Blockchains and the Web

IPFS

W3C®

https://ipfs.io
/ipns/ipfs.io

@juanbenet
2016-06-29

0. IPFS
1. multi formats
2. IPLD
3. libp2p

0. IPFS
1. multi formats
2. IPLD
3. libp2p

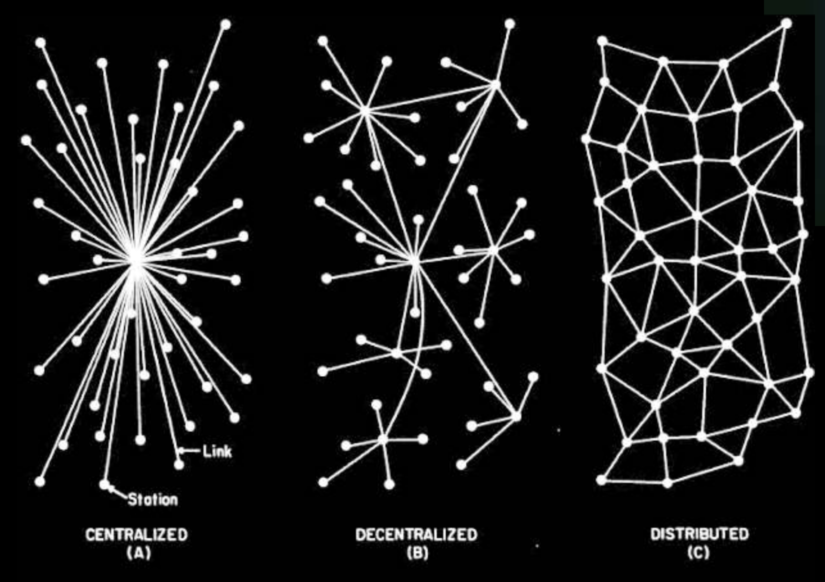
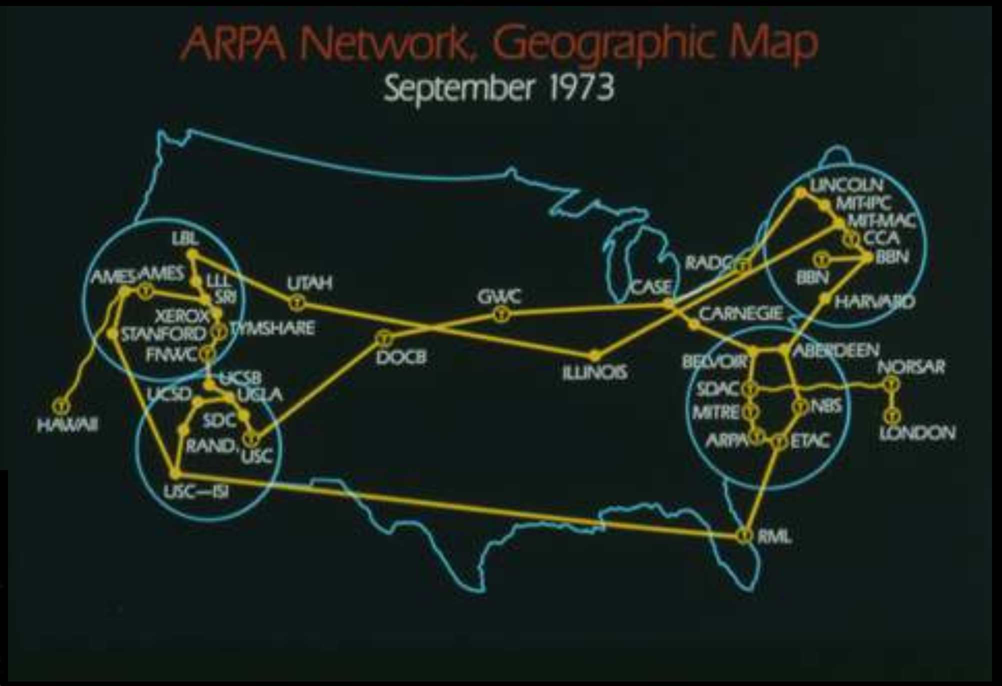
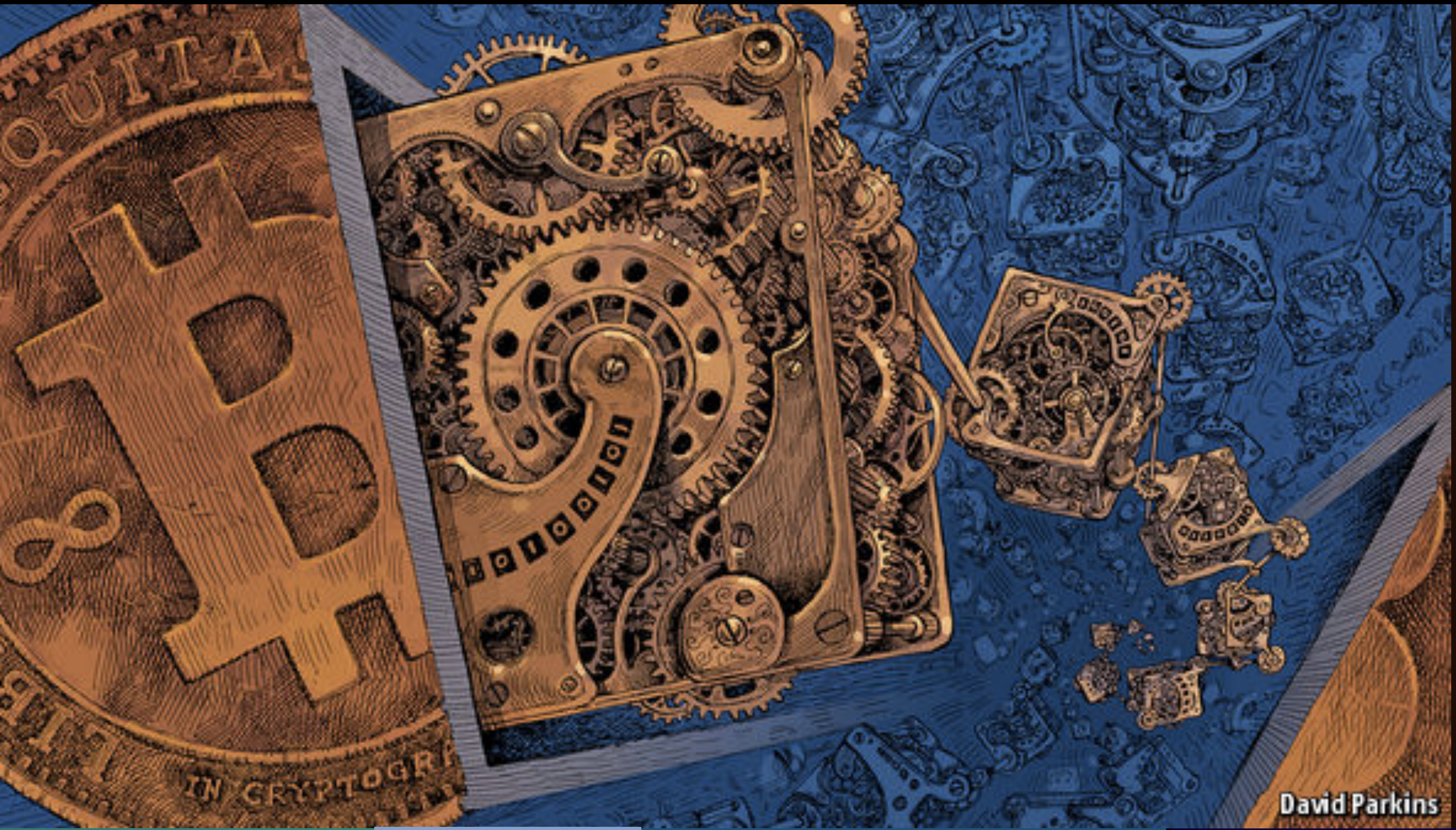
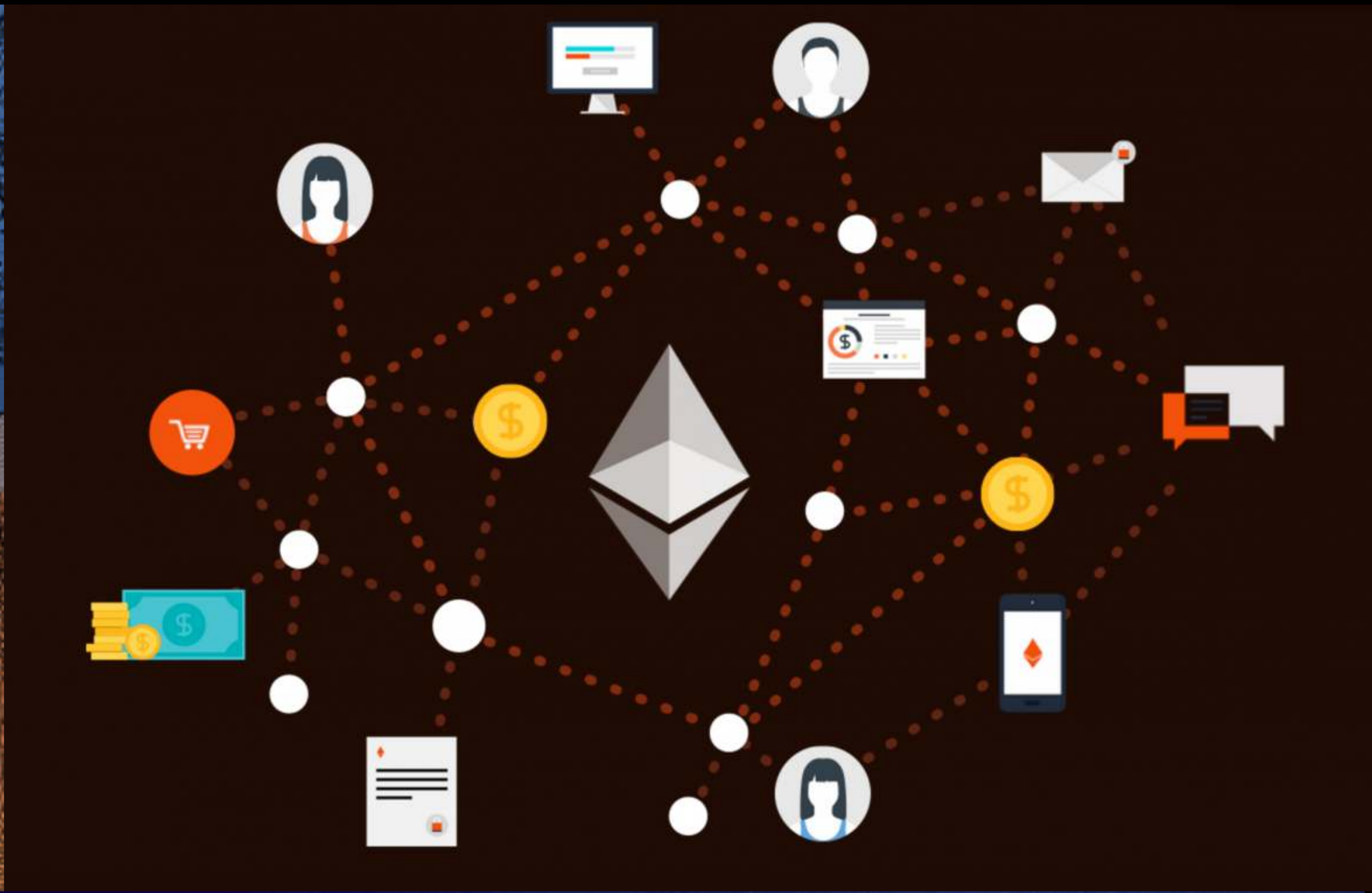


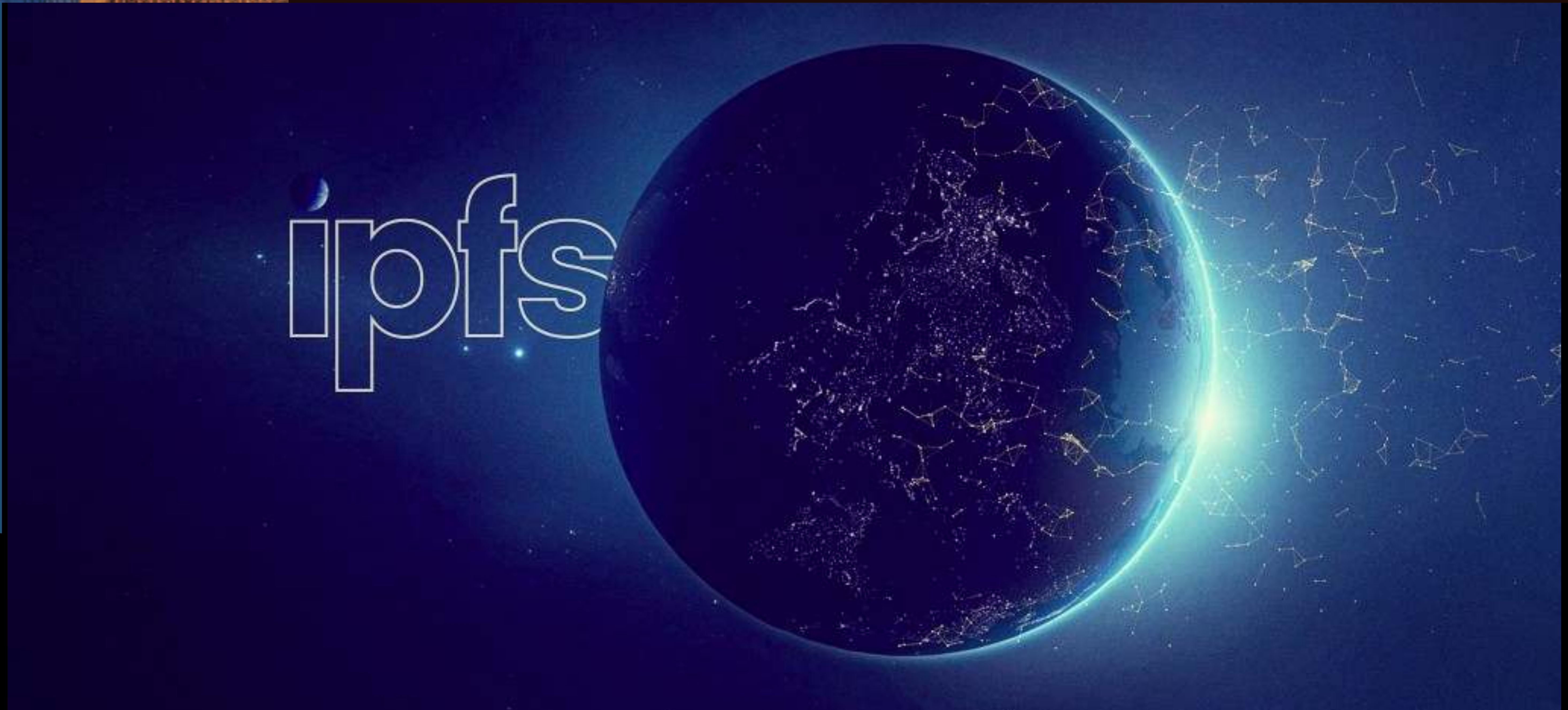
FIG. I — Centralized, Decentralized and Distributed Networks



