Decentralized Systems

IPFS & ENS

InterPlanetary File System

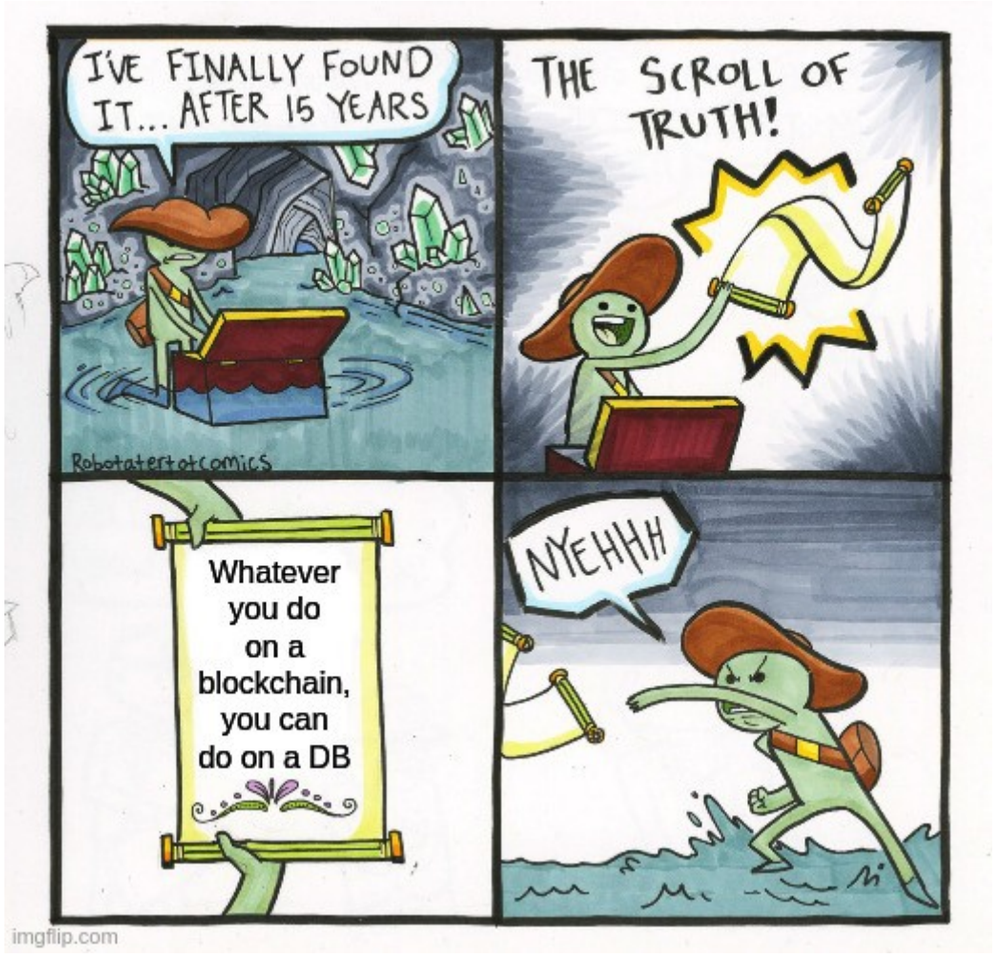
(thanks to Marina Ribaudo)

A Centralized Internet

- We do **not own our data**
- Storing all information in central servers can cause problems, like having **single points of failure** and **censorship**
- Nowadays, the Internet is way less centralized than one may imagine, and that even causes catastrophic failures…

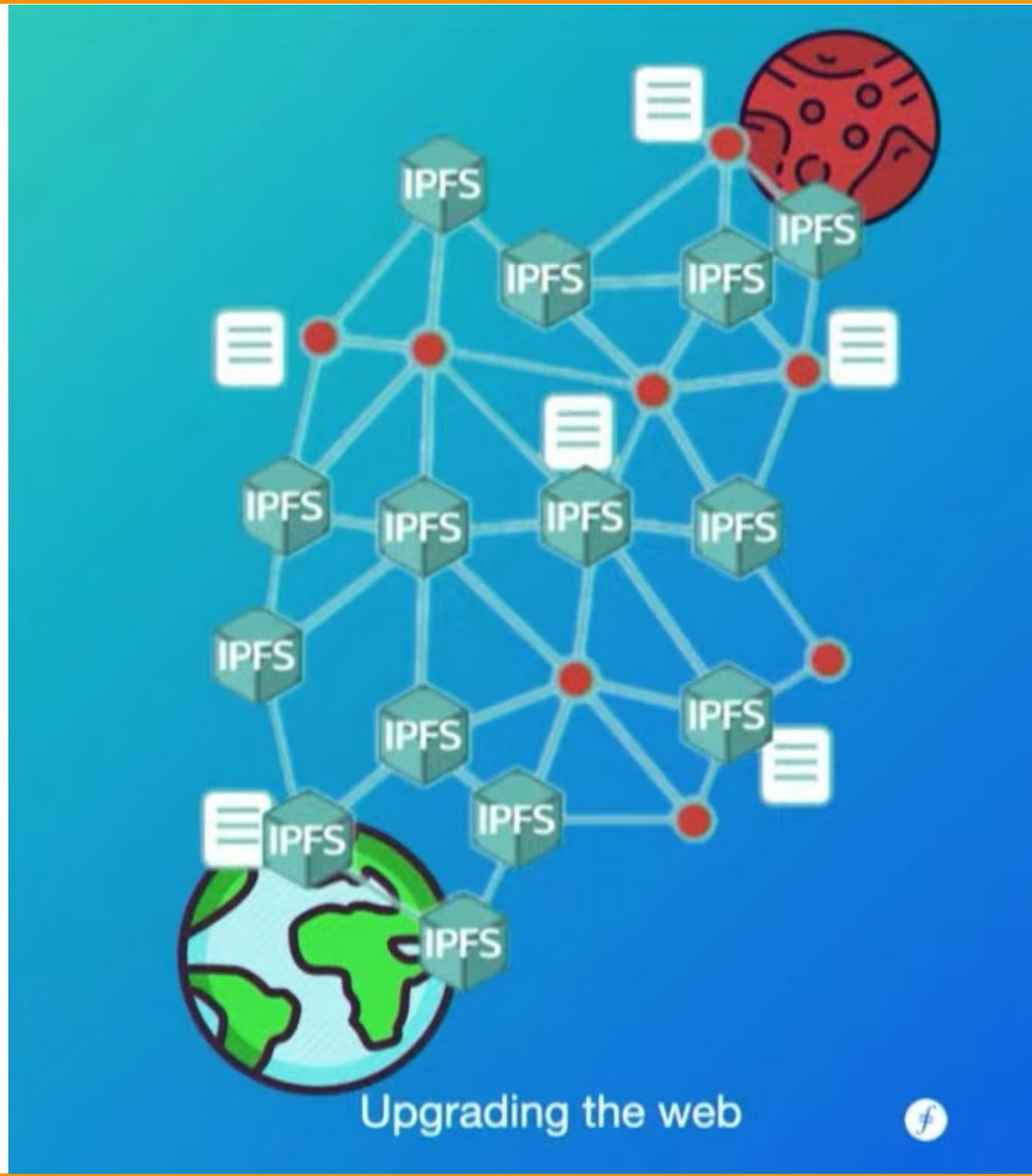
A Centralized Internet

- The current way in which information is provided to users is **location-based**
  - Users refer to resources by URL
  - http://gallery.com/meme1.png



