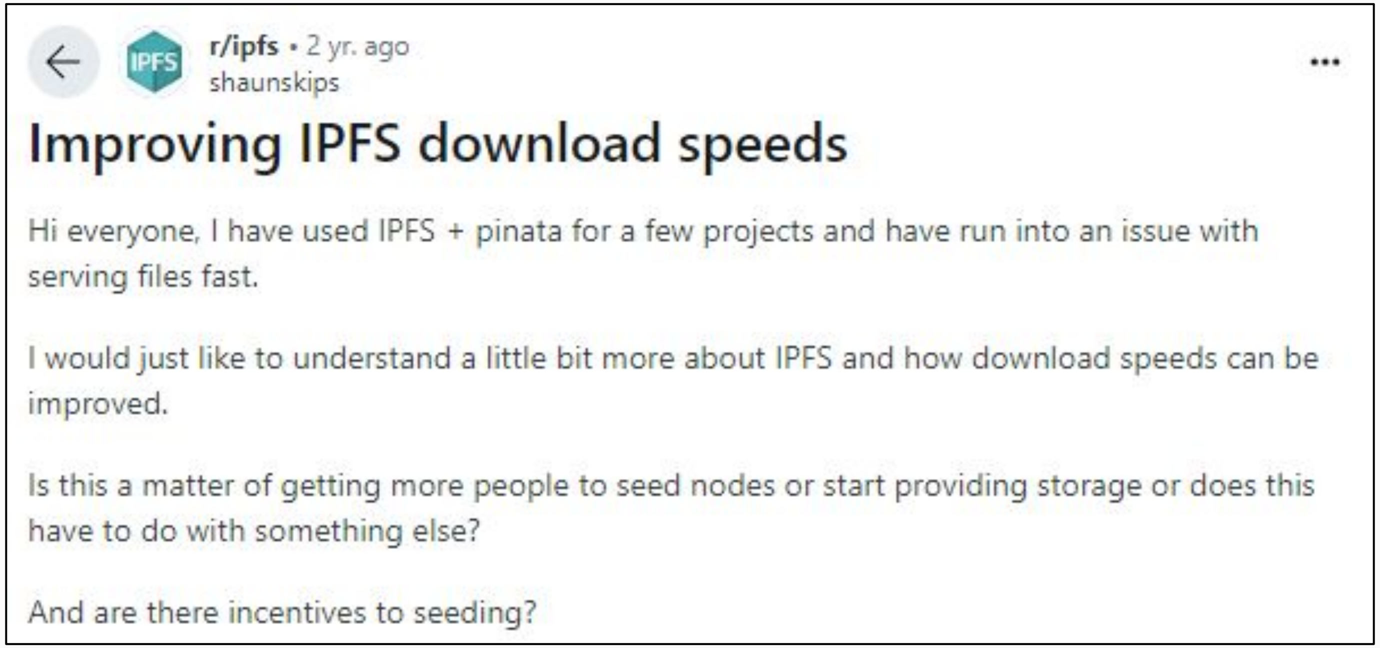
Unless

- you run an IPFS node yourself
- you keep it running and
- you pin your files to it to guarantee its availability in the network

Or use an IPFS pinning service within its **free tier**

Common Misconceptions about IPFS

#2 Download speeds are consistent



Improving IPFS download speeds

Hi everyone, I have used IPFS + pinata for a few projects and have run into an issue with serving files fast.

I would just like to understand a little bit more about IPFS and how download speeds can be improved.

Is this a matter of getting more people to seed nodes or start providing storage or does this have to do with something else?

And are there incentives to seeding?

Common Misconceptions about IPFS

Download speeds are **not really** consistent

Common Misconceptions about IPFS

Download speeds are **not really** consistent

- popular files tend to have faster download speeds

swarm

Common Misconceptions about IPFS

Download speeds are **not really** consistent

- popular files tend to have faster download speeds
- not so popular files tend to have much slower download speeds

Common Misconceptions about IPFS

Download speeds are **not really** consistent

- popular files tend to have faster download speeds
- not so popular files tend to have much slower download speeds
- much like bittorrent

Common Misconceptions about IPFS

Download speeds are **not really** consistent

- popular files tend to have faster download speeds
- not so popular files tend to have much slower download speeds
- much like bittorrent
- third party pinning services can be quite slow