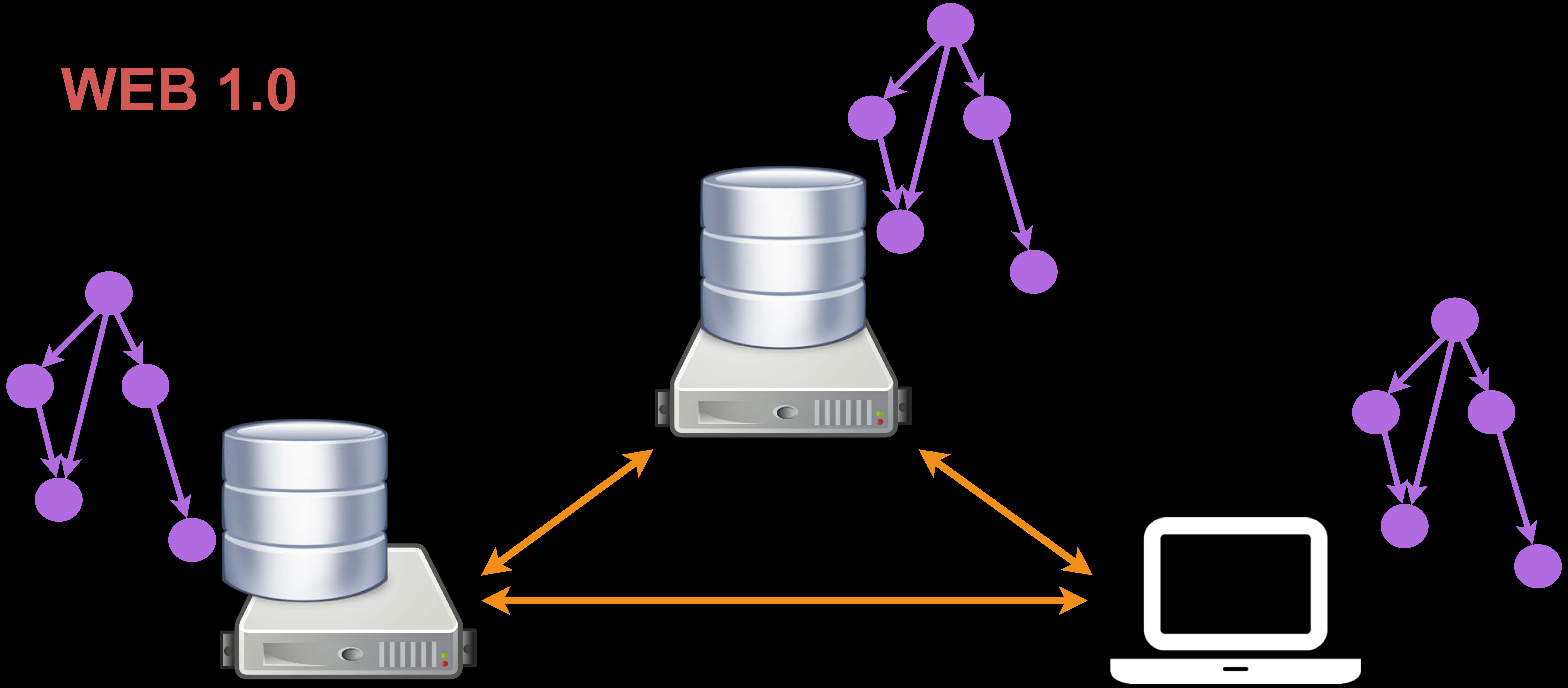


David Parkins

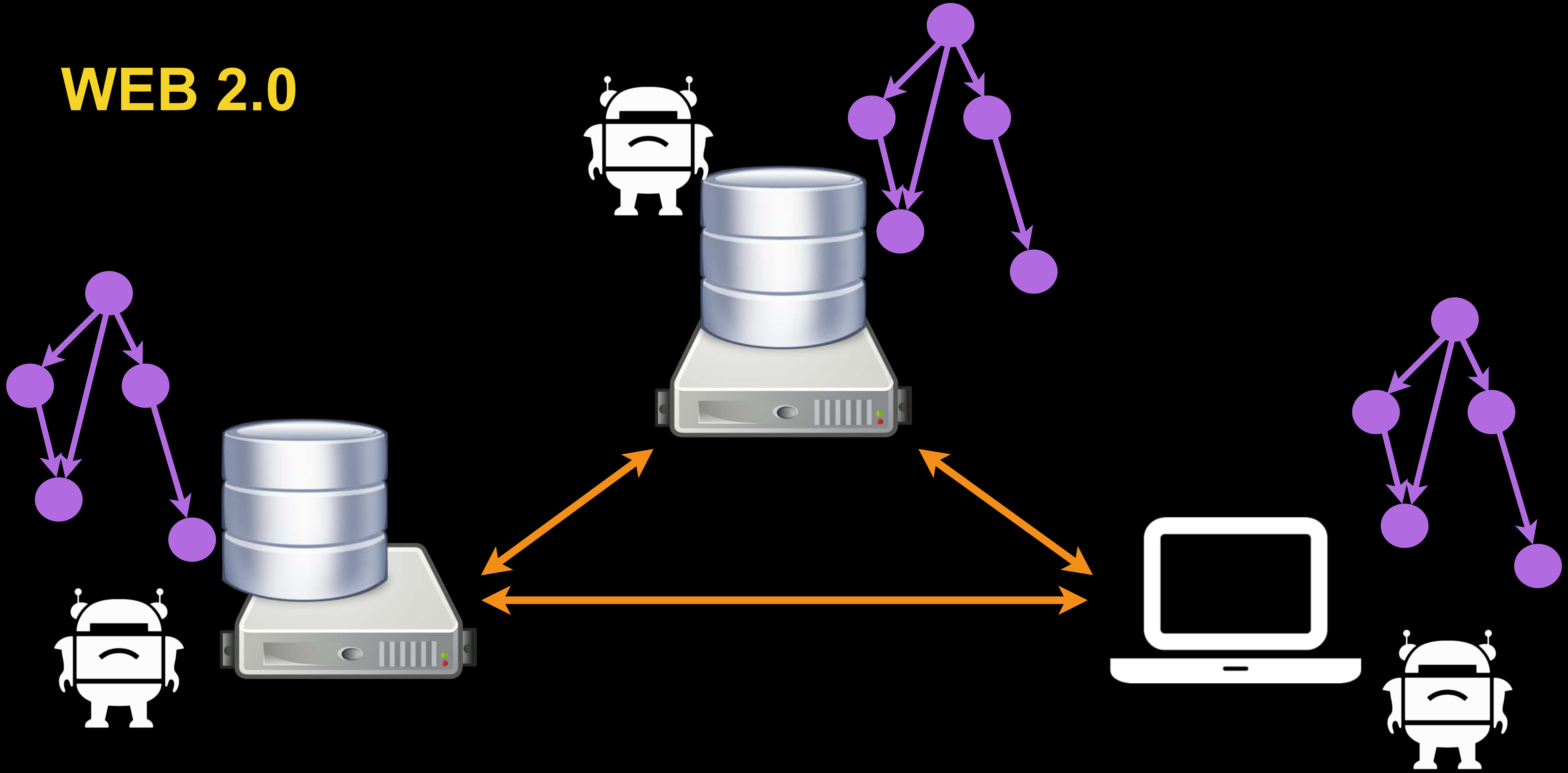




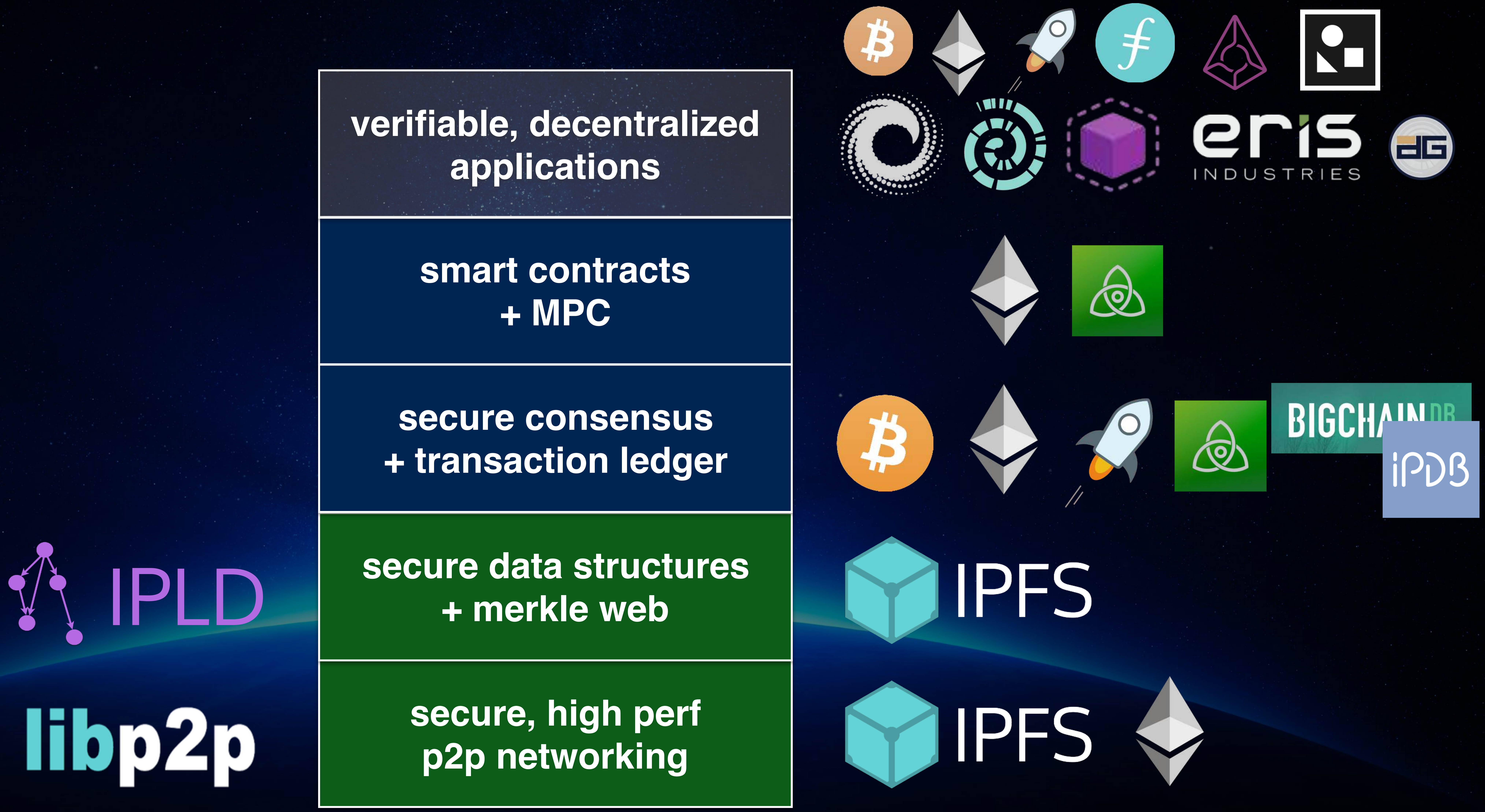
WEB 1.0



WEB 2.0



WEB 3.0



verifiable, decentralized applications

smart contracts + MPC

secure consensus + transaction ledger

secure data structures + merkle web

secure, high perf p2p networking

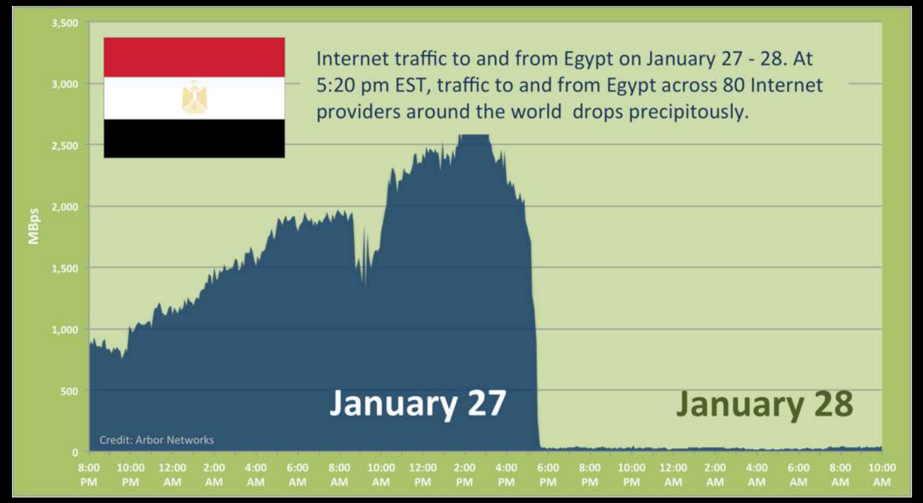
IPFS

a new hypermedia distribution protocol

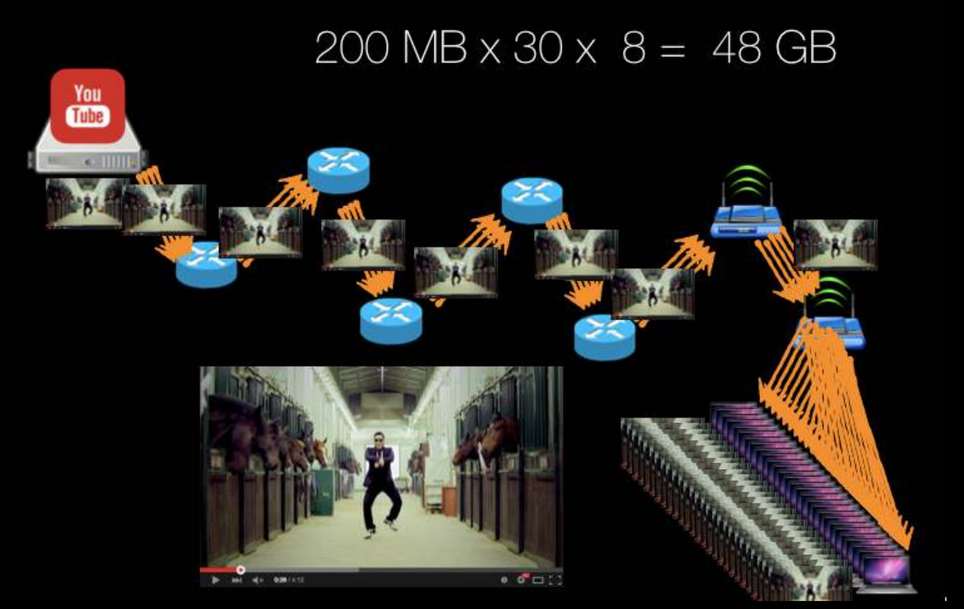
(a new web transport protocol)

the web has problems

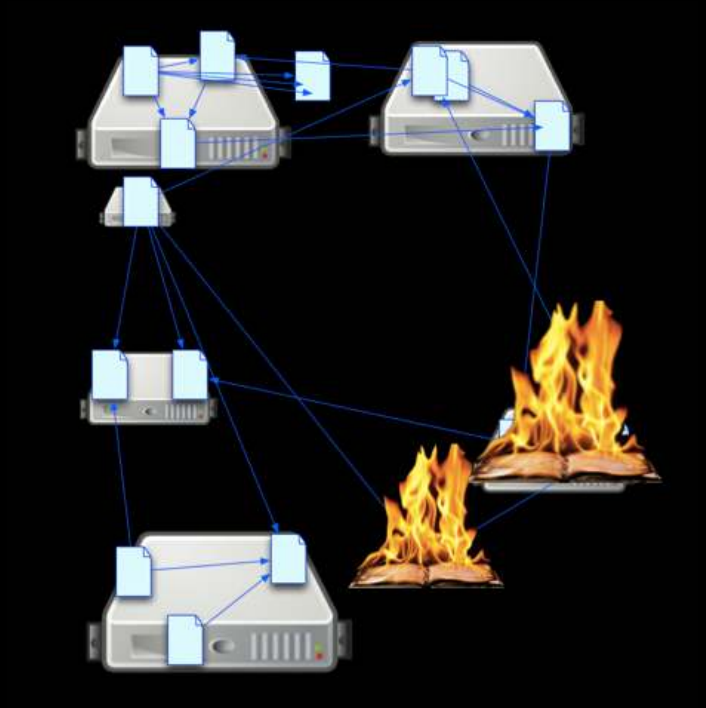
bad in mobile and IoT



censorship



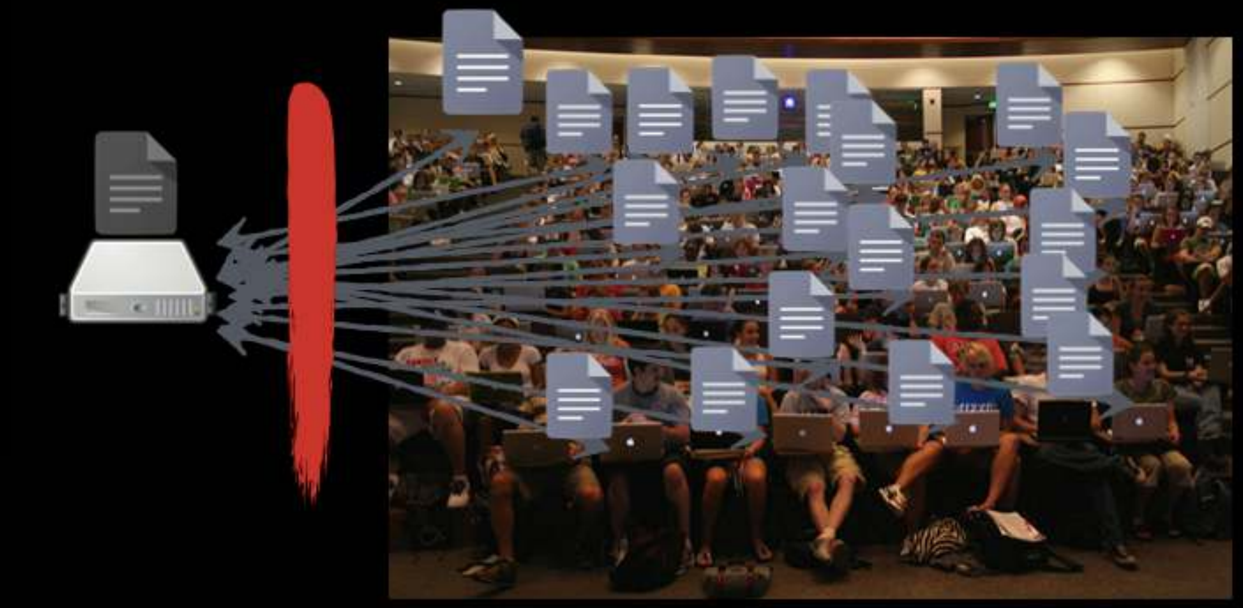
huge inefficiencies



links break

AUTHENTICATED
& ENCRYPTED
AT REST

bad security model



no offline use

large open source project

400+ contributors

70+ contribute weekly

The IPFS Stack

applications

Using the Data

IPNS

IPLD

naming

merkledag

Defining the Data

libp2p

exchange

routing

network

Moving the Data

applications: Etherpad, Websites, Orbit, Mediachain, uPort

naming: Blockstack, DNS, IPNS, Namecoin, EthNames

IPLD

exchange: BitTorrent, Bitswap, FTP, HTTP

routing: Gossip, Chord, Kad DHT, mDNS, Delegated, I2P, TOR

network: CJDNS, UDT, uTP, WebRTC, QUIC, TCP, WebSockets, I2P, TOR

dns name

/dns/example.com/foo/bar/baz.png
/ipns/example.com/foo/bar/baz.png

↓

key name

/ipns/QmYJPtosPTfoC/foo/bar/baz.png

↓

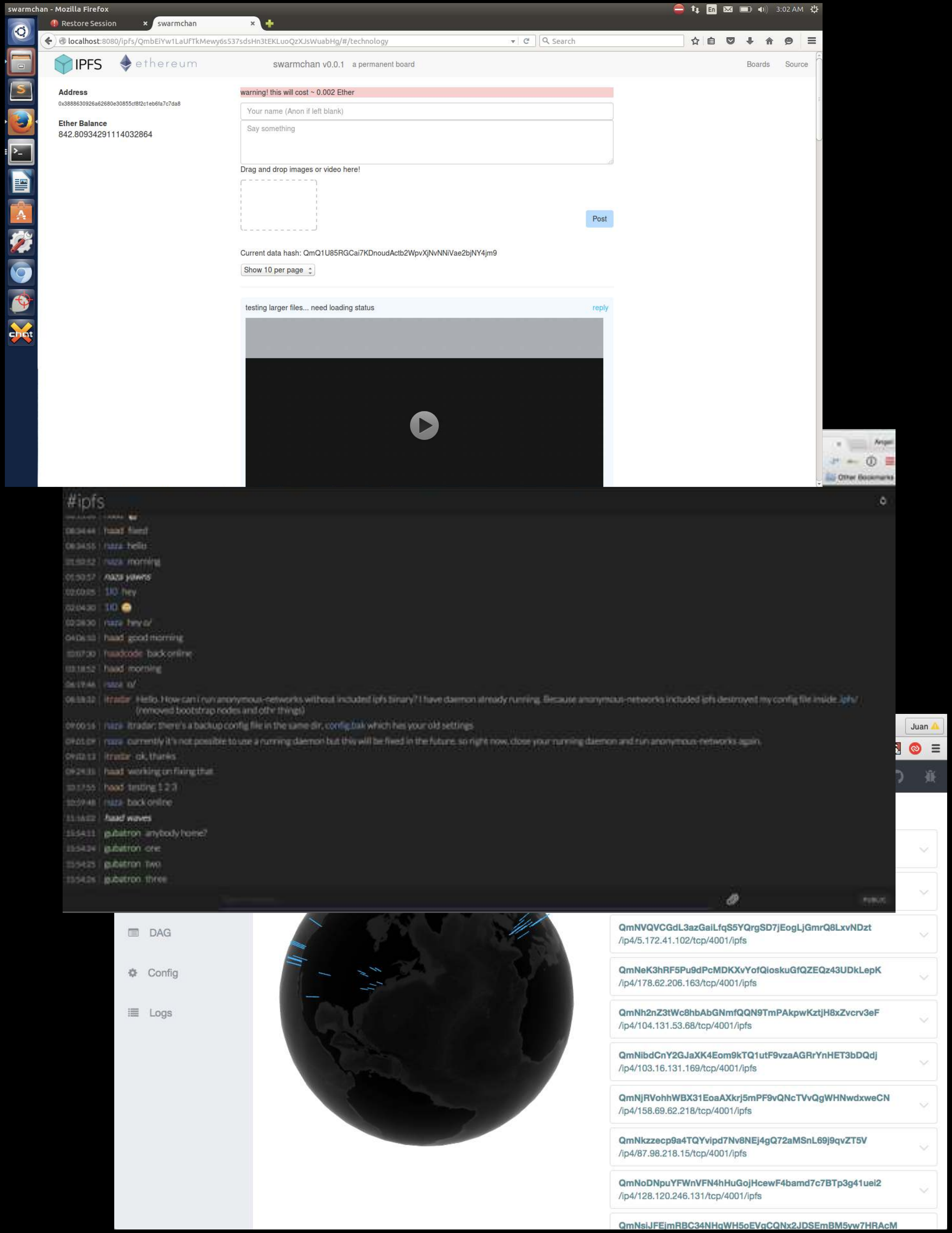
content addr

/ipfs/QmW98pJrc6FZ6/foo/bar/baz.png
fs:/ipfs/QmW98pJrc6FZ6/foo/bar/baz.png
ipfs:/ipfs/QmW98pJrc6FZ6/foo/bar/baz.png