InterPlanetary File System

BitTorrent + git → IPFS



A Centralized Internet

- The current way in which information is provided to users is **location-based**
  - Users refer to resources by URL
  - http://gallery.com/meme1.png
- Problem: images can change behind the same URL

IPFS

- The InterPlanetary File System (**IPFS**) is a protocol and P2P network for **storing** and **sharing data** in a distributed file system
  - **Ethereum** does **computation**
  - **IPFS** does **storage**

- IPFS allows users to not only receive but host content, in a similar manner to BitTorrent

IPFS

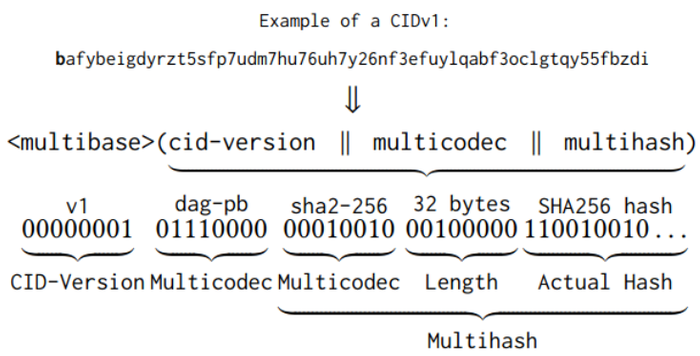
- IPFS lets users **back up files and websites** by hosting them across **numerous nodes**
- Content is **resistant to censorship** and **centralized points of failure**, such as server issues or **coordinated attacks**
- Rather than using file locations, **IPFS points towards contents**, which could be stored on any number of computers around the world

Content-Based Addressing

- Addresses content by **what** it is, rather then **where** it is stored
  - Computer → file://path-to-file/index.html
  - Internet → https://domain.com/path-to-file/index.html
  - IPFS → ipfs://[CID]/path-to-file/index.html
- Example CID: `QmcniBv7UQ4gGPQQW2BwbD4ZZHzN3o3tPuNLZCbBchd1zh`
  - Long binary string encoded to be **printable**

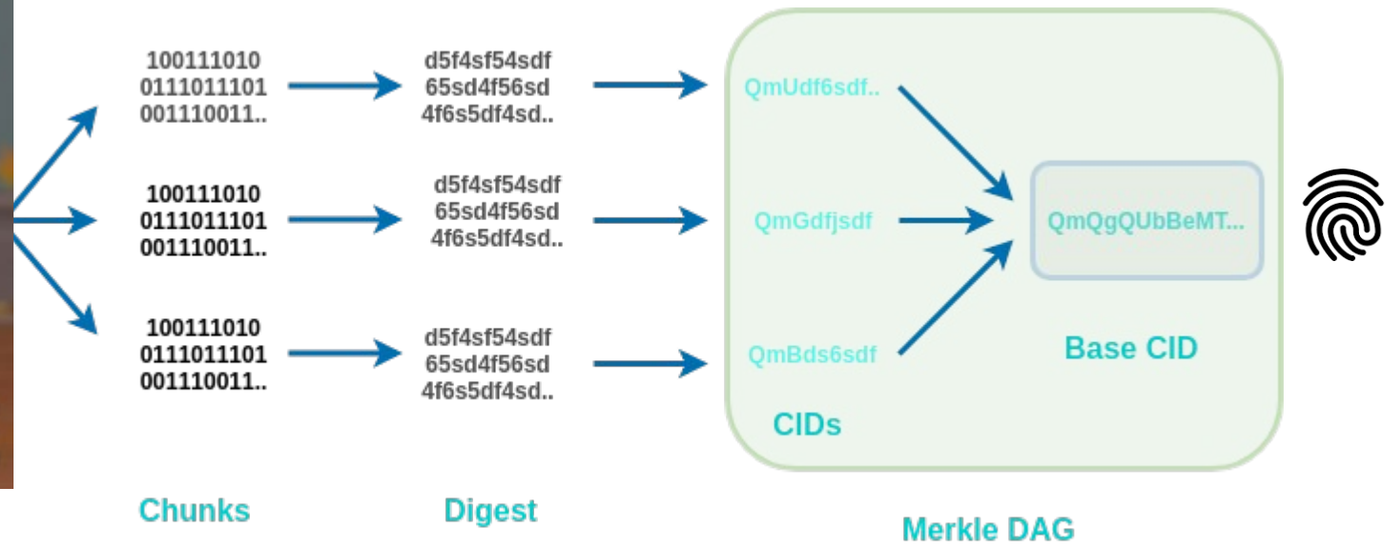
Content-Based Addressing

- For each file, and for each folder, a Content Identifier, or **CID,** is computed using a hash function
- This hash is used to store and retrieve all files in IPFS