Common Misconceptions about IPFS

My Issues with IPFS

Common Misconceptions about IPFS

My Issues with IPFS

- to guarantee data availability

swarm

Common Misconceptions about IPFS

My Issues with IPFS

- to guarantee data availability
  - i have to keep running an IPFS node

Common Misconceptions about IPFS

My Issues with IPFS

- to guarantee data availability
  - i have to keep running an IPFS node
  - or rely on centralised third party pinning services

swarm

Common Misconceptions about IPFS

My Issues with IPFS

- to guarantee data availability
  - i have to keep running an IPFS node
  - or rely on centralised third party pinning services
- slow base download speeds

Common Misconceptions about IPFS

My Issues with IPFS

- to guarantee data availability
  - i have to keep running an IPFS node
  - or rely on centralised third party pinning services
- slow base download speeds

From IPFS to Swarm

So, how does **Swarm** compare?

	IPFS	Swarm
Atomic Unit	~256 KB blocks/objects/blobs	4KB chunks
Mutable pointers	IPNS	Swarm Feeds Faster resolution & Better guarantees Powerful features & utilities
Download Speed	VARIES Unpopular files download slower Popular files download faster	Fairly higher base speeds Even unpopular files download fast Popular files download much faster
Censorship Resistance	GOOD Difficult to take down content But not impossible.	EXCELLENT Nearly impossible to take down content by self censorship / external agents
DDOS Resistance	GOOD But vulnerable to IP targeted content attacks	EXCELLENT
Privacy / Anonymity	Not much	EXCELLENT
Storage Payment	External/Filecoin	Built-In / BZZ
Incentives	External/Filecoin	Built-In / BZZ

From IPFS to Swarm

IPFS & Swarm - A few Gotchas

IPFS & Swarm - A few Gotchas

- Terms used by IPFS & Swarm
- But mean very different things:
  - **Pinning**
  - **CID**

Pinning

- **Pinning on IPFS**
    - guarantees data availability on the IPFS network

Pinning

- **Pinning on IPFS**
  - guarantees data availability on the IPFS network
- **Pinning on Swarm**
  - does **not** guarantee data availability

Pinning

- **Pinning on IPFS**
  - guarantees data availability on the IPFS network
- **Pinning on Swarm**
  - does **not** guarantee data availability
    - only a valid postage batch can guarantee that

Pinning

- **Pinning on IPFS**
  - guarantees data availability on the IPFS network
- **Pinning on Swarm**
  - does **not** guarantee data availability
    - only a valid postage batch can guarantee that
  - only caches (pins) the chunks locally to the bee node

Pinning

- **Pinning on IPFS**
  - guarantees data availability on the IPFS network
- **Pinning on Swarm**
  - does not guarantee data availability
    - only a valid postage batch can guarantee that
  - only caches (pins) the chunks to the bee node
  - this makes it possible to re-push chunks to the network using the `/stewardship` API endpoint

CID

- **CID on IPFS**
    - refers to **C**ontent addressed **ID**entifier

CID

- **CID on IPFS**
  - refers to **C**ontent addressed **ID**entifier
    - a.k.a the **IPFS hash**

CID

- **CID on IPFS**
  - refers to **C**ontent addressed **ID**entifier
    - a.k.a the **IPFS hash**
  - always returns the exact same content

CID

- **CID on Swarm**
  - does not always return the same content

CID

- **CID on Swarm**
  - does not always return the same content
  - refers to the **61 character length**, base32 encoded string derived from the Swarm Hash using `swarm-cid-js` / `swarm-cid-py` libraries

CID

- **CID on Swarm**
  - does not always return the same content
  - refers to the **61 character length**, base32 encoded string derived from the Swarm Hash using `swarm-cid-js` / `swarm-cid-py` libraries
  - designed to fit max subdomain length restrictions

CID

- **CID on Swarm**
  - does not always return the same content
  - refers to the **61 character length**, base32 encoded string derived from the Swarm Hash using `swarm-cid-js` / `swarm-cid-py` libraries
  - designed to fit max subdomain length restrictions
  - eg: `bah5acgzazjrvpieogf6rl3cwb7xtjzgel6hrt4a4g4vkody5u4v7u7y2im4a`