distributed webapps

- app code stored + distributed with ipfs
- app data stored + distributed with ipfs
- browsers can connect to each other
- no origin servers!
- no central point of failure
- everything end-to-end encrypted
- app "lives on the network"

examples: forums, chat, messaging,
cms, blogs, github, ...


swarmchan - Mozilla Firefox
Restore Session
swarmchan
3:02 AM
localhost:8080/ipfs/QmbEiYw1LaUfTkMewy6s537sdsHn3tEKLuoQzXJsWuabHg/#/technology
Search
IPFS
ethereum
swarmchan v0.0.1
a permanent board
Boards
Source

Address
0x3888630926a62680e30855cf8f2c1eb6fa7c7da8

Ether Balance
842.80934291114032864

warning! this will cost ~ 0.002 Ether

Your name (Anon if left blank)

Say something

Drag and drop images or video here!

Post

Current data hash: QmQ1U85RGCai7KDnoudActb2WpvXjNvNNiVae2bjNY4jm9

Show 10 per page

testing larger files... need loading status

reply

#ipfs

```
08:34:44 | haad fixed
08:34:55 | nazza hello
01:50:52 | nazza morning
01:50:57 | nazza yawns
02:00:05 | 110 hey
02:04:30 | 110 😊
02:28:30 | nazza hey o/
04:06:53 | haad good morning
10:07:30 | haadcode back online
03:18:52 | haad morning
06:19:46 | nazza o/
06:58:32 | itradar Hello. How can i run anonymous-networks without included ipfs binary? I have daemon already running. Because anonymous-networks included ipfs destroyed my config file inside .ipfs/
           (removed bootstrap nodes and othr things)
09:00:16 | nazza itradar: there's a backup config file in the same dir, config.bak which has your old settings
09:01:09 | nazza currently it's not possible to use a running daemon but this will be fixed in the future. so right now, close your running daemon and run anonymous-networks again.
09:02:13 | itradar ok, thanks
09:24:31 | haad working on fixing that
10:17:55 | haad testing 1 2 3
10:59:48 | nazza back online
11:16:22 | haad waves
15:54:11 | gubatron anybody home?
15:54:24 | gubatron one
15:54:25 | gubatron two
15:54:26 | gubatron three
```

QmNVQVCGdL3azGaiLfqS5YQrgSD7jEogLjGmrQ8LxvNDzt
/ip4/5.172.41.102/tcp/4001/ipfs

QmNeK3hRF5Pu9dPcMDKXvYofQioskuGfQZEQz43UDkLepK
/ip4/178.62.206.163/tcp/4001/ipfs

QmNh2nZ3tWc8hbAbGNmfQQN9TmPAkpwKztjH8xZvcrv3eF
/ip4/104.131.53.68/tcp/4001/ipfs

QmNibdCnY2GJaXK4Eom9kTQ1utF9vzaAGRrYnHET3bDQdj
/ip4/103.16.131.169/tcp/4001/ipfs

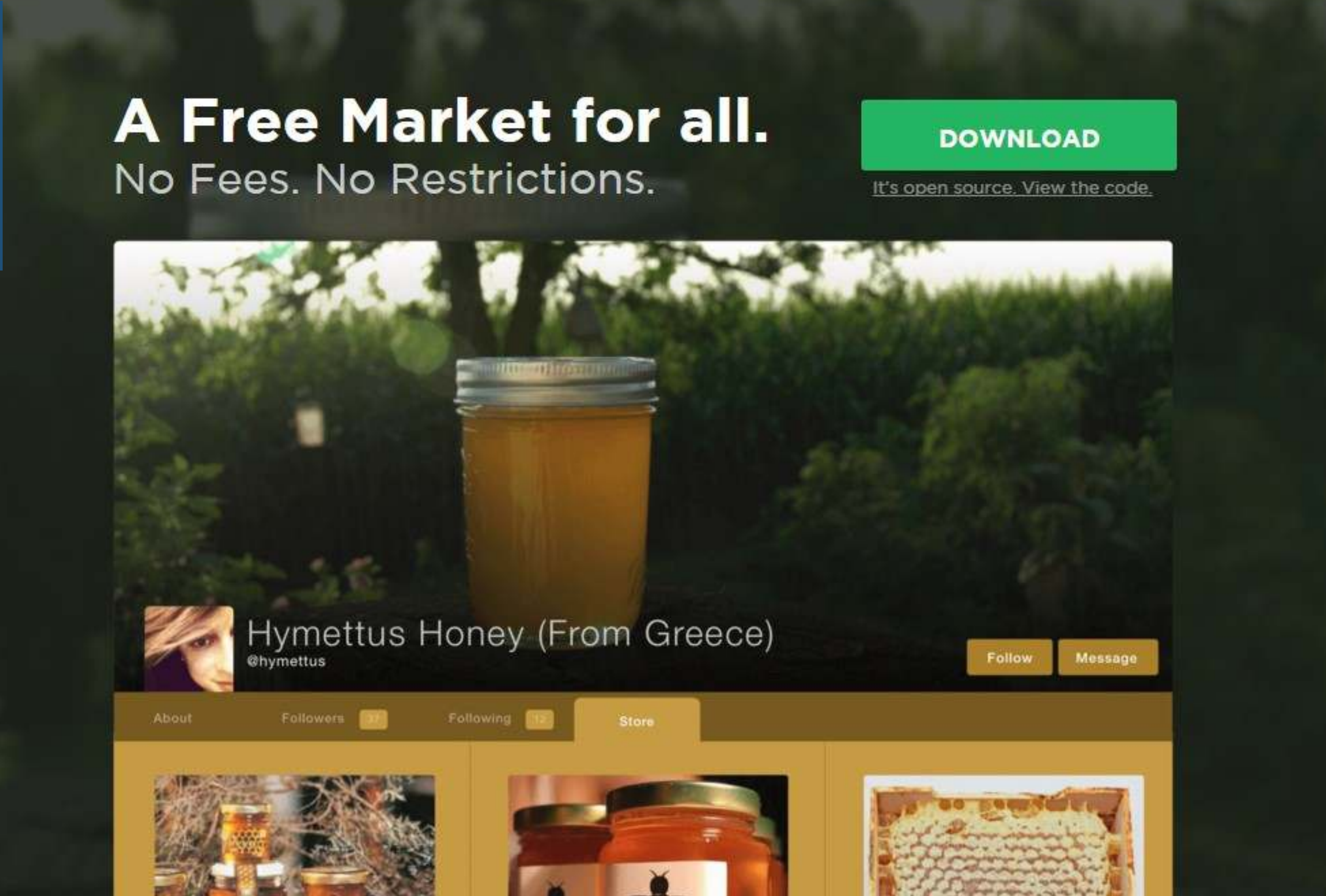
QmNjRVohhWBX31EoaAXkrj5mPF9vQNcTVvQgWHNwdxweCN
/ip4/158.69.62.218/tcp/4001/ipfs

QmNkzzecp9a4TQYvipd7Nv8NEj4gQ72aMSnL69j9qvZT5V
/ip4/87.98.218.15/tcp/4001/ipfs

QmNoDNpuYFWnVFN4hHuGojHcewF4bamd7c7BTp3g41uei2
/ip4/128.120.246.131/tcp/4001/ipfs

QmNsiJFEjmRBC34NHqWH5oEVgCQNx2JDSEmBM5yw7HRAcM

OpenBazaar



A Free Market for all.

No Fees. No Restrictions.

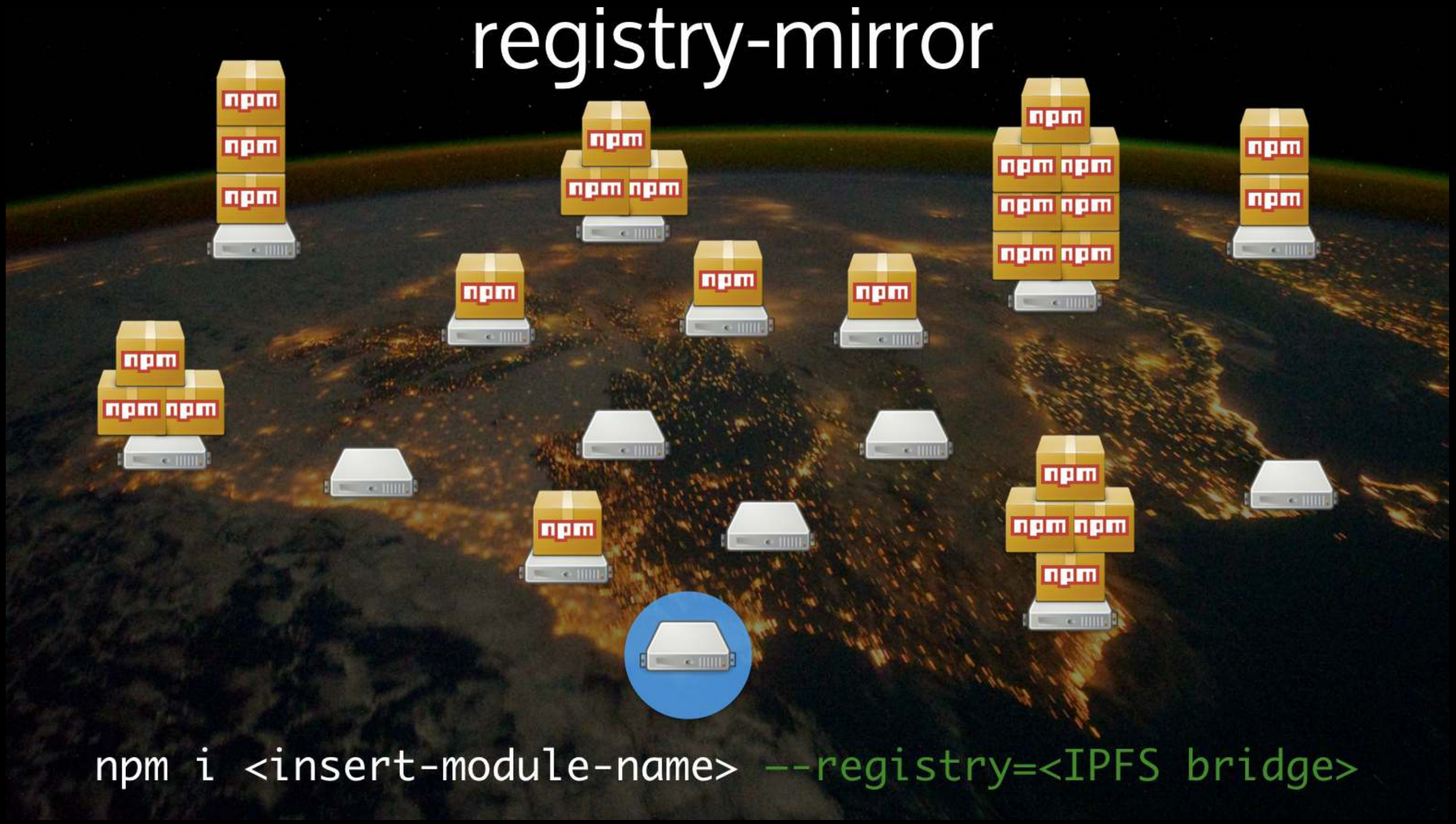
It's open source. View the code.

Hymettus Honey (From Greece)

@hymettus

used for package managers

- distributed / peer-to-peer
- cryptographically verified links
- digitally signed links
- "everyone is a mirror"
- save lots of bandwidth
- versioning built in

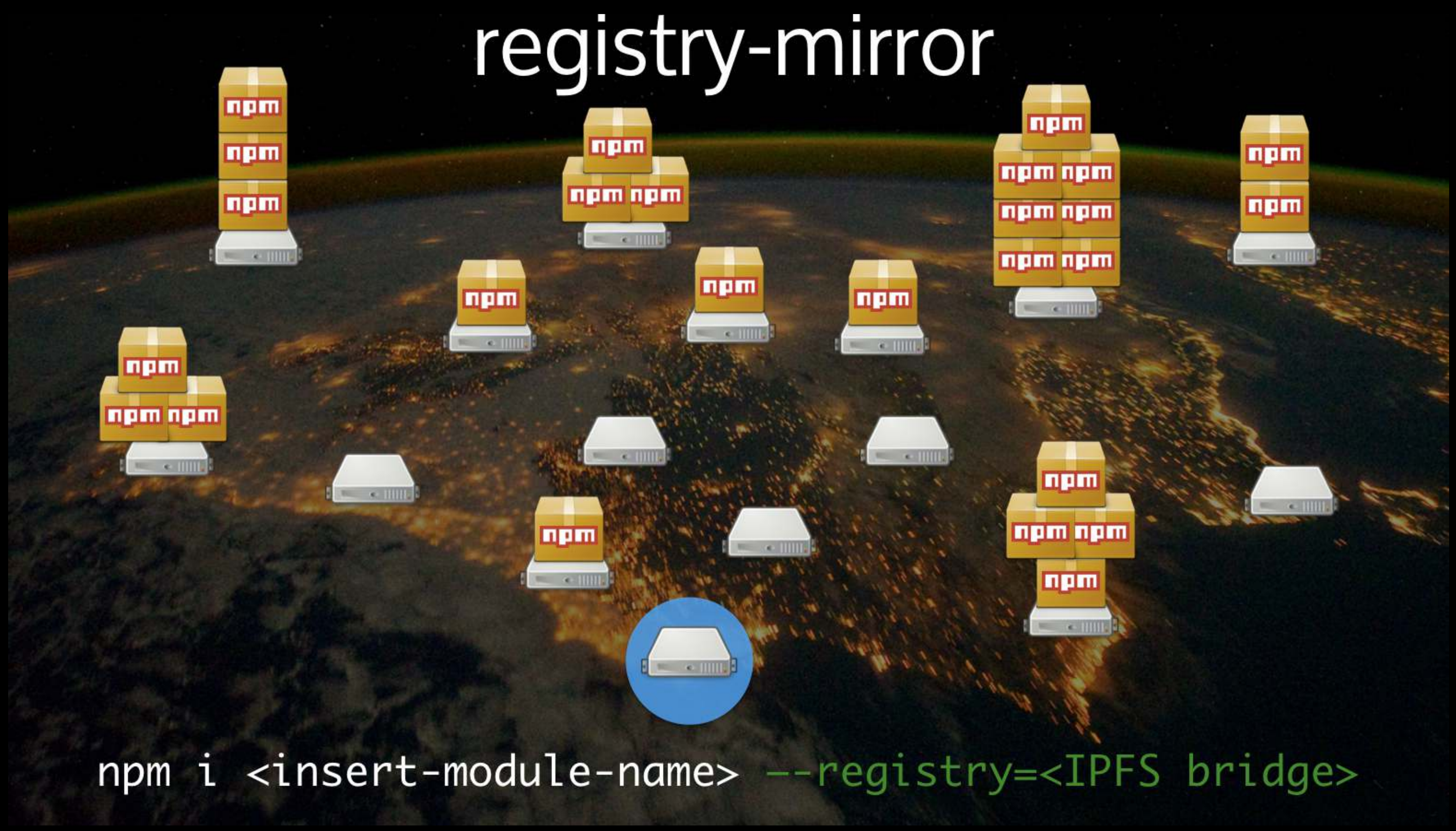


```
npm i <insert-module-name> --registry=<IPFS bridge>
```