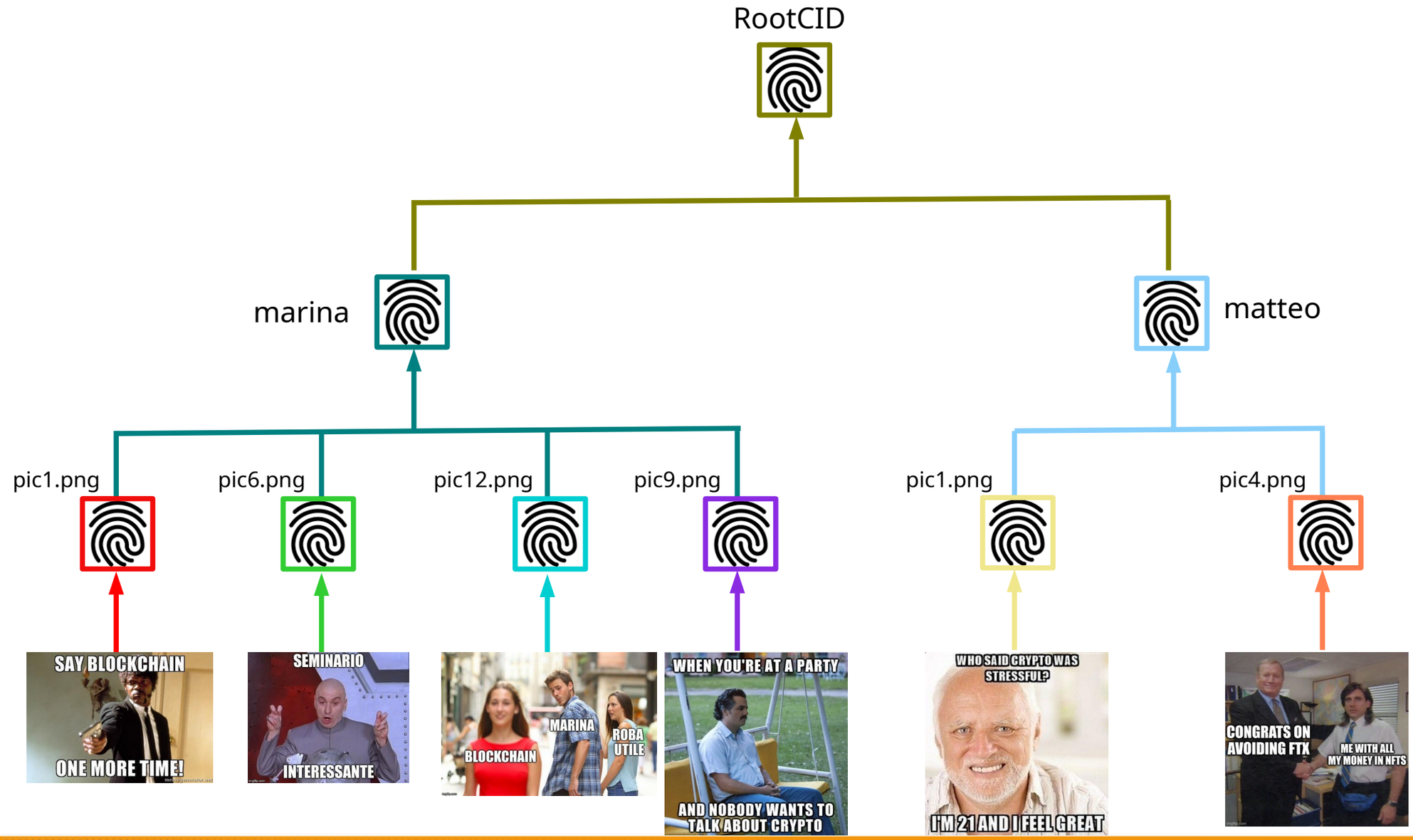


Content-Based Addressing

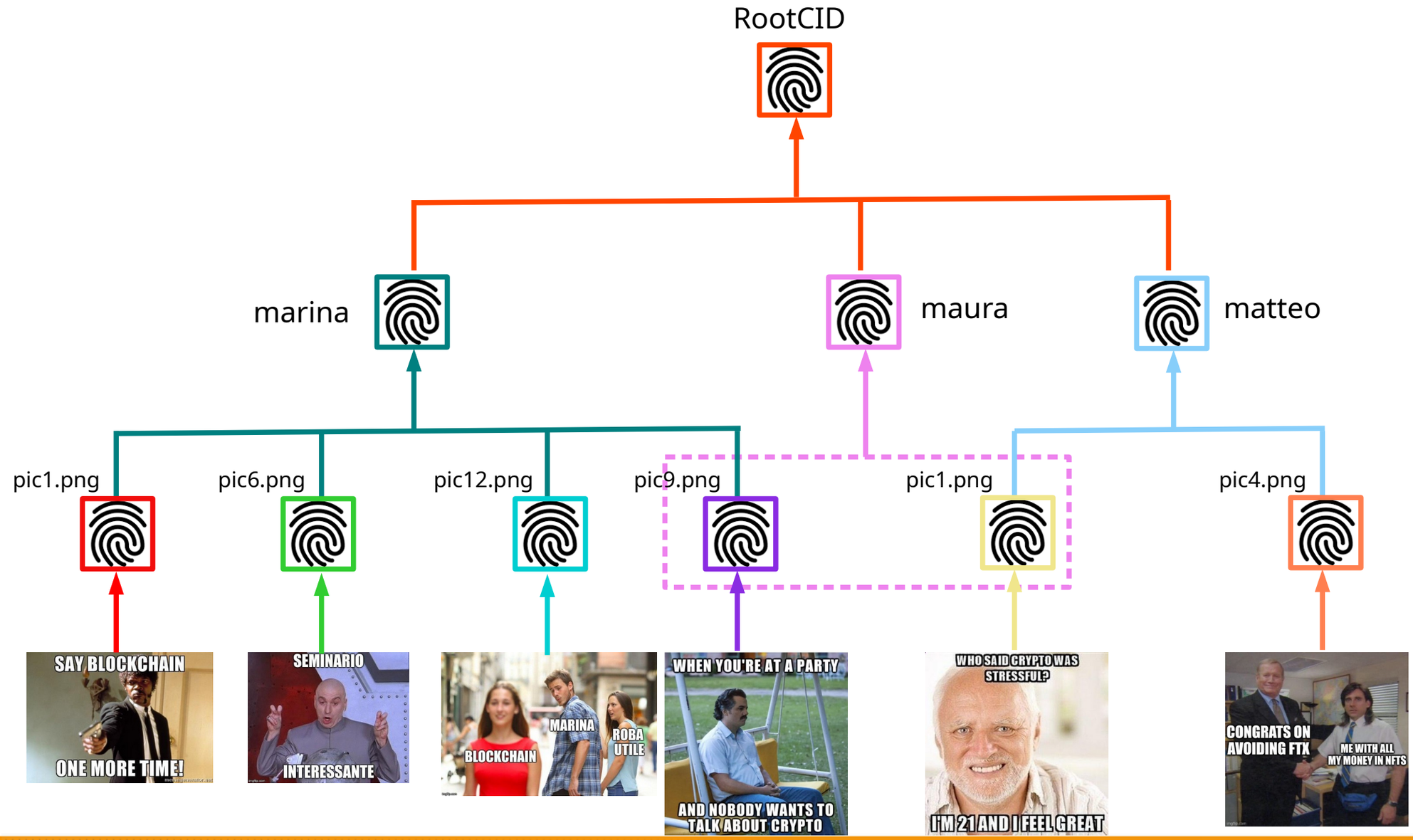
- Large files are **chunked**, **hashed**, and **organized** into **Merkle** DAG objects https://docs.ipfs.tech/concepts/merkle-dag/
- Blocks equal or smaller than 256 kB