How - IPFS vs Swarm

- **Upload on IPFS - CLI**
    - Install Kubo

      ```
      https://github.com/ipfs/kubo
      ```

    - Upload

      ```
      ipfs add <path-to-file-or-directory>
      ```

    - Download

      ```
      ipfs get <ipfs-cid>
      ```

How - IPFS vs Swarm

- **Upload on Swarm - CLI**
  - Install **swarm-cli**

    ```
    npm install -g @ethersphere/swarm-cli
    ```

  - Upload

    ```
    swarm-cli upload <path>
    ```

  - Download

    ```
    swarm-cli download <swarm-hash>
    ```

How - IPFS vs Swarm

- **Upload on IPFS - JS**
  - helia

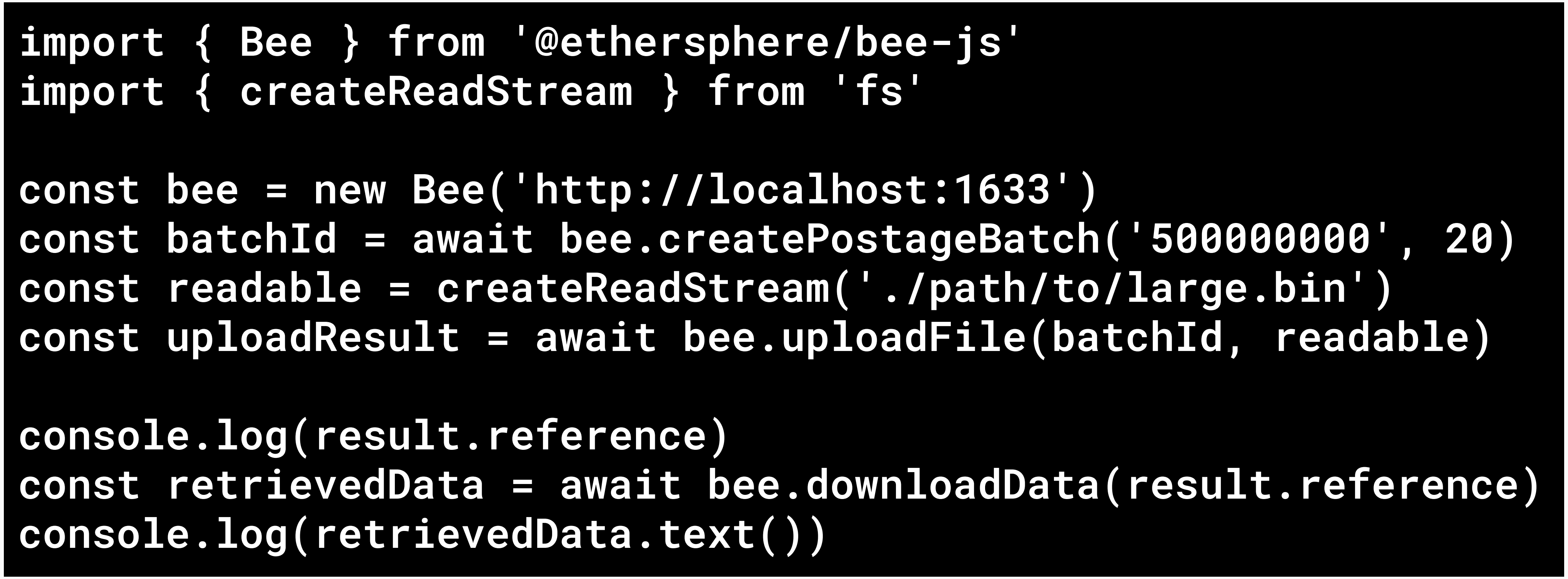
```
import { createHelia } from 'helia'
import { unixfs } from '@helia/unixfs'

const helia = await createHelia()

const fs = unixfs(helia)
const cid = await fs.addBytes(buf)
console.log(cid)
```

How - IPFS vs Swarm

- **Upload/Download on Swarm - JS**
    - bee-js



```
import { Bee } from '@ethersphere/bee-js'
import { createReadStream } from 'fs'

const bee = new Bee('http://localhost:1633')
const batchId = await bee.createPostageBatch('500000000', 20)
const readable = createReadStream('./path/to/large.bin')
const uploadResult = await bee.uploadFile(batchId, readable)

console.log(result.reference)
const retrievedData = await bee.downloadData(result.reference)
console.log(retrievedData.text())
```

How - IPFS vs Swarm

- **Upload/Download on Swarm - Python**
    - bee-py

```
from bee_py.bee import Bee

bee = Bee("http:localhost:1633")
batch_id = bee.create_postage_batch('500000000', 20)
upload_result = bee.upload_data(batch_id, "Bee is Awesome!")
print(str(upload_result.reference))

data = bee.download_data(upload_result.reference.value)
print(data.text())
```

From IPFS to Swarm

Thank You 🙏

Ramesh Pallikara - DevRel @ EthSwarm