npm on ipfs

github.com/diasdavid/registry-mirror

works with vanilla npm

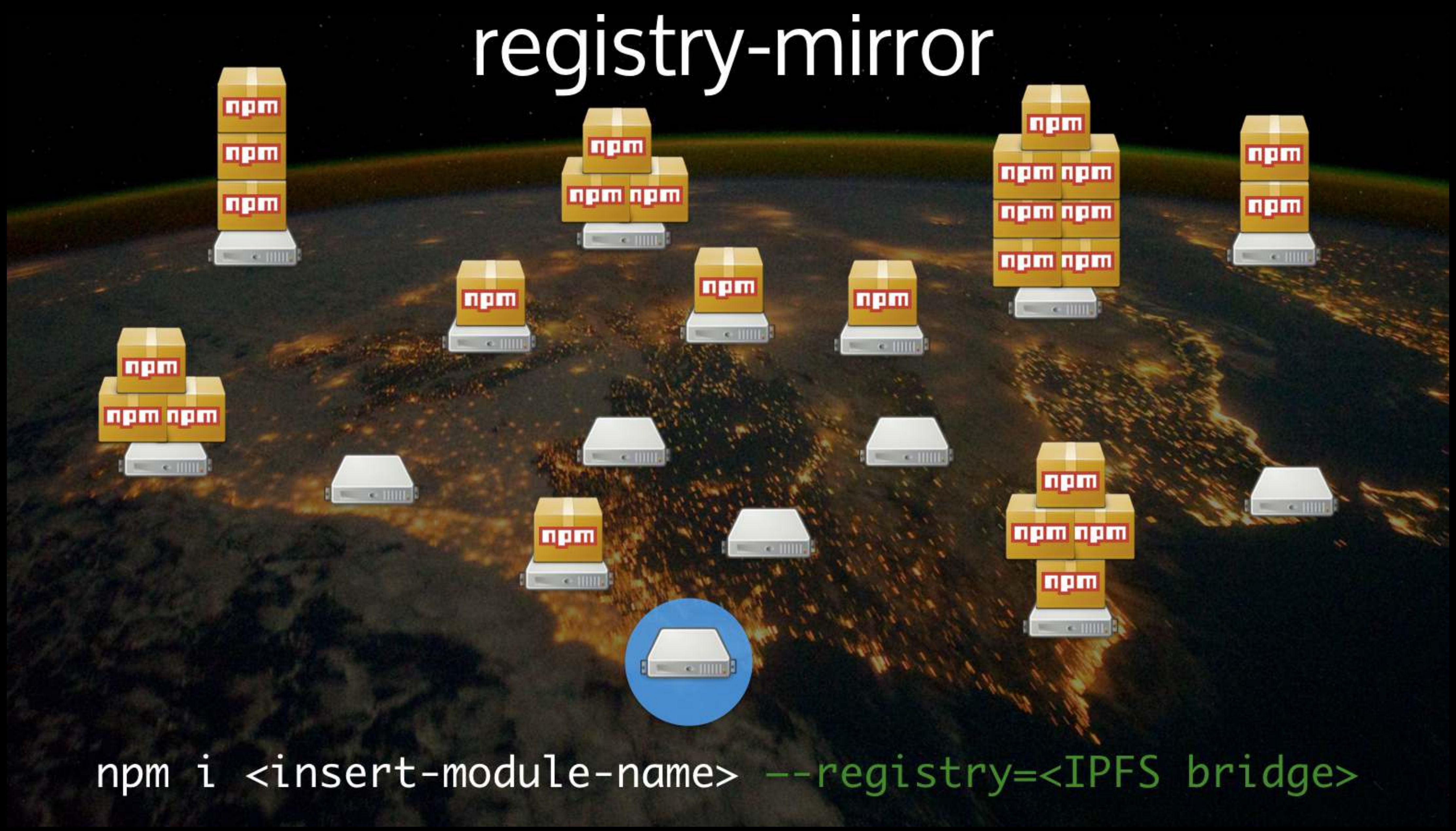


```
npm i <insert-module-name> --registry=<IPFS bridge>
```

npm on ipfs

github.com/diasdavid/registry-mirror

works with vanilla npm



```
npm i <insert-module-name> --registry=<IPFS bridge>
```

gx

github.com/whyrusleeping/gx

extensible pkg mgr

gx-go for Golang!

used for secure documents

- content addressed hash links
- digitally signed links
- trustless ledgers
- permanent links
- secure document web

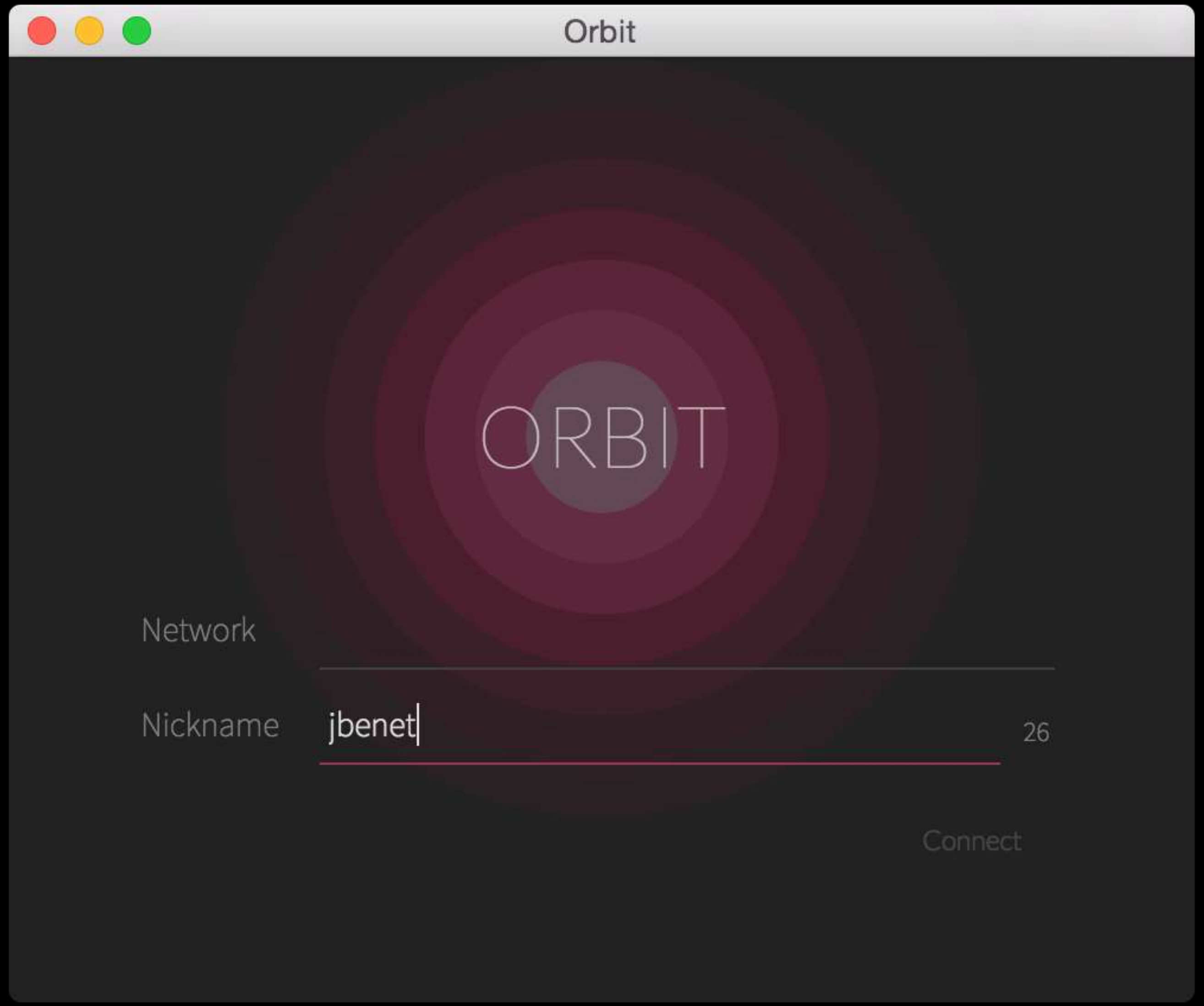
already in use at:

- **banks**
- legal archives
- blockchain companies
- smart contract apps

Orbit

github.com/haadcode/orbit

p2p chat on IPFS



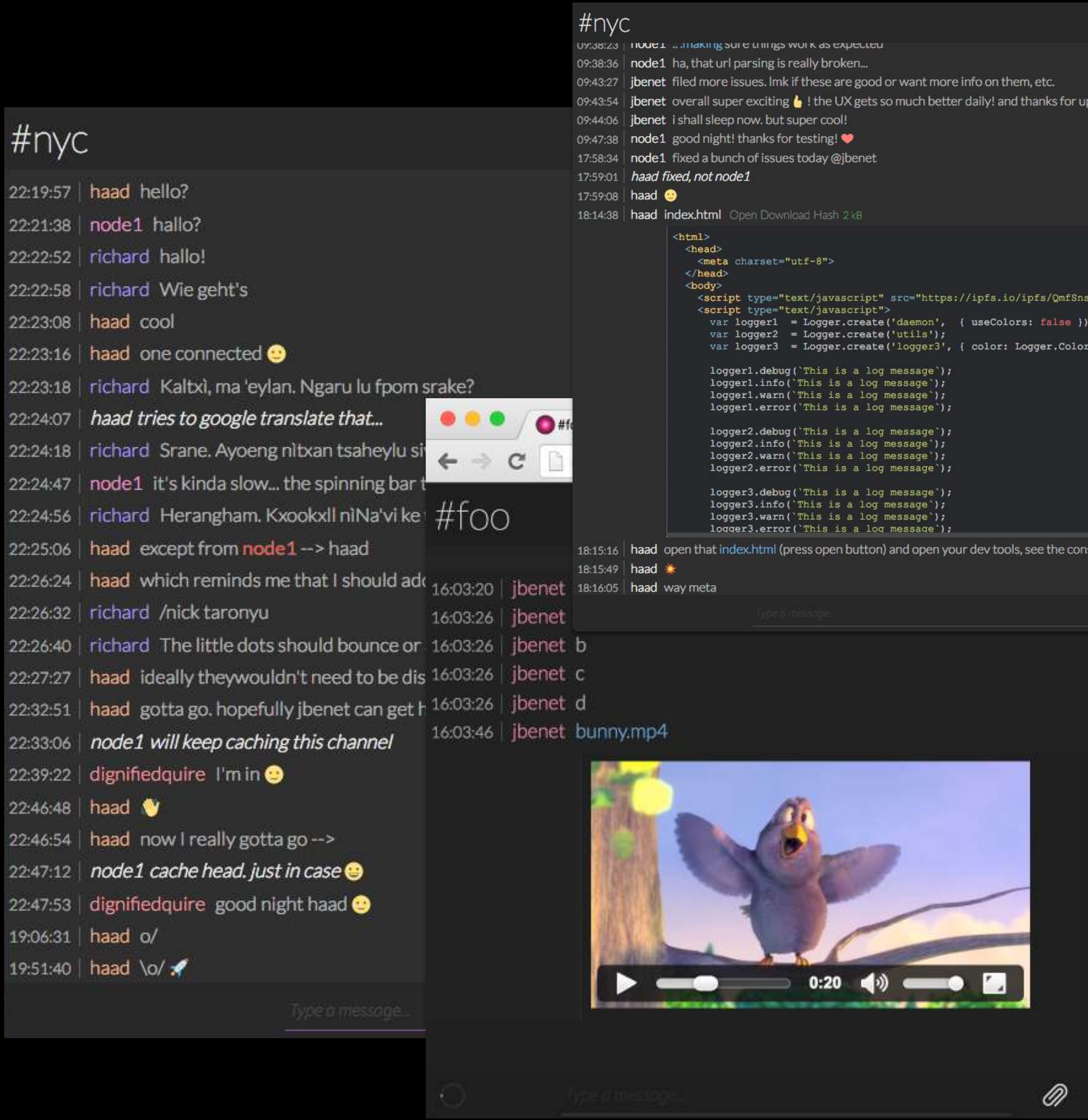
Network

Nickname jbenet 26

IPFS

IPLD

libp2p



#nyc

```
22:19:57 | haad hello?
22:21:38 | node1 hallo?
22:22:52 | richard hallo!
22:22:58 | richard Wie geht's
22:23:08 | haad cool
22:23:16 | haad one connected 🙂
22:23:18 | richard Kaltxì, ma 'eylan. Ngaru lu fpom srake?
22:24:07 | haad tries to google translate that...
22:24:18 | richard Srane. Ayoeng nìtxan tsaheylu si
22:24:47 | node1 it's kinda slow... the spinning bar t
22:24:56 | richard Herangham. Kxookxll niNa'vi ke
22:25:06 | haad except from node1 --> haad
22:26:24 | haad which reminds me that I should ad
22:26:32 | richard /nick taronyu
22:26:40 | richard The little dots should bounce or
22:27:27 | haad ideally theywouldn't need to be dis
22:32:51 | haad gotta go. hopefully jbenet can get h
22:33:06 | node1 will keep caching this channel
22:39:22 | dignifiedquire I'm in 🙂
22:46:48 | haad 👋
22:46:54 | haad now I really gotta go -->
22:47:12 | node1 cache head. just in case 🙂
22:47:53 | dignifiedquire good night haad 🙂
19:06:31 | haad o/
19:51:40 | haad \o/ 🚀
```

Type a message...

#foo

```
16:03:20 | jbenet
16:03:26 | jbenet
16:03:26 | jbenet b
16:03:26 | jbenet c
16:03:26 | jbenet d
16:03:46 | jbenet bunny.mp4
```

#nyc

```
09:38:23 | node1 ...making sure things work as expected
09:38:36 | node1 ha, that url parsing is really broken...
09:43:27 | jbenet filed more issues. lmk if these are good or want more info on them, etc.
09:43:54 | jbenet overall super exciting 👍 ! the UX gets so much better daily! and thanks for u
09:44:06 | jbenet i shall sleep now. but super cool!
09:47:38 | node1 good night! thanks for testing! ❤
17:58:34 | node1 fixed a bunch of issues today @jbenet
17:59:01 | haad fixed, not node1
17:59:08 | haad 🙂
18:14:38 | haad index.html Open Download Hash 2 kB
```

```
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <script type="text/javascript" src="https://ipfs.io/ipfs/QmfSns
    <script type="text/javascript">
      var logger1  = Logger.create('daemon',  { useColors: false })
      var logger2  = Logger.create('utils');
      var logger3  = Logger.create('logger3', { color: Logger.Color

      logger1.debug(`This is a log message`);
      logger1.info(`This is a log message`);
      logger1.warn(`This is a log message`);
      logger1.error(`This is a log message`);

      logger2.debug(`This is a log message`);
      logger2.info(`This is a log message`);
      logger2.warn(`This is a log message`);
      logger2.error(`This is a log message`);

      logger3.debug(`This is a log message`);
      logger3.info(`This is a log message`);
      logger3.warn(`This is a log message`);
      logger3.error(`This is a log message`);
```

```
18:15:16 | haad open that index.html (press open button) and open your dev tools, see the con
18:15:49 | haad ✴
18:16:05 | haad way meta
```

Type a message...

0. IPFS
1. multi formats
2. IPLD
3. libp2p

multiformats - self describing values

protocol agility, interop, avoid lock in

multihash

multiaddr

multibase

multicodec

multistream

multikey

multiformats - self describing values

protocol agility, interop, avoid lock in

multihash - cryptographic hashes

multiaddr

multibase

multicodec

multistream

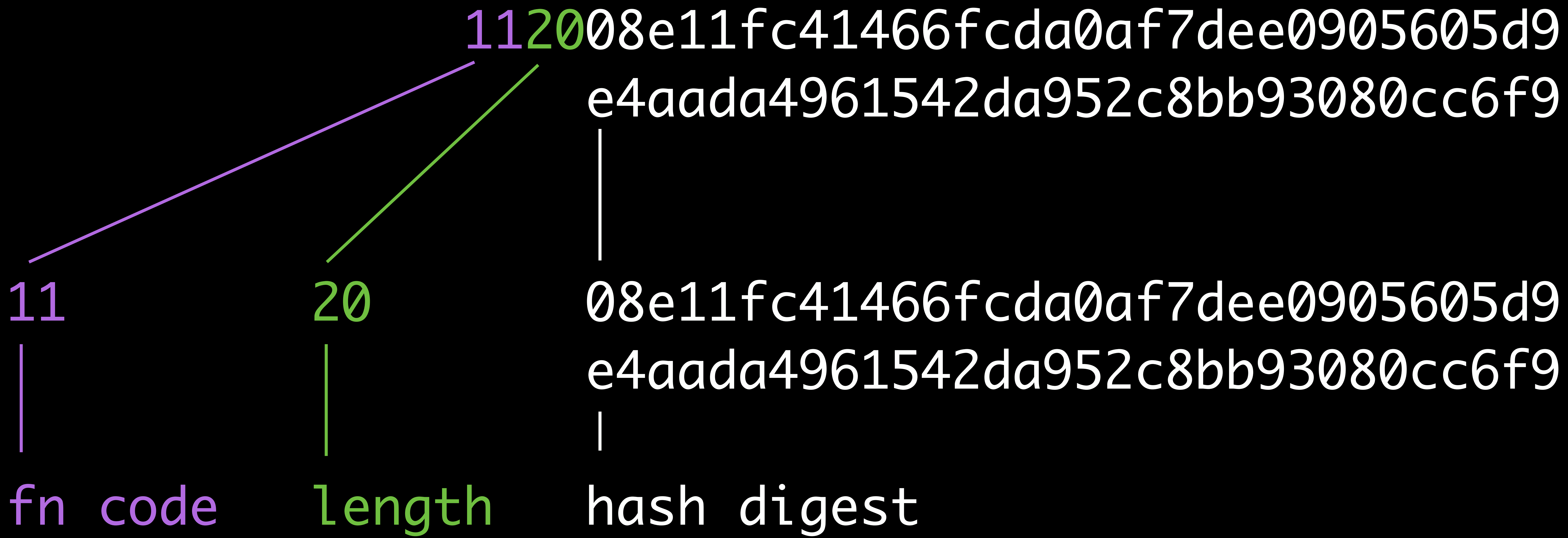
multikey

```
0x08e11fc41466fcda0af7dee0905605d9
  e4aada4961542da952c8bb93080cc6f9

0x95a1b32bd70332e24f63f3802aae5f5e
  1fa4622cc72750e0073bbbb6dcf6fce7

0xcaadb37a46daeda4e0d5e61574a9aaca
  211d513806a026e6cc4461f7ba7867f9

0x08fbea061a5dea457d69fe5c12575c1d
  9d30c49f575936f6e1c6d4ea0ab078df
```