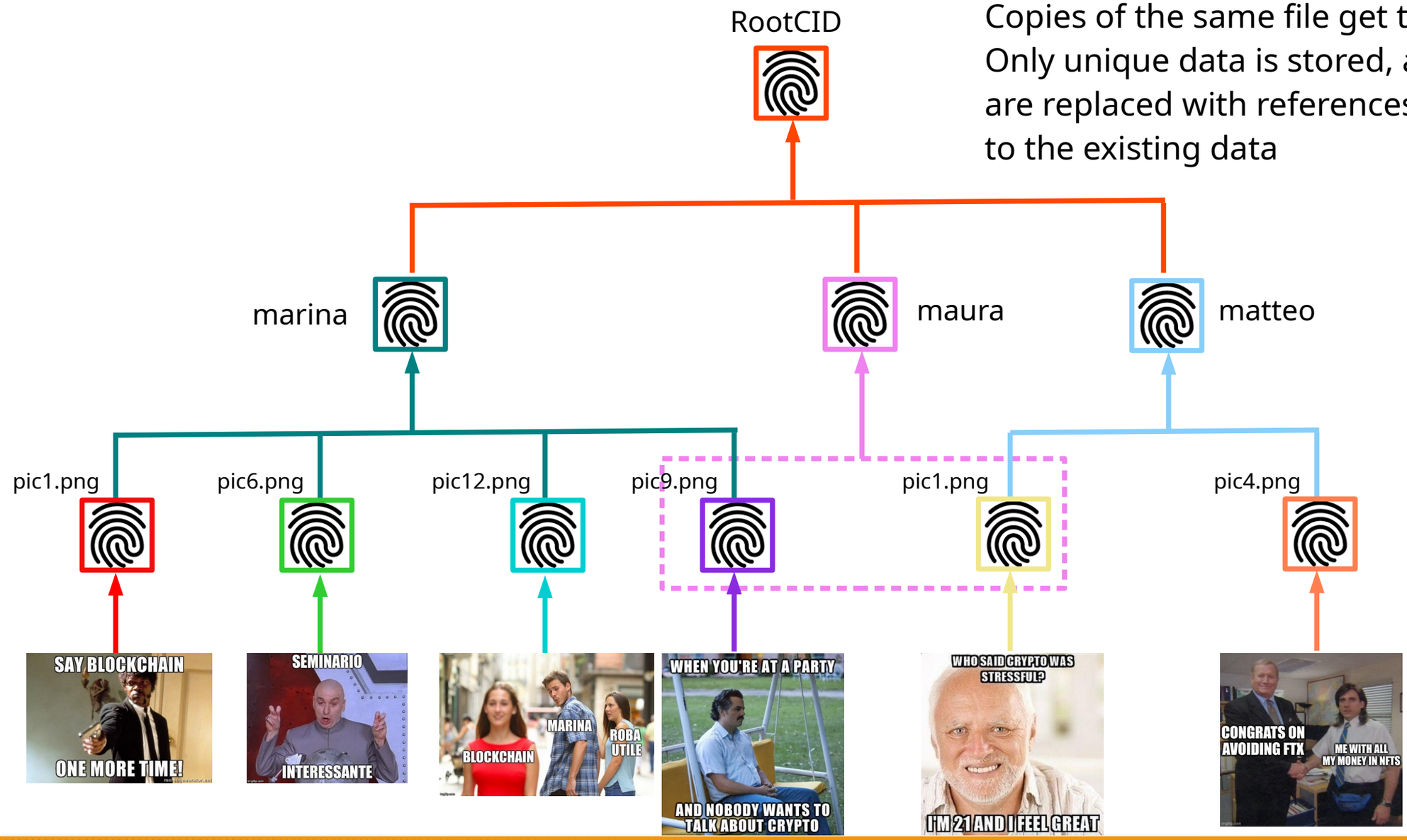
IPFS: files and folders



IPFS: deduplication



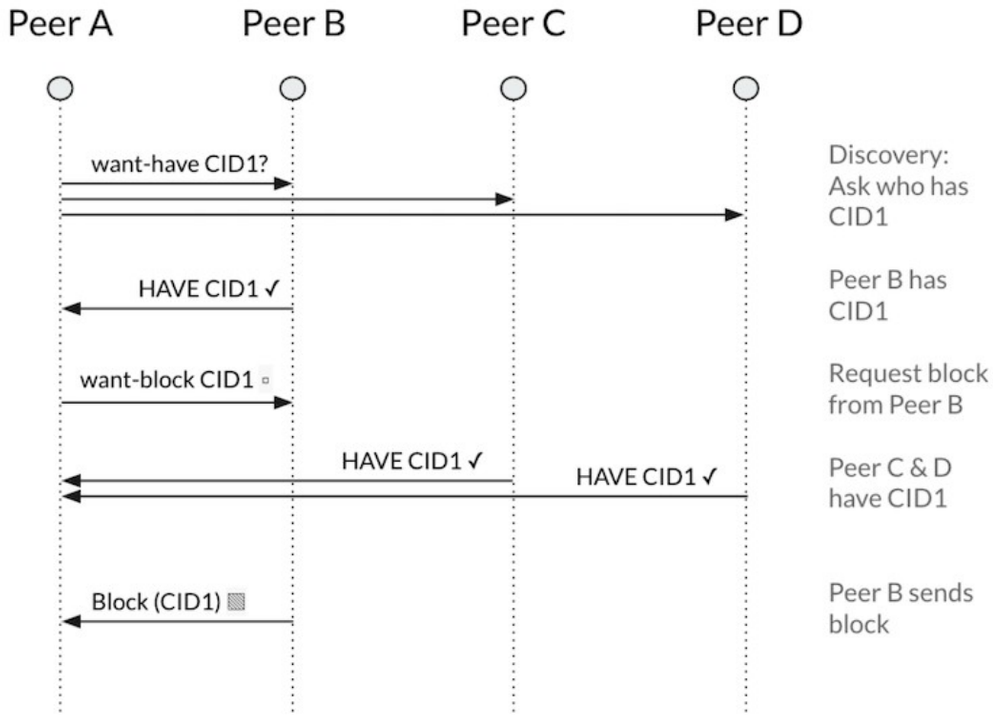
IPFS: deduplication



Copies of the same file get the same CID
Only unique data is stored, and duplicates are replaced with references or pointers to the existing data

Upload/Download

- Uses the BitSwap protocol, inspired by BitTorrent
- Remember? A **swarm** of peers, exchanging a file made of **small blocks**
  - Tit-for-tat incentives
- Here, a single swarm for **all of IPFS**



https://docs.ipfs.tech/concepts/bitswap/

Content Storage

- After you download a file/block, you **cache** it and **serve it to others**
  - Similarly to **seeding** in BitTorrent
- When disk space is over, you will erase **old pieces of data** you didn’t access anymore
- You can decide you’ll keep storing a piece of data by **pinning** it
  - It won’t be evicted from the cache
- A file remains available as long as **somebody is still keeping it**

Versioning: IPNS

- IPFS supports **versioning** through IPNS (InterPlanetary Naming System)
  - Instead of a CID, you use the **hash of a public key**
  - As content, you have a file with
    - The full public key
    - The **version number**
    - The CID of the **current version**
    - Signature
- Storage nodes will store the **last correctly signed version**

DNSLink

- You can have a **human-readable** IPFS address by putting **an IPFS link in your DNS record**
- It’s going to be automatically substituted to your DNS name