256	0x08e11fc41466fcda0af7dee0905605d9e4aada4961542da952c8bb93080cc6f9
256	0x95a1b32bd70332e24f63f3802aae5f5e1fa4622cc72750e0073bbbb6dcf6fce7
256	0xcaadb37a46daeda4e0d5e61574a9aaca211d513806a026e6cc4461f7ba7867f9
256	0x08fbea061a5dea457d69fe5c12575c1d9d30c49f575936f6e1c6d4ea0ab078df

```
sha2 256    256       0x08e11fc41466fcda0af7dee0905605d9
                        e4aada4961542da952c8bb93080cc6f9

sha2 512    256       0x95a1b32bd70332e24f63f3802aae5f5e
                        1fa4622cc72750e0073bbbb6dcf6fce7

sha3        256       0xcaadb37a46daeda4e0d5e61574a9aaca
                        211d513806a026e6cc4461f7ba7867f9

blake2b     256       0x08fbea061a5dea457d69fe5c12575c1d
                        9d30c49f575936f6e1c6d4ea0ab078df
```

```
sha2 256    256     112008e11fc41466fcda0af7dee0905605d9
                        e4aada4961542da952c8bb93080cc6f9


sha2 512    256     122095a1b32bd70332e24f63f3802aae5f5e
                        1fa4622cc72750e0073bbbb6dcf6fce7


sha3        256     1420caadb37a46daeda4e0d5e61574a9aaca
                        211d513806a026e6cc4461f7ba7867f9


blake2b     256     402008fbea061a5dea457d69fe5c12575c1d
                        9d30c49f575936f6e1c6d4ea0ab078df
```



112008e11fc41466fcda0af7dee0905605d9
e4aada4961542da952c8bb93080cc6f9

11

20

08e11fc41466fcda0af7dee0905605d9
e4aada4961542da952c8bb93080cc6f9

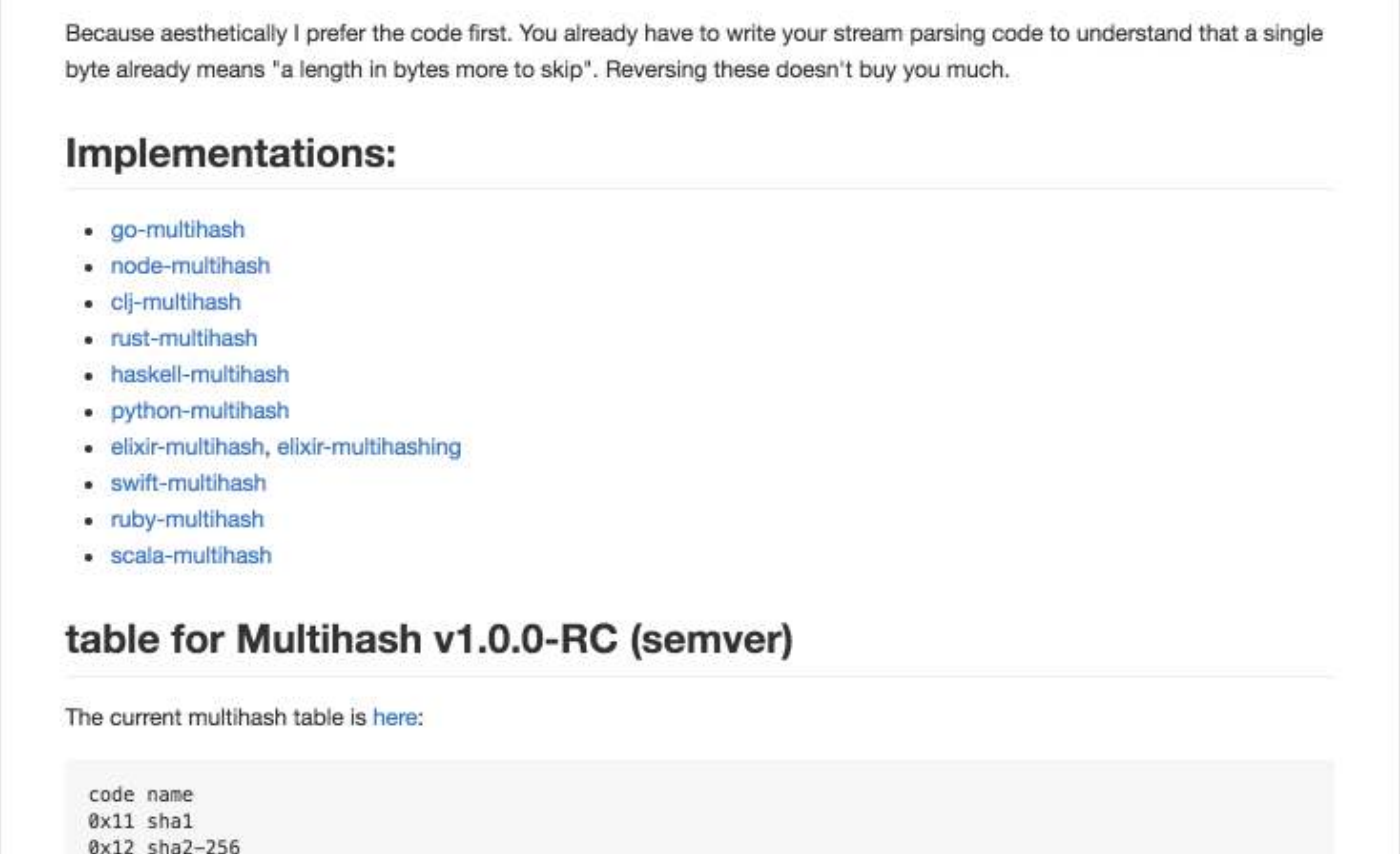
fn code

length

hash digest

multihash - cryptographic hashes

- self describing
- in the value itself (not out of band)
- as small as possible
- no assumptions
- no lock in
- interop of hash functions



Because aesthetically I prefer the code first. You already have to write your stream parsing code to understand that a single byte already means "a length in bytes more to skip". Reversing these doesn't buy you much.

Implementations:

- go-multihash
- node-multihash
- clj-multihash
- rust-multihash
- haskell-multihash
- python-multihash
- elixir-multihash, elixir-multihashing
- swift-multihash
- ruby-multihash
- scala-multihash

table for Multihash v1.0.0-RC (semver)

The current multihash table is here:

```
code name
0x11 sha1
0x12 sha2-256
```

multiformats - self describing values

protocol agility, interop, avoid lock in

multihash - cryptographic hashes

multiaddr - network addresses

multibase - base encodings

multicodec - serialization codecs

multistream - stream wire protocols

multikey - cryptographic keys and artifacts

multiaddr - network addresses

```
/ip6/::1/tcp/80/http

/ip4/1.2.3.4/udp/5001/sctp/sip

/ip4/1.2.3.4/udp/5002/utp/bittorrent

/ip4/1.2.3.4/udp/5003/quic/ipfs

/onion/3g2upl4pq6kufc4m/80/http
```

multiformats - self describing values

protocol agility, interop, avoid lock in

multihash - cryptographic hashes

multiaddr - network addresses

multibase - base encodings

multicodec - serialization codecs

multistream - stream wire protocols

multikey - cryptographic keys and artifacts

multiformats - self describing values

protocol agility, interop, avoid lock in

multihash
multiaddr
multibase
multicodec
multistream
multikey

- In Value (not OOB)
- Small, Binary (perf)
- Human Readable

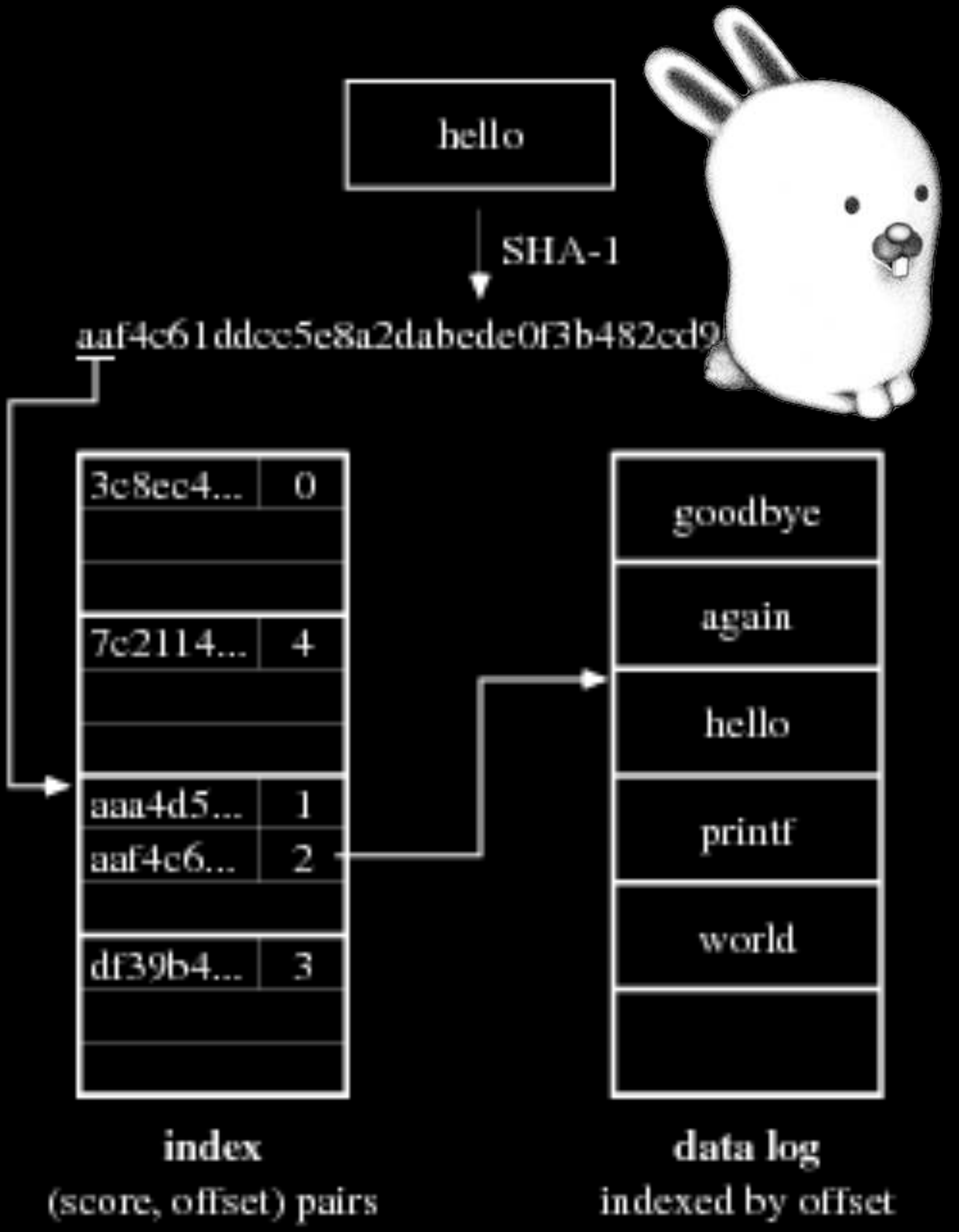
- Stable
- Starting Standard Now
- Impls in many langs

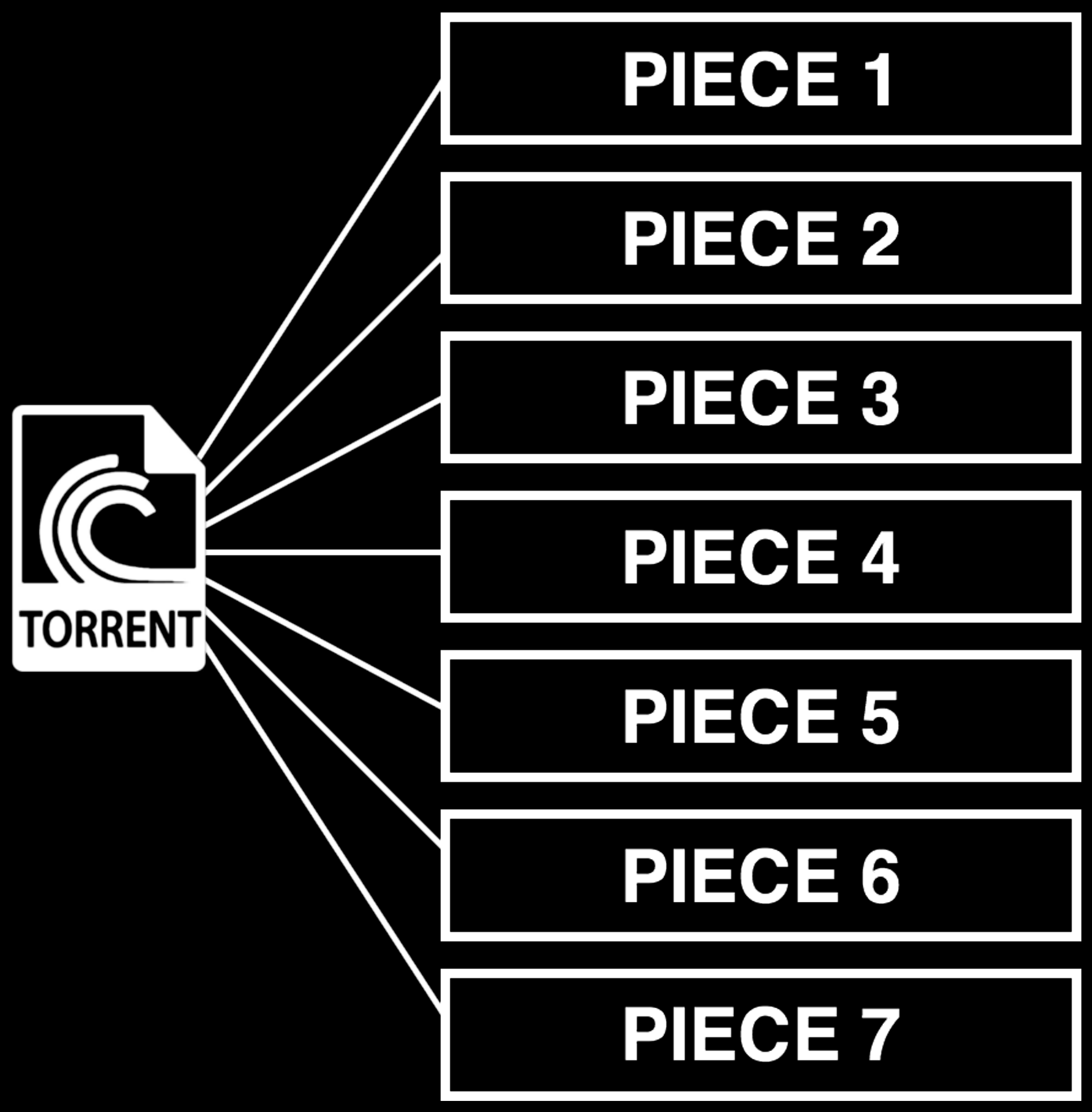
0. IPFS

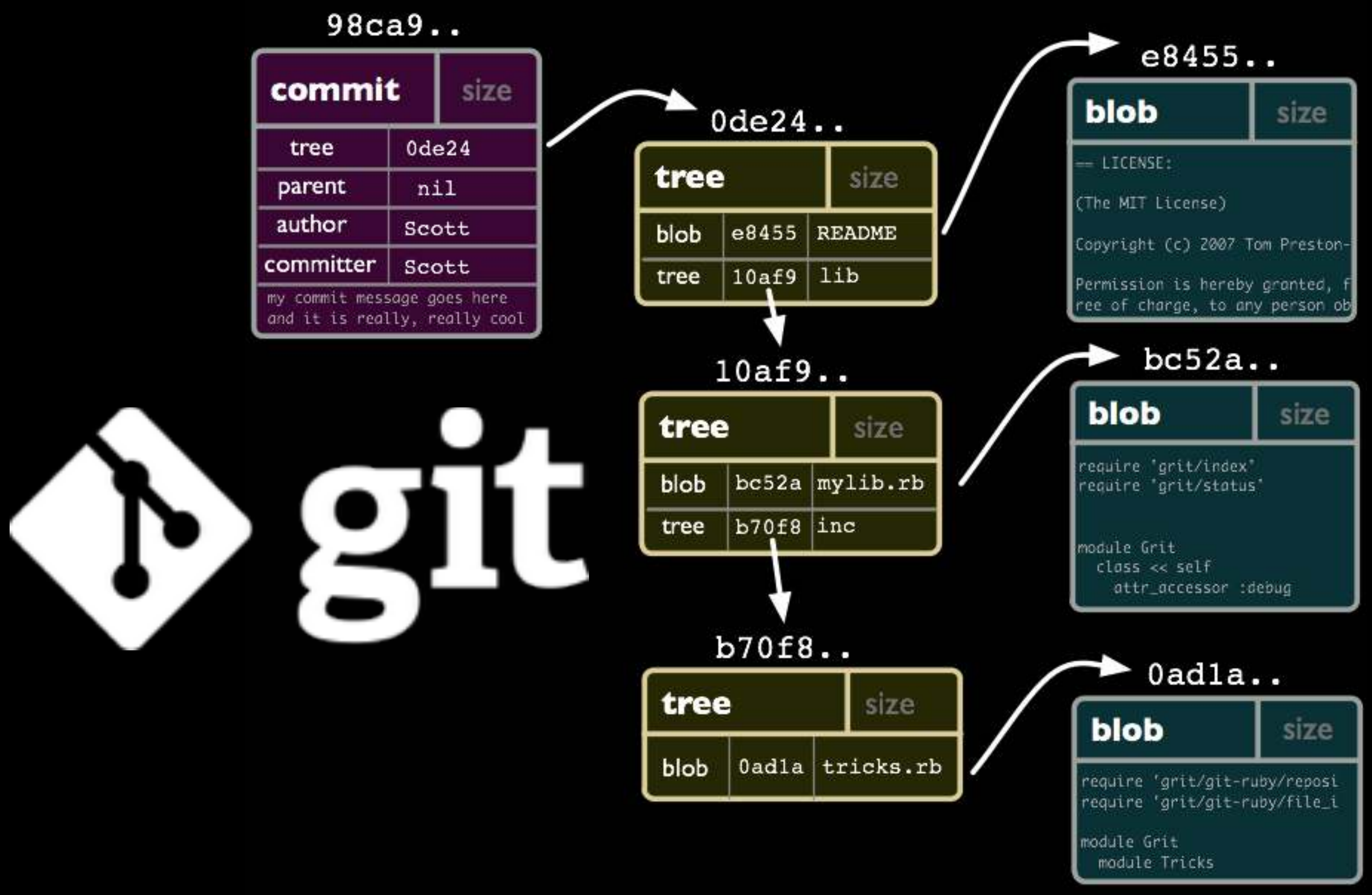
1. multi formats

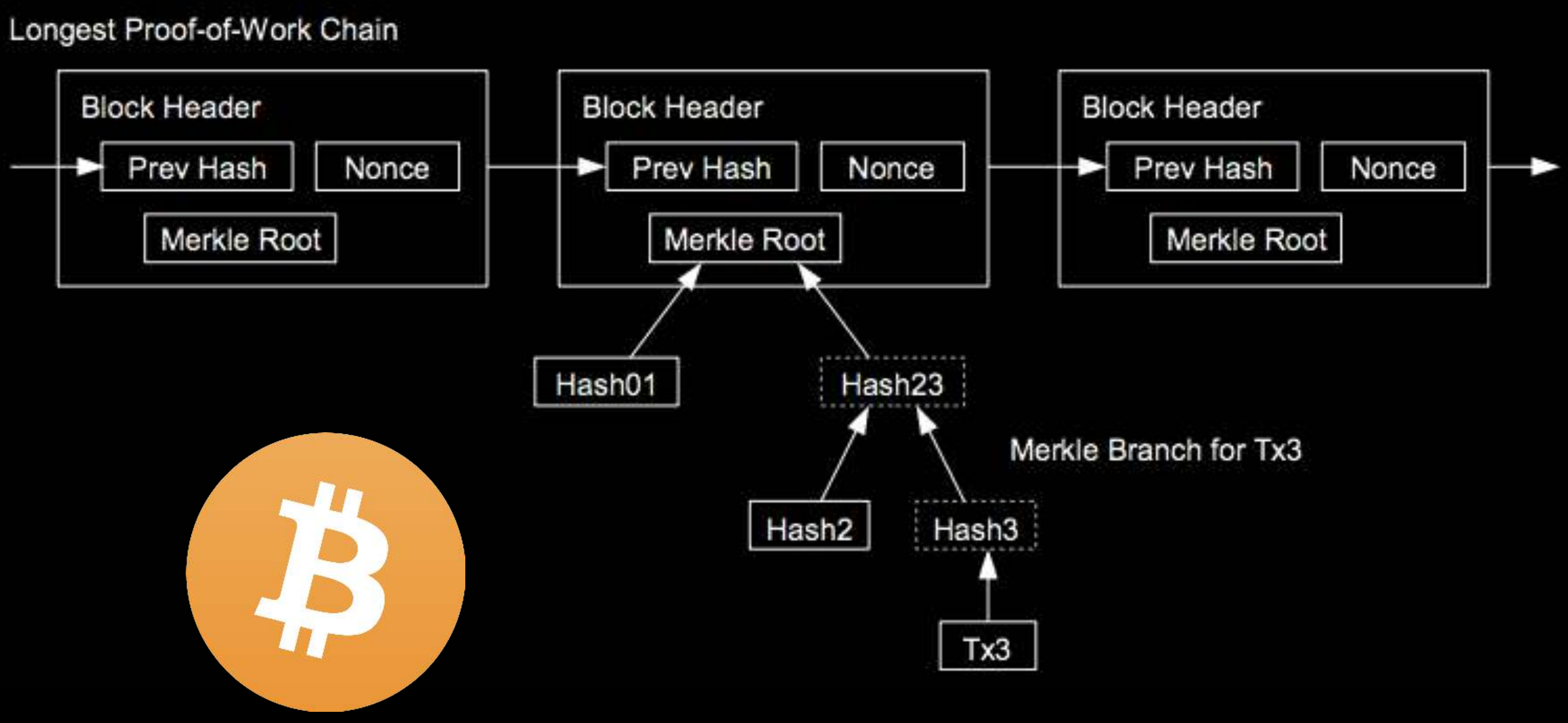
2. IPLD

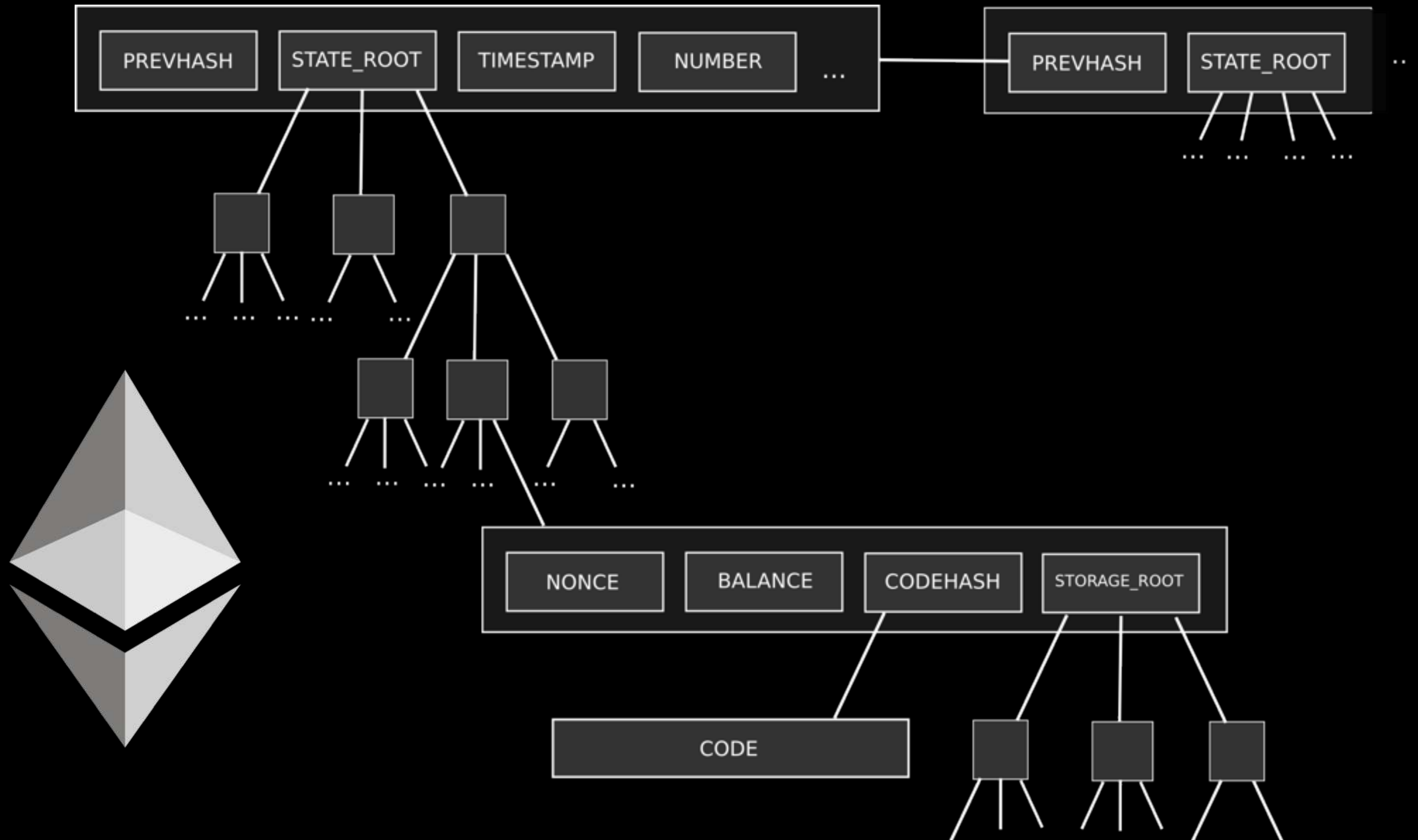
3. libp2p


hello
SHA-1
aaf4c61ddcc5e8a2dabede0f3b482cd9
3c8ec4... 0
7c2114... 4
aaa4d5... 1
aaf4c6... 2
df39b4... 3
goodbye
again
hello
printf
world
index
(score, offset) pairs
data log
indexed by offset


TORRENT
PIECE 1
PIECE 2
PIECE 3
PIECE 4
PIECE 5
PIECE 6
PIECE 7


git
98ca9..
commit size
tree 0de24
parent nil
author Scott
committer Scott
my commit message goes here
and it is really, really cool
0de24..
tree size
blob e8455 README
tree 10af9 lib
10af9..
tree size
blob bc52a mylib.rb
tree b70f8 inc
b70f8..
tree size
blob 0ad1a tricks.rb
e8455..
blob size
— LICENSE:
(The MIT License)
Copyright (c) 2007 Tom Preston-
Permission is hereby granted, f
ree of charge, to any person ob
bc52a..
blob size
require 'grit/index'
require 'grit/status'
module Grit
class << self
attr_accessor :debug
0ad1a..
blob size
require 'grit/git-ruby/reposi
require 'grit/git-ruby/file_i
module Grit
module Tricks


Longest Proof-of-Work Chain
Block Header
Prev Hash
Nonce
Merkle Root
Block Header
Prev Hash
Nonce
Merkle Root
Block Header
Prev Hash
Nonce
Merkle Root
Hash01
Hash23
Merkle Branch for Tx3
Hash2
Hash3
Tx3


PREVHASH
STATE_ROOT
TIMESTAMP
NUMBER
...
PREVHASH
STATE_ROOT
..
NONCE
BALANCE
CODEHASH
STORAGE_ROOT
CODE

distributed data structures

authenticated data structures

hash linked data structures

hello
SHA-1
aaf4c61ddcc5e8a2dabede0f3b482cd9
3c8ec4... 0
7c2114... 4
aaa4d5... 1
aaf4c6... 2
df39b4... 3
goodbye
again
hello
printf
world
index
(score, offset) pairs
data log
indexed by offset
TORRENT
PIECE 1
PIECE 2
PIECE 3
PIECE 4
PIECE 5
PIECE 6
PIECE 7
git
98ca9..
commit size
tree 0de24
parent nil
author Scott
committer Scott
my commit message goes here
and it is really, really cool
0de24..
tree size
blob e8455 README
tree 10af9 lib
10af9..
tree size
blob bc52a mylib.rb
tree b70f8 inc
b70f8..
tree size
blob 0ad1a tricks.rb
e8455..
blob size
== LICENSE:
(The MIT License)
Copyright (c) 2007 Tom Preston-
Permission is hereby granted, f
ree of charge, to any person ob
bc52a..
blob size
require 'grit/index'
require 'grit/status'
module Grit
class << self
attr_accessor :debug
0ad1a..
blob size
require 'grit/git-ruby/reposi
require 'grit/git-ruby/file_i
module Grit
module Tricks
Longest Proof-of-Work Chain
Block Header
Prev Hash
Nonce
Merkle Root
Block Header
Prev Hash
Nonce
Merkle Root
Block Header
Prev Hash
Nonce
Merkle Root
Hash01
Hash23
Merkle Branch for Tx3
Hash2
Hash3
Tx3
PREVHASH
STATE_ROOT
TIMESTAMP
NUMBER
PREVHASH
STATE_ROOT
NONCE
BALANCE
CODEHASH
STORAGE_ROOT
CODE

IPFS is like a forest of linked merkle-trees

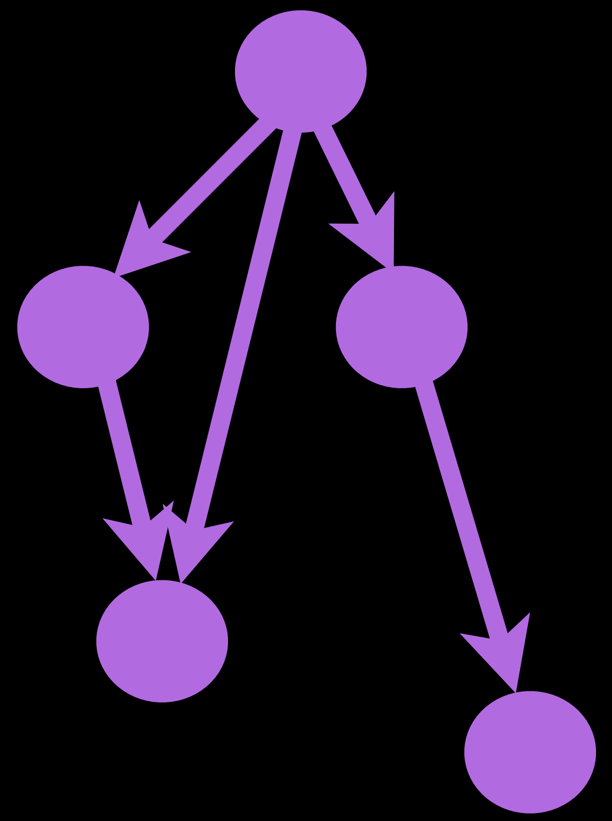
IPLD

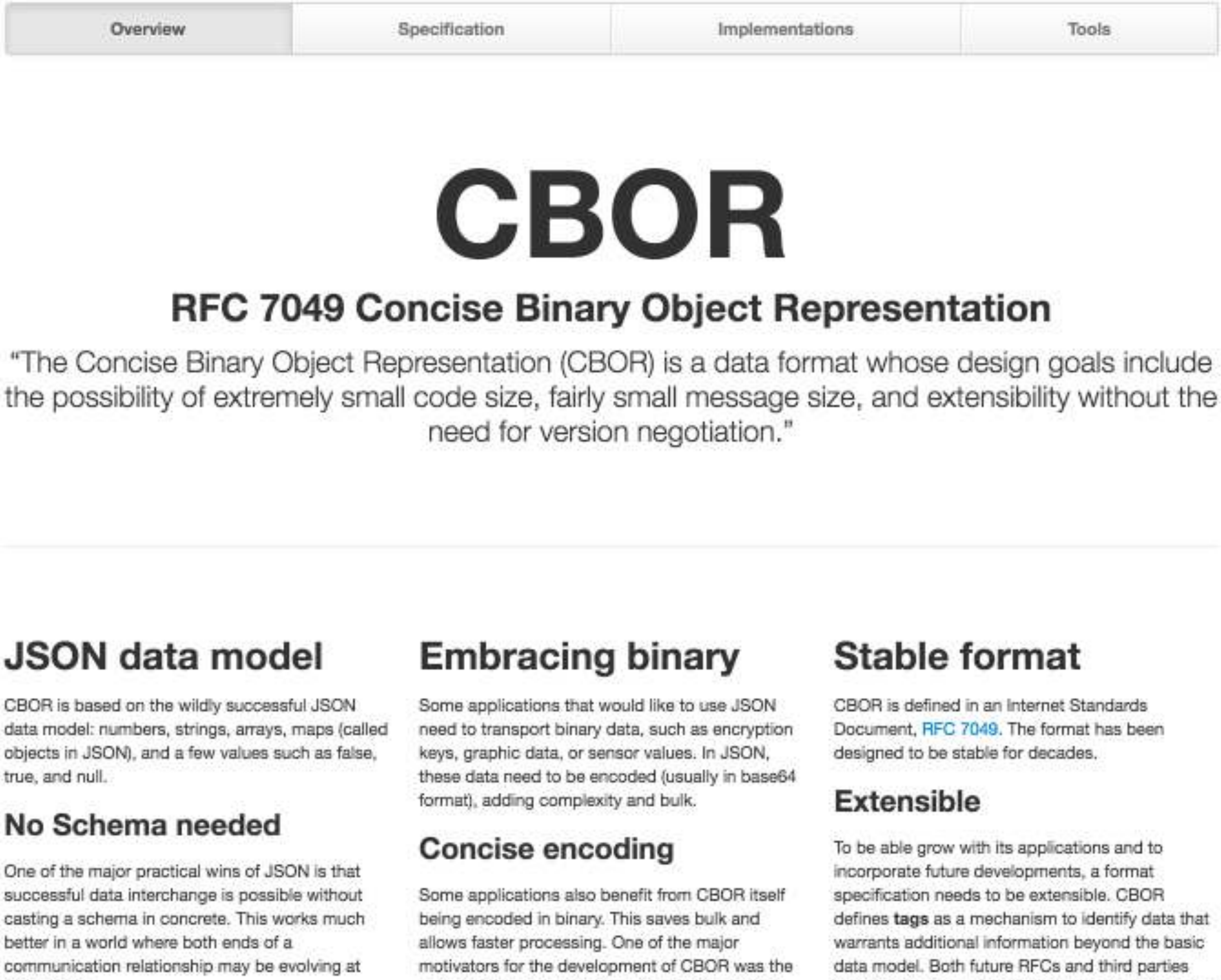
a common hash-chain format
for distributed data structures

🔗

IPLD

- **merkle-links** secure, immutable
- **merkle-paths** /ipfs/Qmabc…xyz/foo/bar.jpg
- **canonical** hashing safe
- **universal** nestable URIs
- **serialization** CBOR, JSON, YML, XML, PB
- **linked data** JSON-LD, RDF compatible





CBOR

RFC 7049 Concise Binary Object Representation

“The Concise Binary Object Representation (CBOR) is a data format whose design goals include the possibility of extremely small code size, fairly small message size, and extensibility without the need for version negotiation.”

JSON data model

CBOR is based on the wildly successful JSON data model: numbers, strings, arrays, maps (called objects in JSON), and a few values such as false, true, and null.

No Schema needed

One of the major practical wins of JSON is that successful data interchange is possible without casting a schema in concrete. This works much better in a world where both ends of a communication relationship may be evolving at

Embracing binary

Some applications that would like to use JSON need to transport binary data, such as encryption keys, graphic data, or sensor values. In JSON, these data need to be encoded (usually in base64 format), adding complexity and bulk.

Concise encoding

Some applications also benefit from CBOR itself being encoded in binary. This saves bulk and allows faster processing. One of the major motivators for the development of CBOR was the

Stable format

CBOR is defined in an Internet Standards Document, RFC 7049. The format has been designed to be stable for decades.