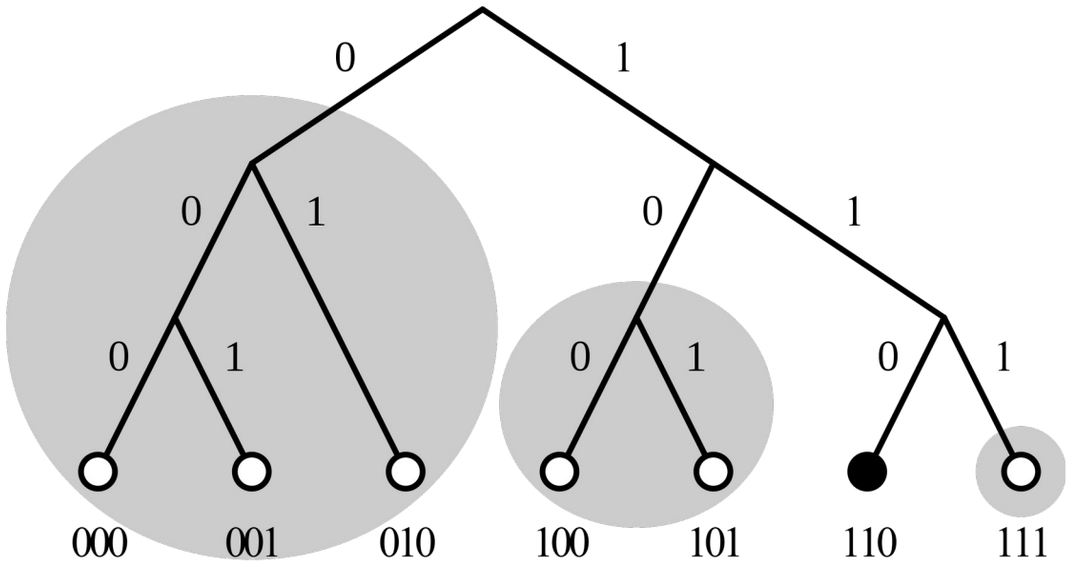
```
$ dig +noall +answer TXT _dnslink.docs.ipfs.tech
_dnslink.docs.ipfs.tech.  34  IN  TXT "dnslink=/ipfs/QmVMxjouRQCA2Qy
```

https://docs.ipfs.tech/concepts/dnslink/

- Here, `/ipns/docs.ipfs.tech/introduction/` will be converted to `/ipns/QmVMxjouR.../introduction/`

Lookup: DHT

- Current connections may be missing some CIDs
- IPFS uses the **Kademlia DHT** (remember?) to support routing and discovery of content and peers on the network
- Peer Ids are directly mapped to CIDs



Content Lookup

- IPFS implements content lookup through a distributed hash table (**DHT**) and **CIDs**
  - Each **node** in the IPFS network is assigned a **unique identifier**, typically a **256-bit value**
  - When a node wants to find a particular piece of content, it queries the DHT with the **content's CID** (**256-bit value**)
- Lookup is performed on Kademlia; at last one will get the **addresses of nodes having that CID** (if any)
  - They’re added to the list of local neighbors, and download proceeds from there

Notable Properties

Nodes can cache content they have recently accessed, making it available to other nodes in the network, this encourages content replication across multiple nodes, enhancing fault tolerance and availability

IPFS uses **deduplication** to help **reducing redundancy at the storage level** by ensuring that identical content is only stored once

IPFS uses **caching** to help **improving access times** by keeping frequently accessed content closer to the requester. The decision to use caching and the specific caching policies can vary among different IPFS implementations

IPFS Nodes (1)

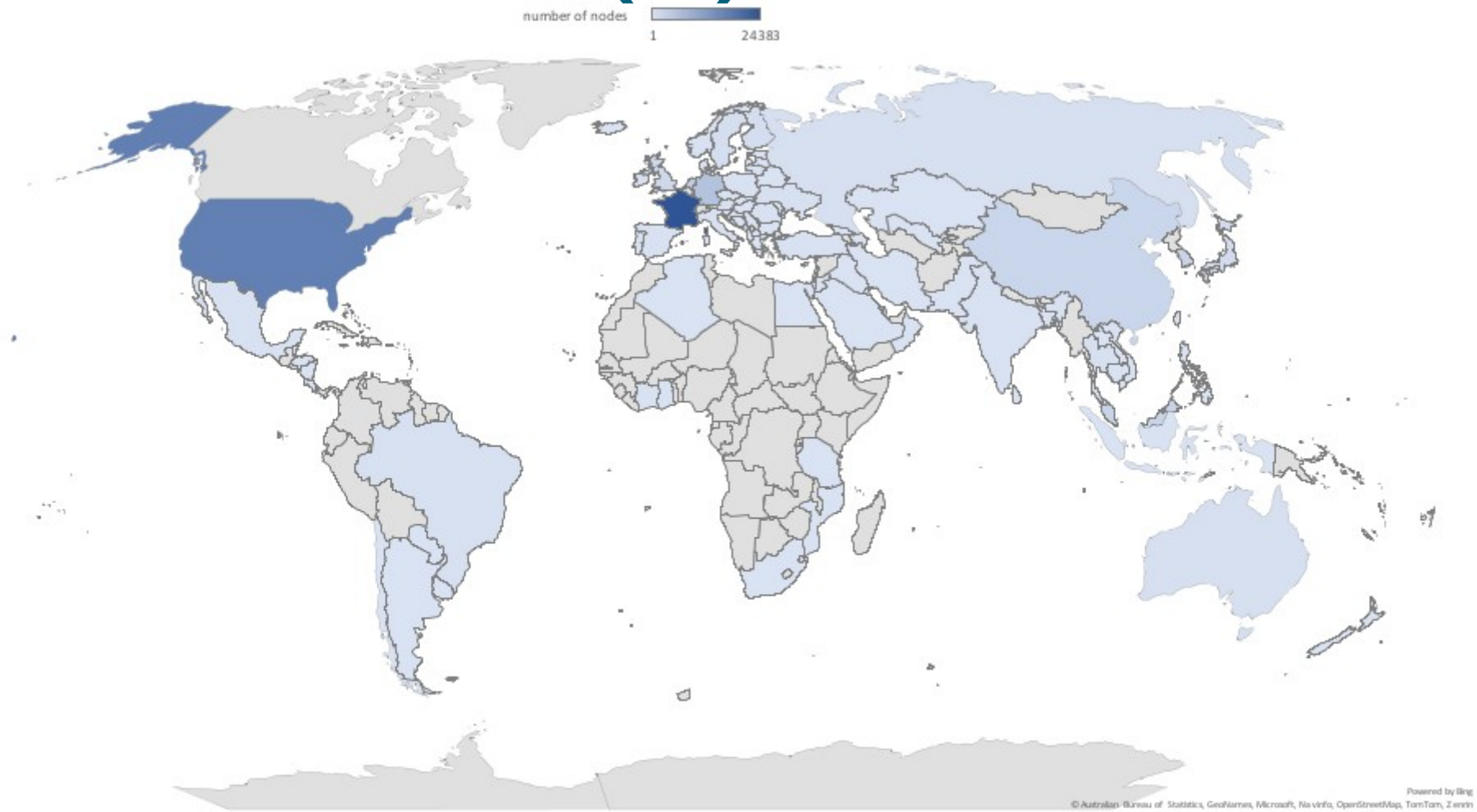
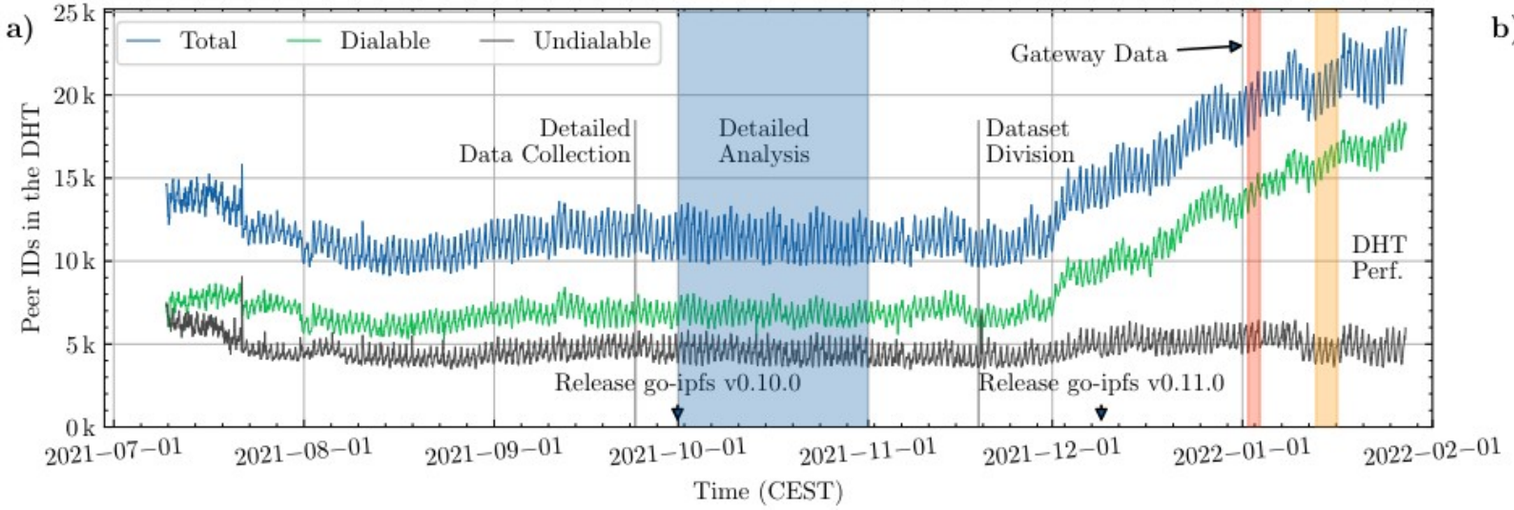
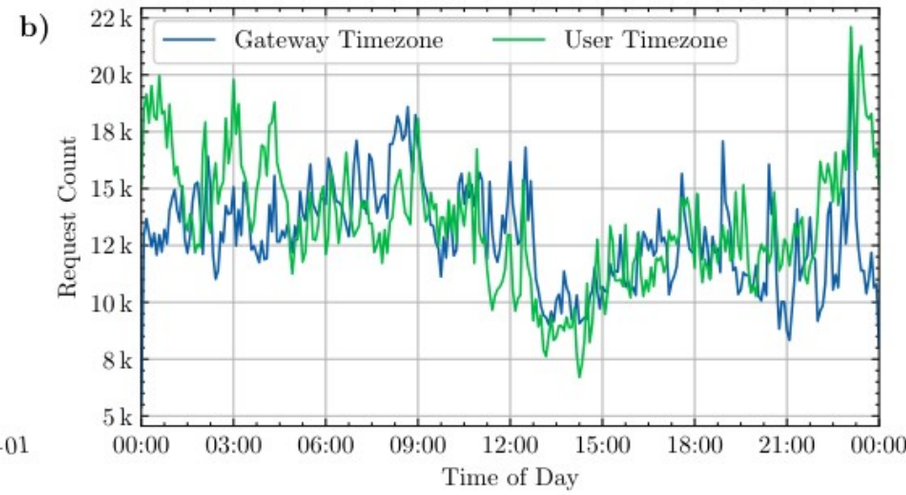