Extensible

To be able grow with its applications and to incorporate future developments, a format specification needs to be extensible. CBOR defines **tags** as a mechanism to identify data that warrants additional information beyond the basic data model. Both future RFCs and third parties

IPLD

a common hash-chain format
for distributed data structures

\o/   Ready for Standardization!   \o/

```
> var ipld = require('ipld')

> var obj1 = { "data": "Hello " }
> var ipld = require('ipld')

> var obj1 = { "data": "Hello " }

> var obj1Data = ipld.marshal(obj)
> obj1Data.toString('base64')
oWRkYXRhZkhlbGxvIA==
> var ipld = require('ipld')

> var obj1 = { "data": "Hello " }

> var obj1Data = ipld.marshal(obj)
> obj1Data.toString('base64')
oWRkYXRhZkhlbGxvIA==

> var obj1Hash = ipld.multihash(obj1Data)
> obj1Hash
QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg
```

```
> var ipld = require('ipld')

> var obj1 = { "data": "Hello " }
> var obj1Data = ipld.marshal(obj1)
> var obj1Hash = ipld.multihash(obj1Data)
> var ipld = require('ipld')

> var obj1 = { "data": "Hello " }
> var obj1Data = ipld.marshal(obj1)
> var obj1Hash = ipld.multihash(obj1Data)
```

```
> var ipld = require('ipld')

> var obj1 = { "data": "Hello " }
> var obj1Data = ipld.marshal(obj1)
> var obj1Hash = ipld.multihash(obj1Data)

> var obj2 = { "data": "World\n" }
> var obj2Data = ipld.marshal(obj2)
> var obj2Hash = ipld.multihash(obj2Data)

> obj2Hash
QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV
```

```
> var obj3 = {
 "files": [
   { "/": "QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg" },
   { "/": "QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV" },
 ]
}
> var obj3Data = ipld.marshal(obj3)
> var obj3Hash = ipld.multihash(obj3Data)

> obj3Hash
QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw
```

obj3

```
> var obj3 = {
 "files": [
   { "/": "QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg" },
   { "/": "QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV" },
 ]
}                                              \___ HASH LINKS!
> var obj3Data = ipld.marshal(obj3)
> var obj3Hash = ipld.multihash(obj3Data)

> obj3Hash
QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw
```

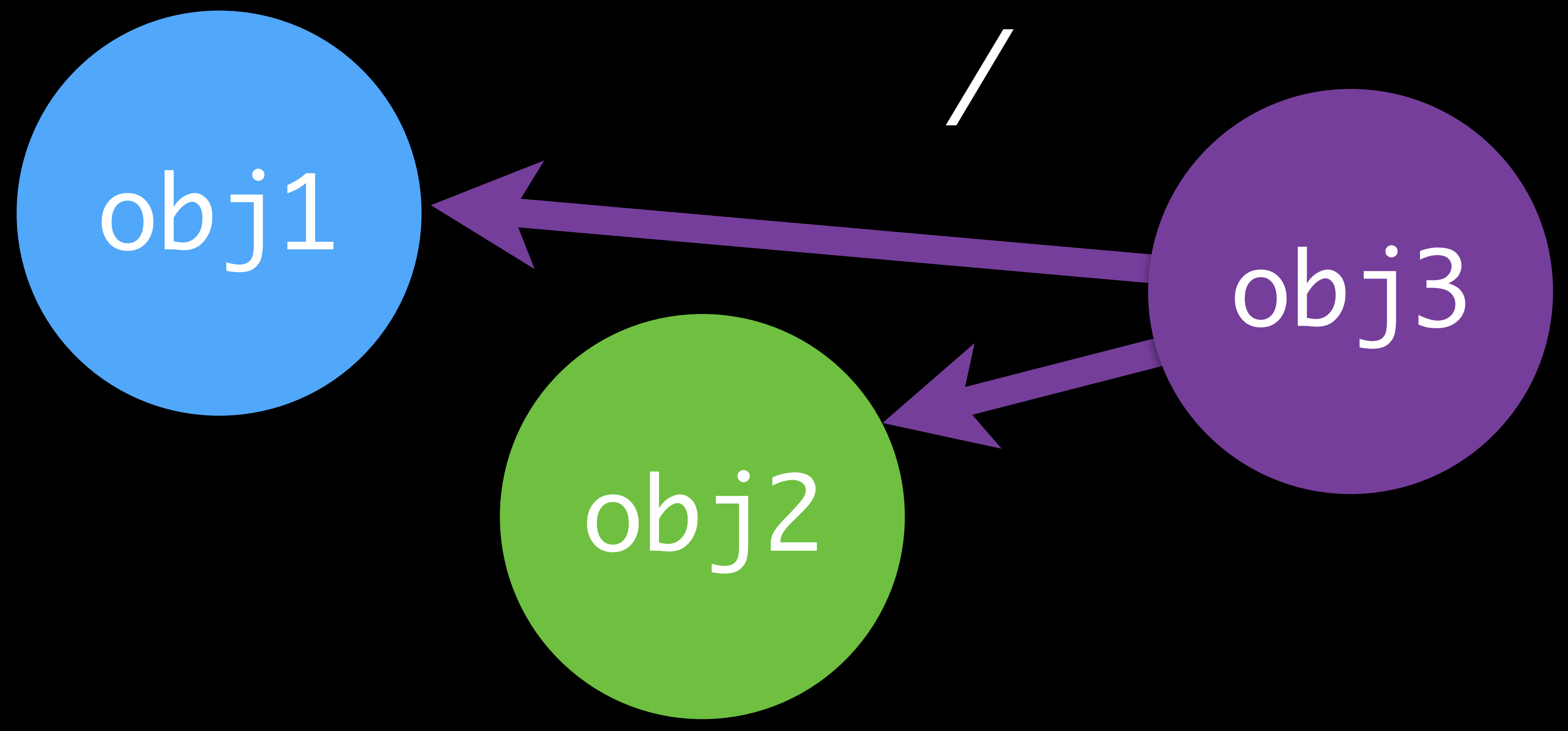
obj3

```
> var obj3 = {
 "files": [
   { "/": "QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg" },
   { "/": "QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV" },
 ]
}
> var obj3Data = ipld.marshal(obj3)
> var obj3Hash = ipld.multihash(obj3Data)
```

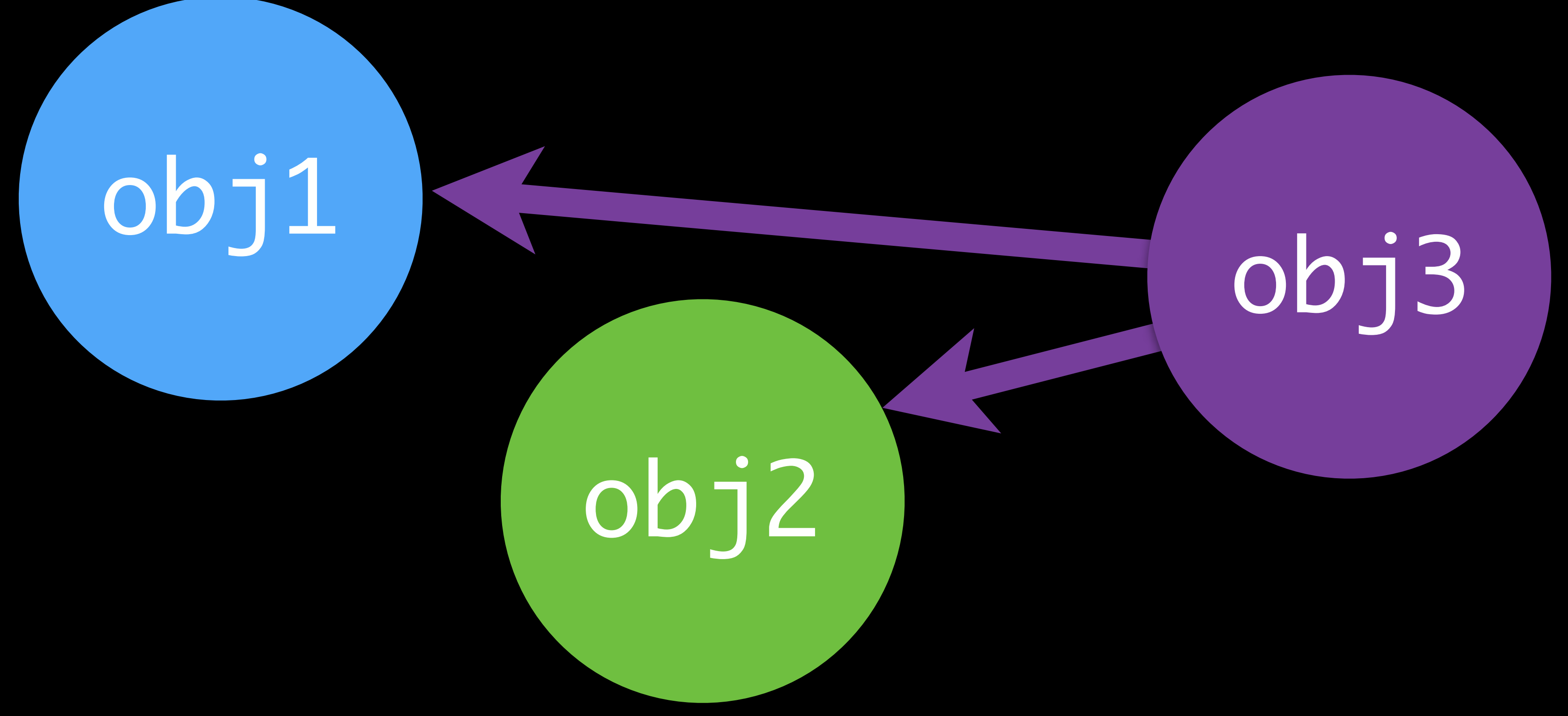
___ HASH LINKS!

HASH LINKS!

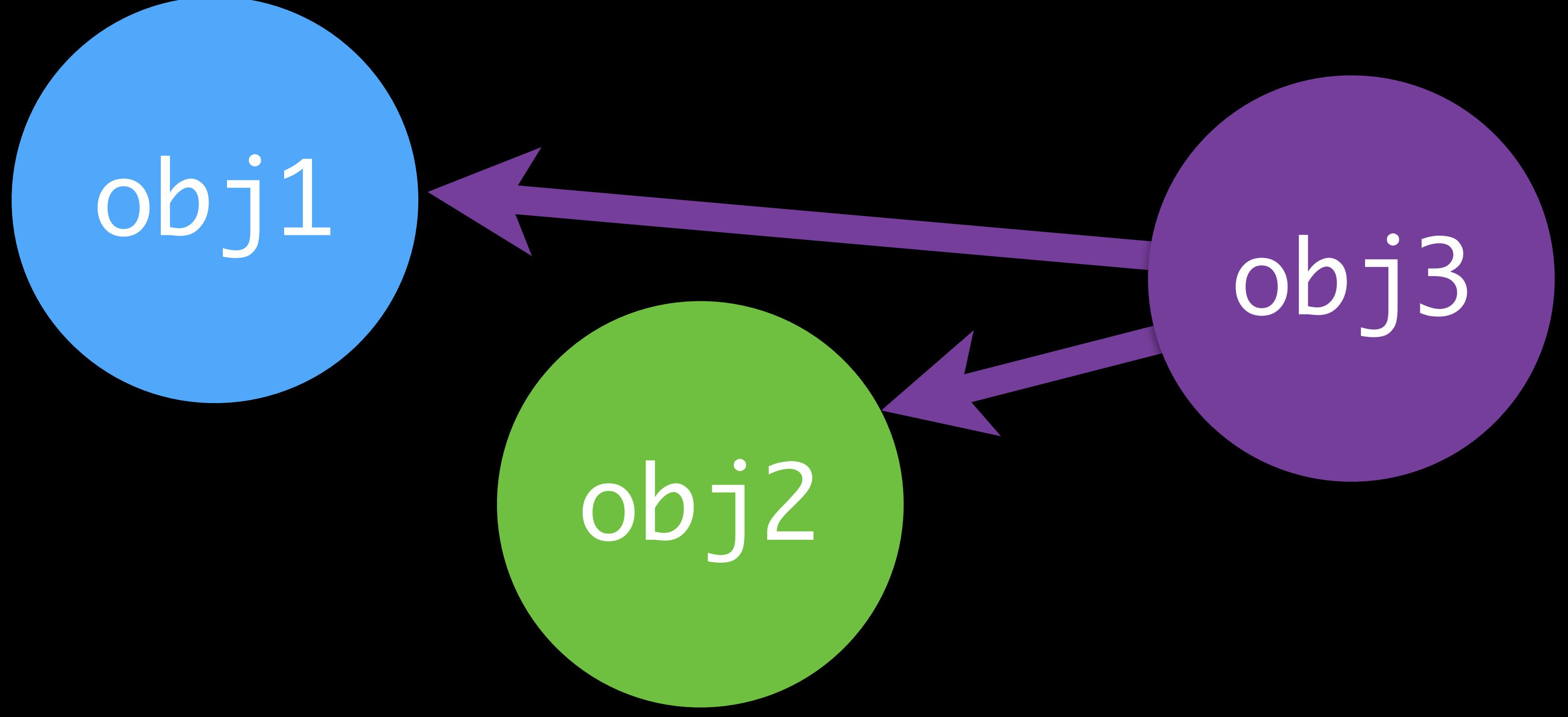
/



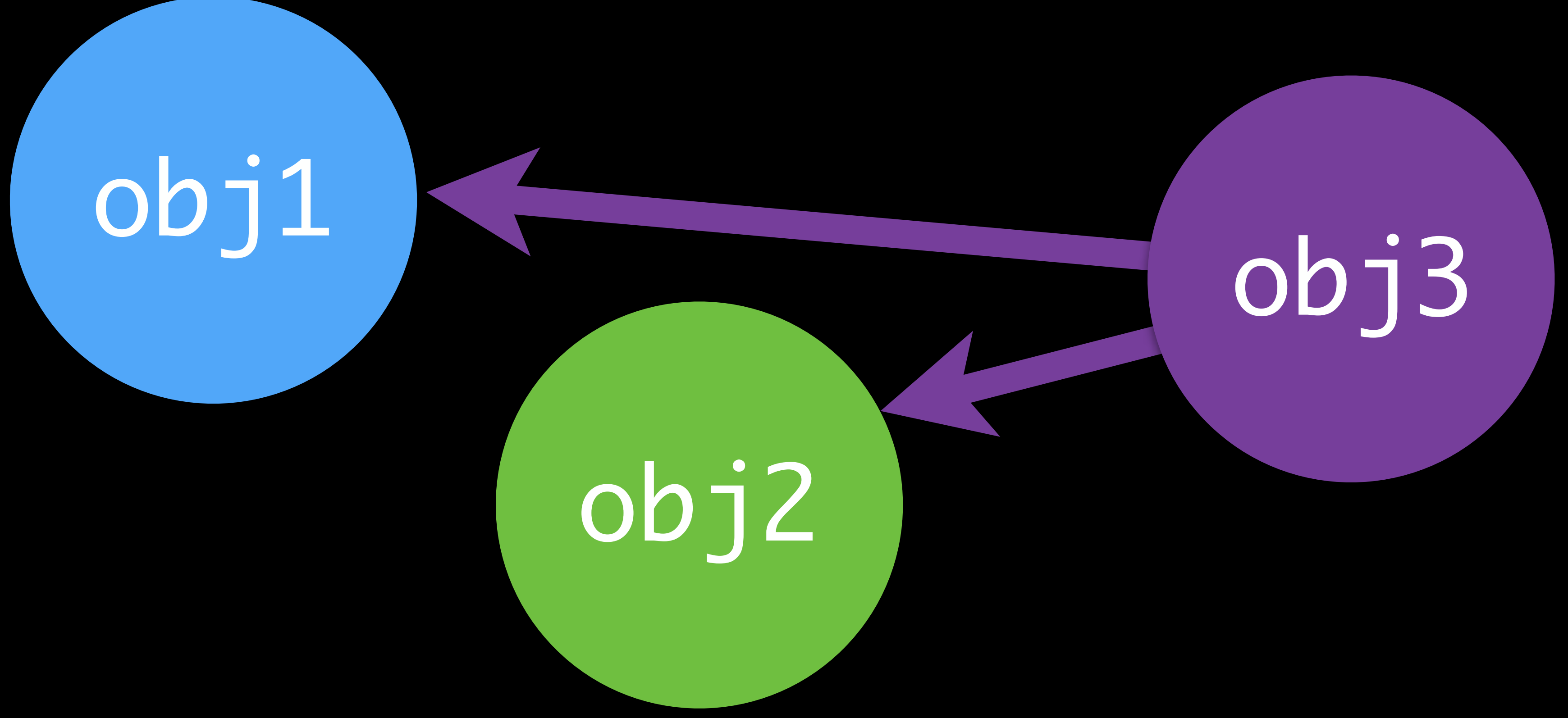
```
> var ipfs = require('ipfs')

> ipfs.add(obj1)
> ipfs.add(obj2)
> ipfs.add(obj3)
```