Fig. 5: Number of IPFS nodes per country.

Source: Confais et al., 2023

IPFS Nodes (2)





- Source: Trautwein et al., ACM SIGCOMM ‘22
- Something I don’t understand in the plots, if anybody wants to download the data and try to reproduce them as their seminar…

Filecoin

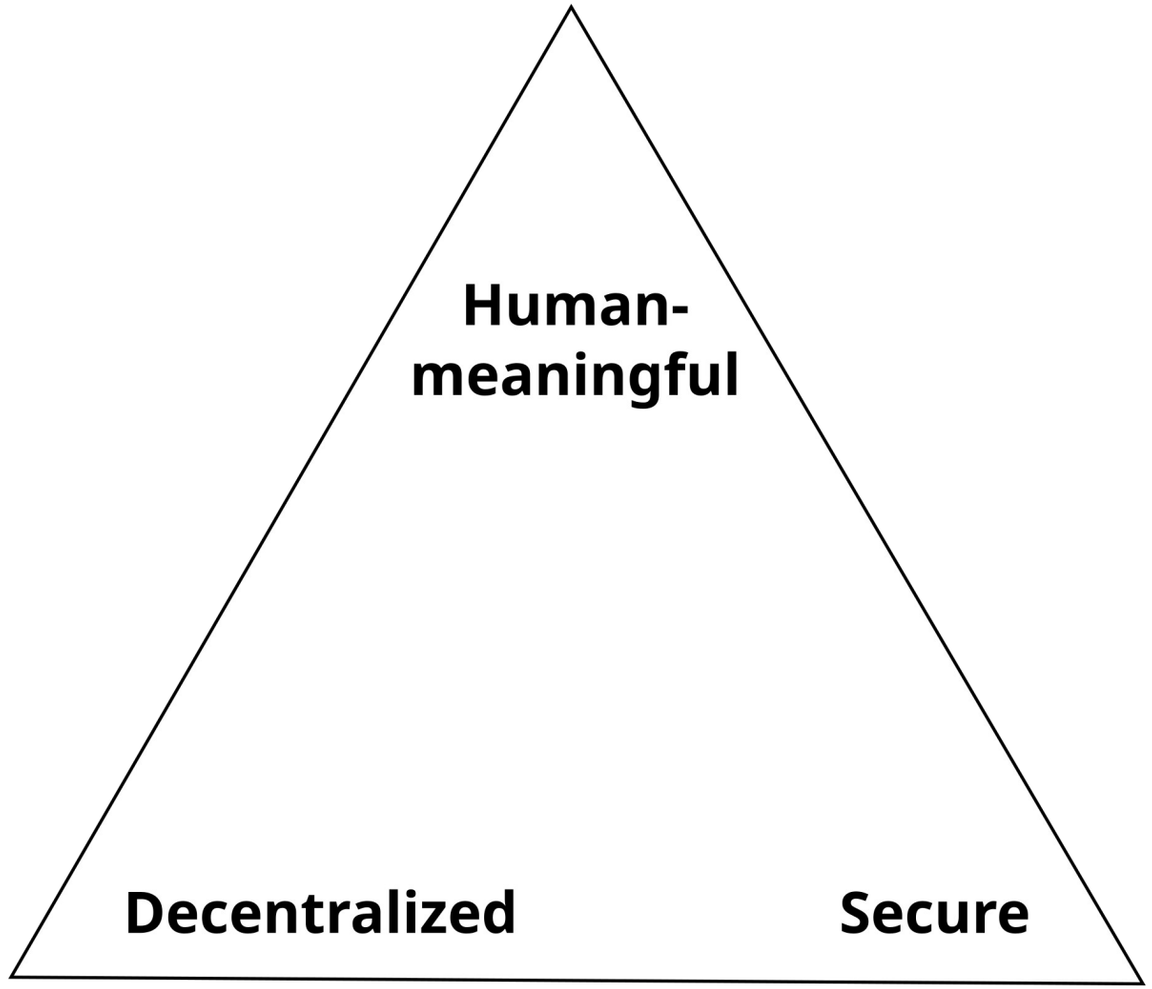
- Ad-hoc blockchain complementing IPFS by providing **incentives to store data**
- To store data persistently, you can pay some filecoin to ensure “miners” store your data for some time
- Based on crypto-magic (ZK-SNARKs, seminar opportunity for adventurous students):
  - Proof of space-time
  - Proof of replication

Ethereum Name Service

(thanks to Federico Fontana)

Zooko's Triangle

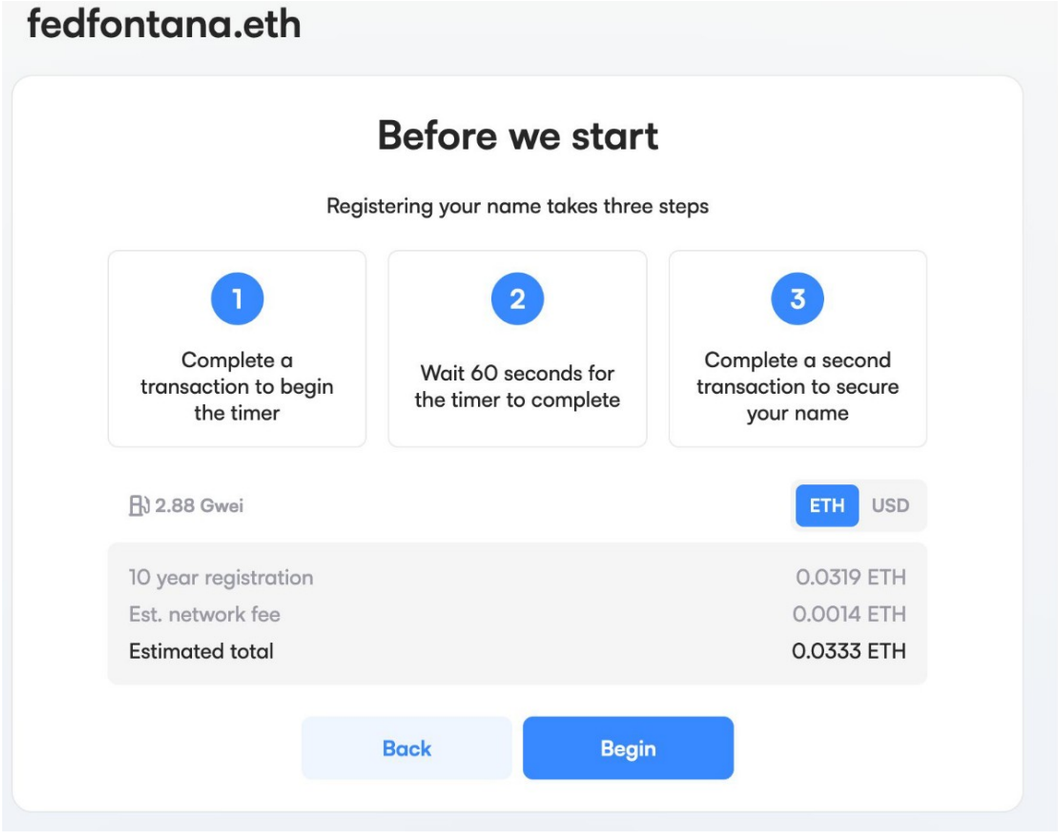
- A 2001 conjecture about the impossibility of having names that are at once **meaningful**, **decentralized** and **secure**
- Q: What about CIDs? Ethereum/Bitcoin addresses? DNSLink?
- It has been **proven false**
  - ENS is one of the solutions



Ethereum Name Service

- A way to associate **human-readable names** to **Ethereum addresses**
- Solves Zooko's triangle: decentralized, secure, meaningful
- Based on Ethereum smart contracts
- Contracts are updateable thanks to the proxy pattern (you'll see)

As a User: Registration Fee

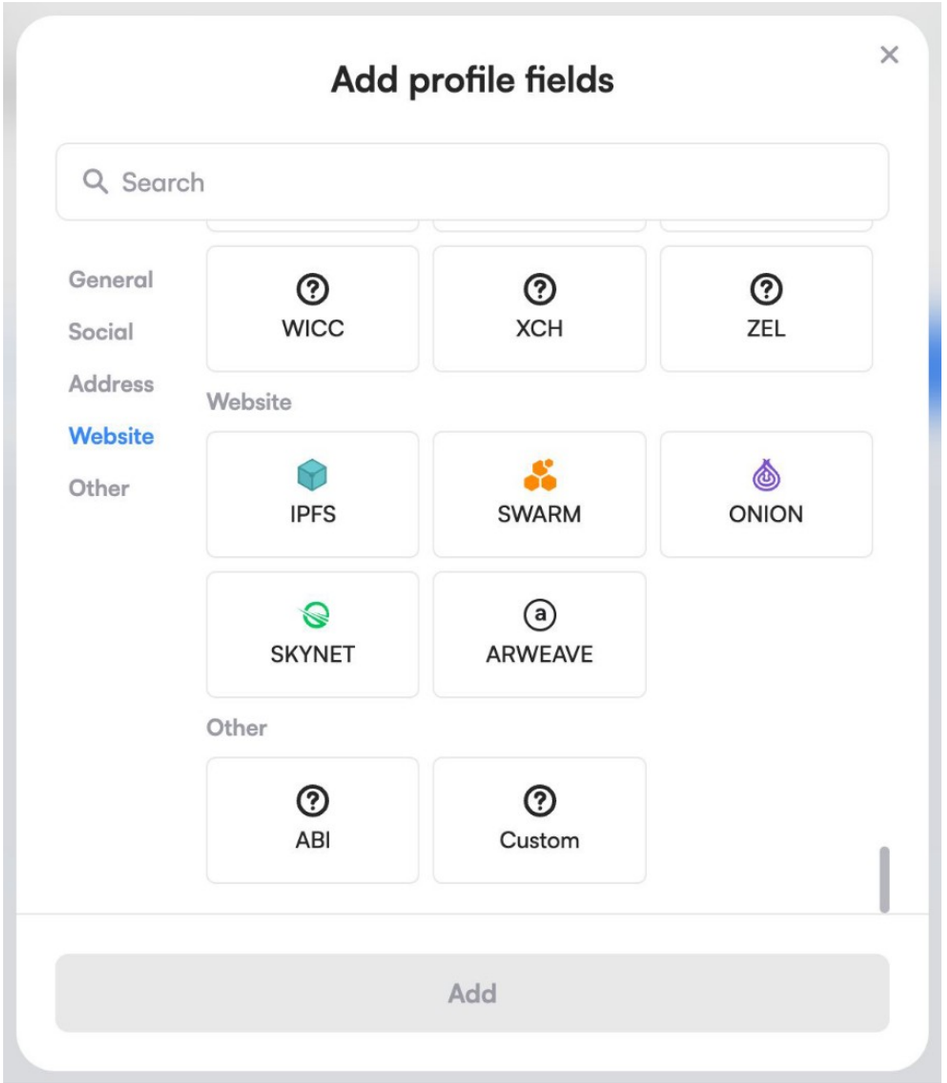


Source: Federico Fontana

- Prices vary (5-640US$ per year) depending on the **length of the name one wants to register**
- Expressed in dollars in the contracts thanks to **an oracle** (you'll see it in the next lesson!)
- Can be renewed until expiration (+ a grace period)

As a User: Fields

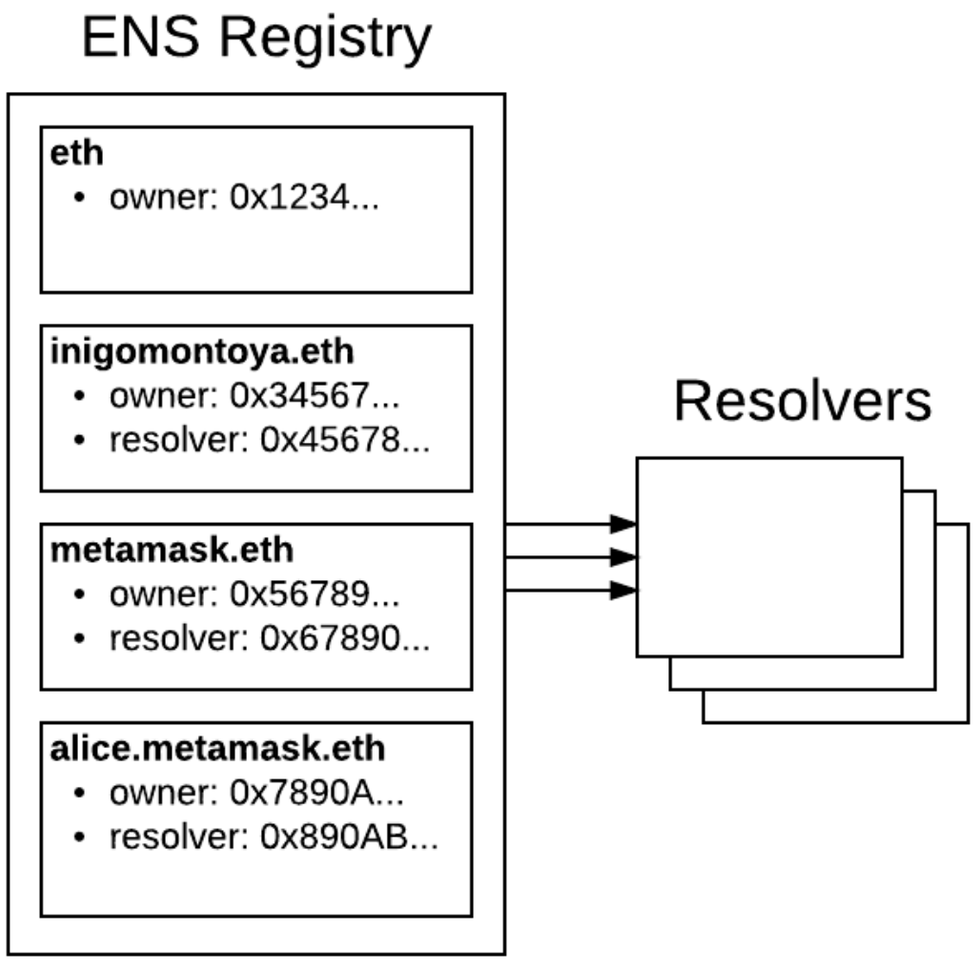
- As with DNS, you can insert **multiple fields** in your record
- Some are standardized, others just use free-form fields
  - Like DNSLink for IPFS+DNS



Source: Federico Fontana

Registry and Resolvers

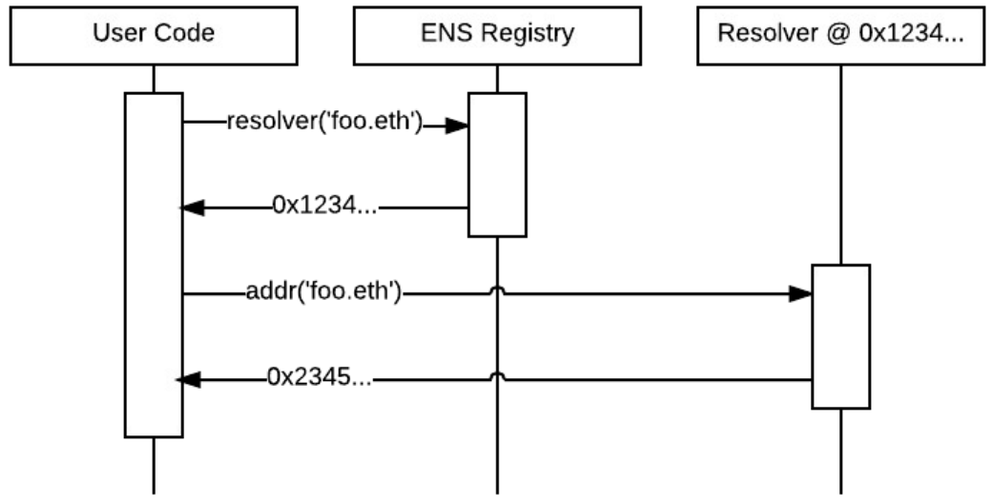
- A **single registry** which maps to **resolvers** for each subdomain
- Resolvers are **smart contracts** anyone can implement
  - They must implement at least the addr method



https://ens.readthedocs.io/en/stable/introduction.html

ENS Queries

- The client calls the **registry** to get the resolver's address
- The resolver answers the queries



https://ens.readthedocs.io/en/stable/implementers.html

Registry

- Stores mapping between names and resolver, owner and caching time-to-live (TTL)
- Emits events such as `NewOwner`, `Transfer`, `NewResolver`
- Interacts with another contract, the *Controller*, responsible for registration, fees, renewal
  - Previously (up to 2020): auction model, no renewal
  - Now: commit-reveal pattern to avoid frontrunning

Other Facilities

- NameWrapper: allows delegating a subdomain, but limiting functionalities (e.g., further transfers, change records, create subdomains)
- Reverse registrar: address `0x<addr>` can claim the `<addr>.addr.reverse` name (and optionally transfer it)
- DNS in ENS: thanks to another oracle, one can use their DNSSEC (DNS+crypto extensions) name online