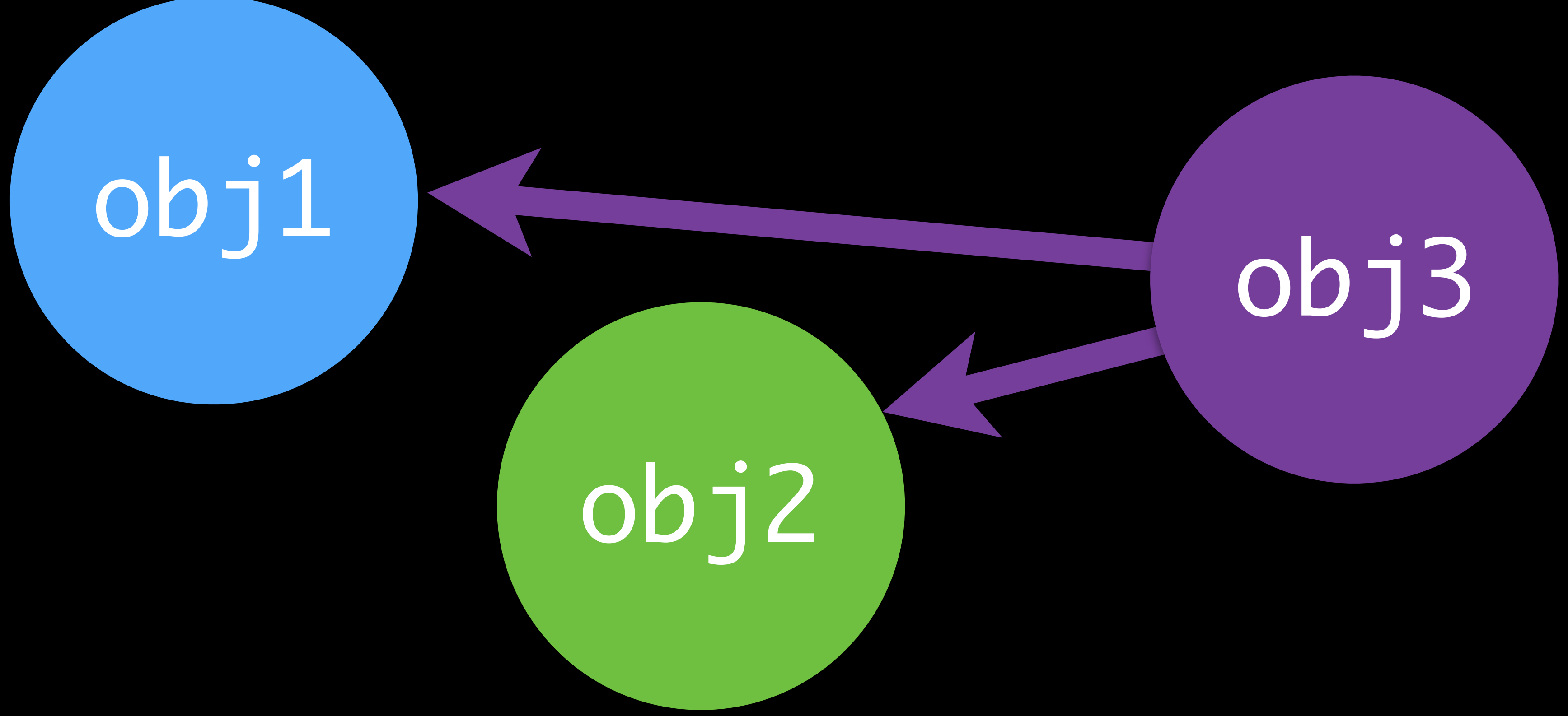


```
> var ipfs = require('ipfs')

> ipfs.add(obj1)
> ipfs.add(obj2)
> ipfs.add(obj3)

> ipfs.resolve("QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg")
{ "data": "Hello " }

> ipfs.resolve("QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV")
{ "data": "World\n" }
```

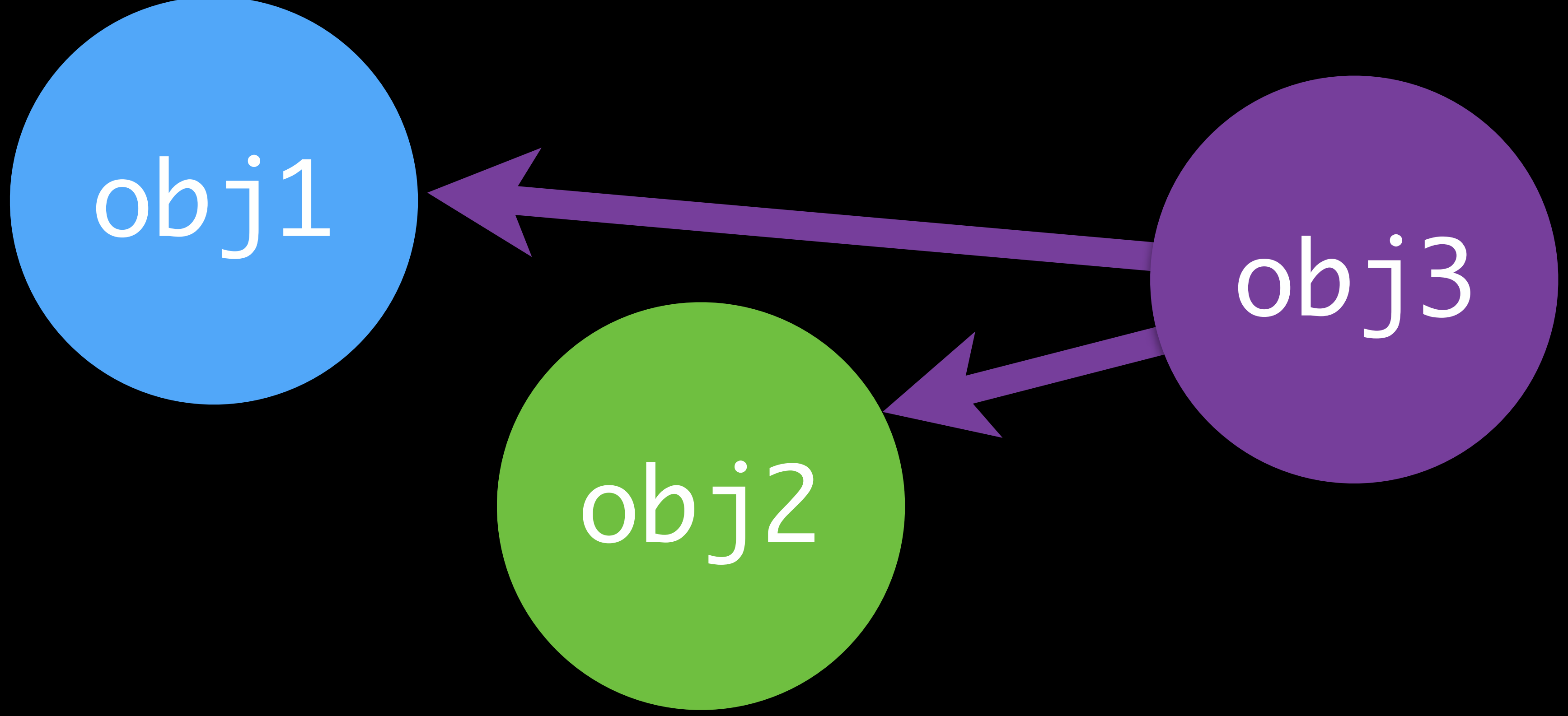


```
> var ipfs = require('ipfs')

> ipfs.add(obj1)
> ipfs.add(obj2)
> ipfs.add(obj3)

> ipfs.resolve("QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg")
{ "data": "Hello " }

> ipfs.resolve("QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV")
{ "data": "World\n" }

> ipfs.resolve("QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg/data")
"Hello "

> ipfs.resolve("QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV/data")
"World "
```

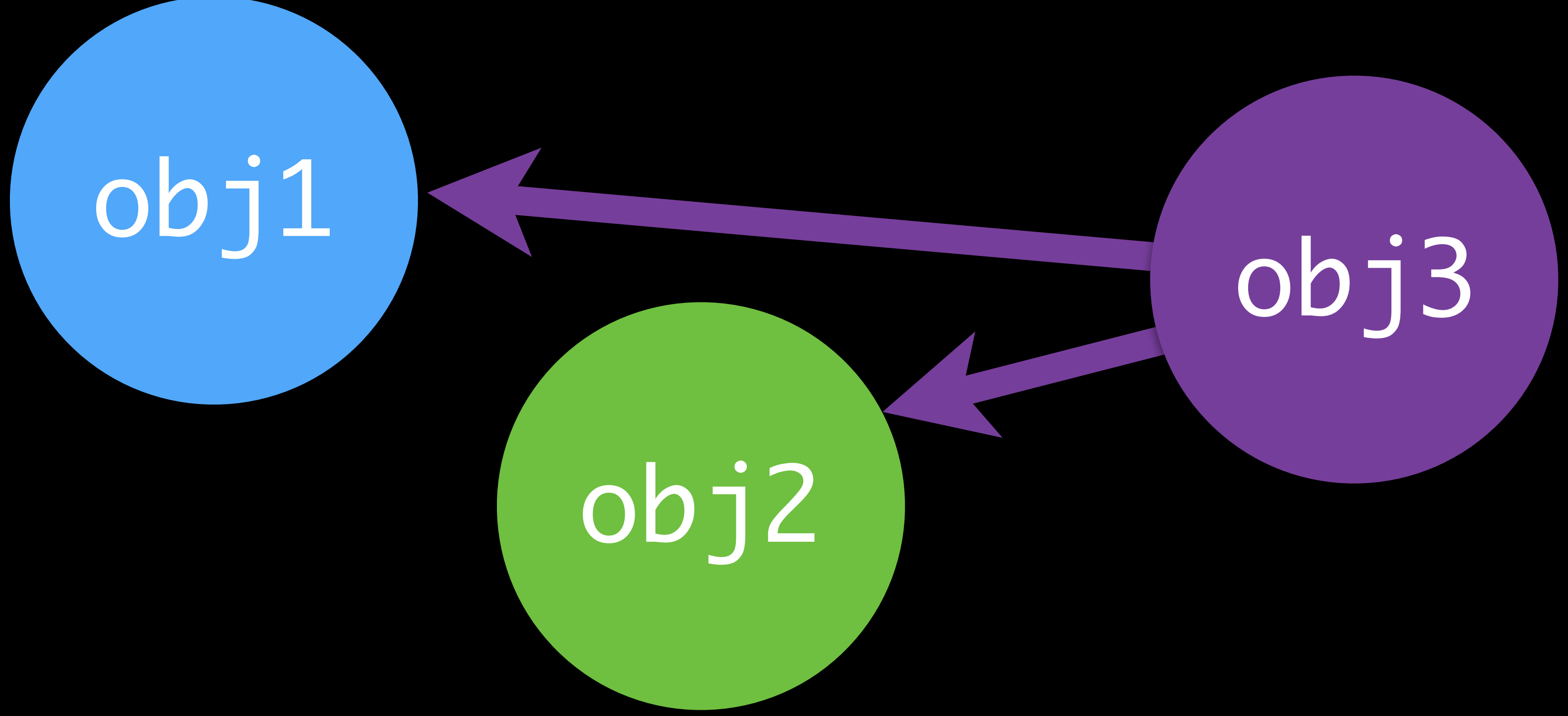




```
> var ipfs = require('ipfs')

> ipfs.add(obj1)
> ipfs.add(obj2)
> ipfs.add(obj3)

> ipfs.resolve("QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw")
{
  "files": [
    { "/": "QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg" },
    { "/": "QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV" },
  ]
}
```

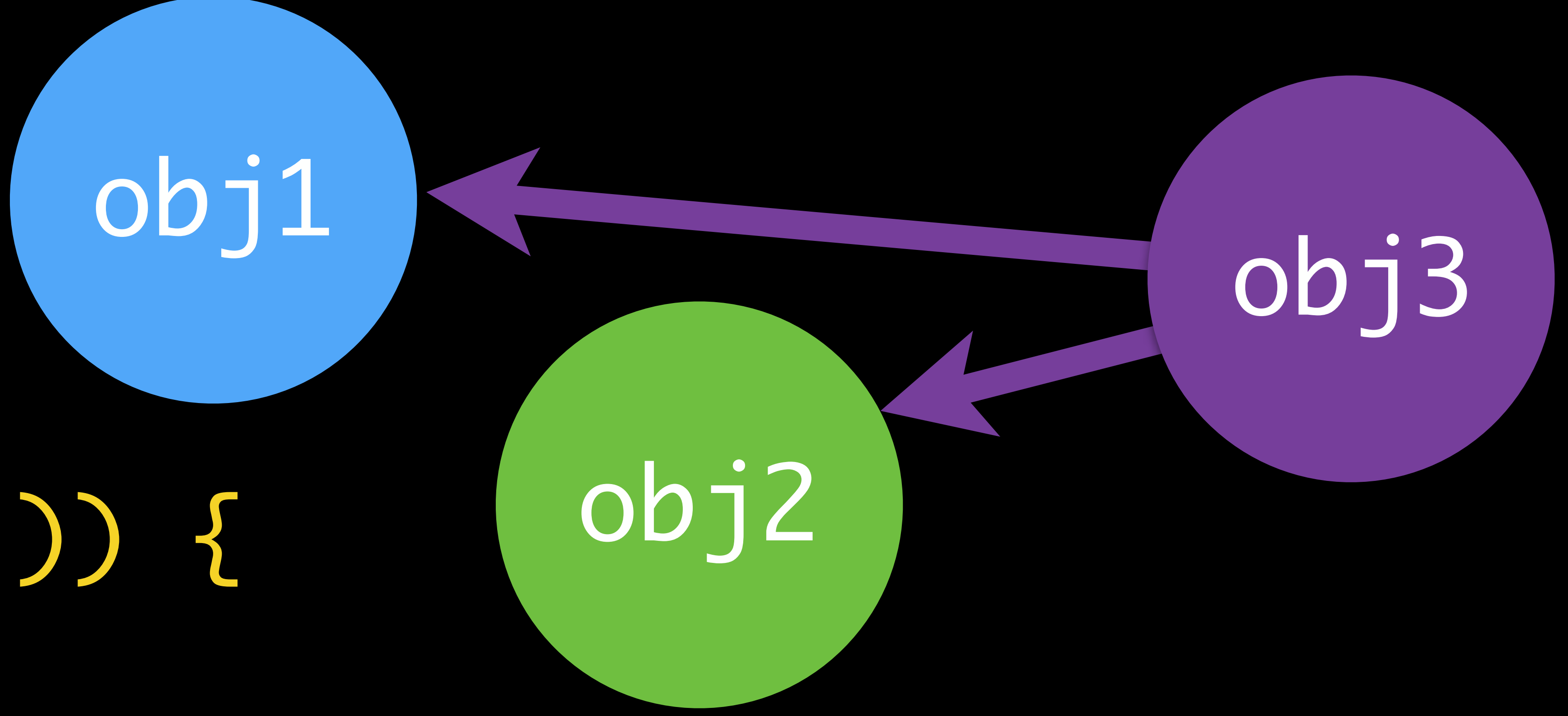


```
> var ipfs = require('ipfs')

> ipfs.add(obj1)
> ipfs.add(obj2)
> ipfs.add(obj3)

> ipfs.resolve("QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw")
{
  "files": [
    { "/": "QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg" },
    { "/": "QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV" },
  ]
}

> ipfs.resolve("QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw/files")
[
  { "/": "QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg" },
  { "/": "QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV" },
]
```



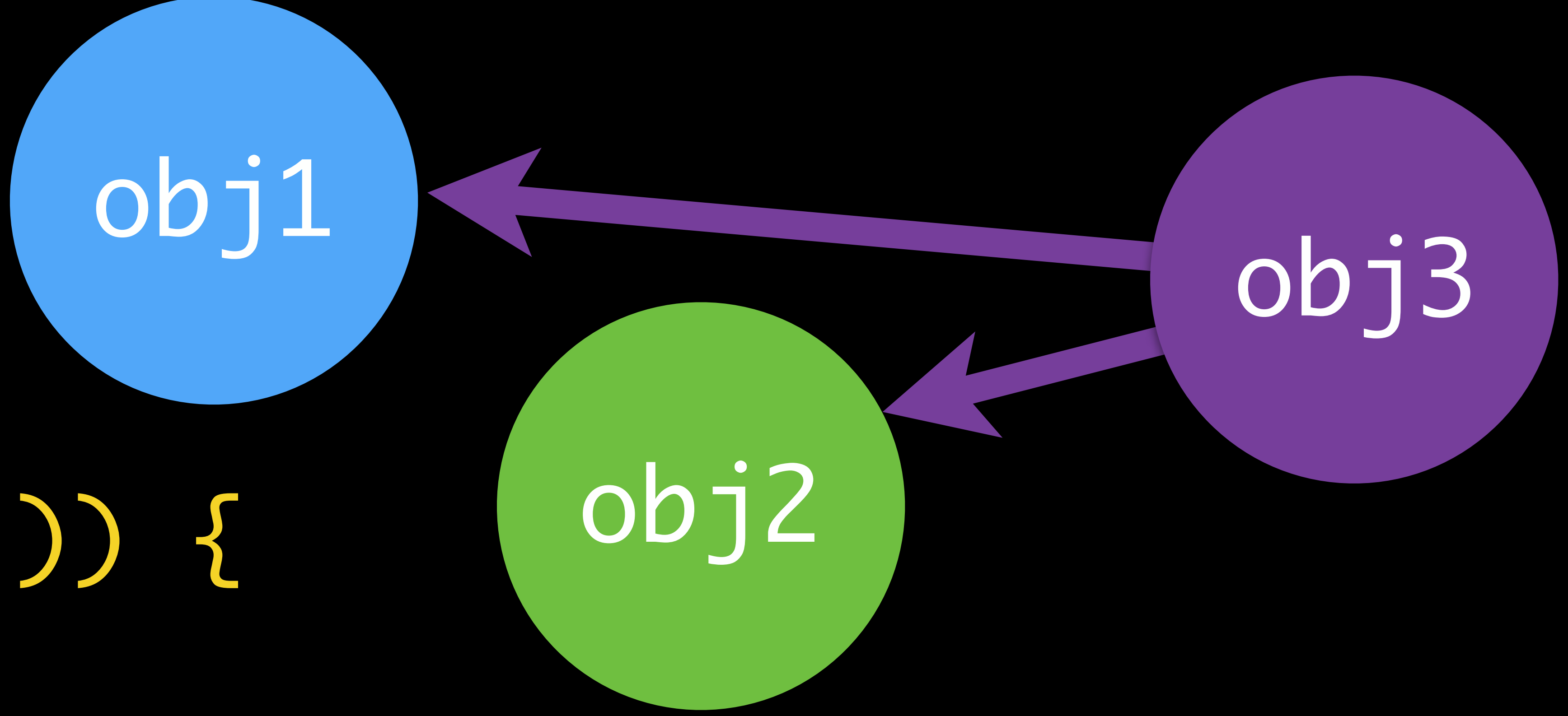
```
> var ipfs = require('ipfs')

> ipfs.add(obj1)
> ipfs.add(obj2)
> ipfs.add(obj3)

> ipfs.resolve("QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw/files/0")
{ "data": "Hello " }

> ipfs.resolve("QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw/files/1")
{ "data": "World\n" }

> ipfs.resolve("QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw/files/0/data")
"Hello "

> ipfs.resolve("QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw/files/1/data")
"World "
```

```
> function catFile(link) {
  var obj = ipfs.resolve(link)
  var out = obj.data || ""
  for (var file of (obj.files || [])) {
    out += catFile(file)
  }
  return out
}
```



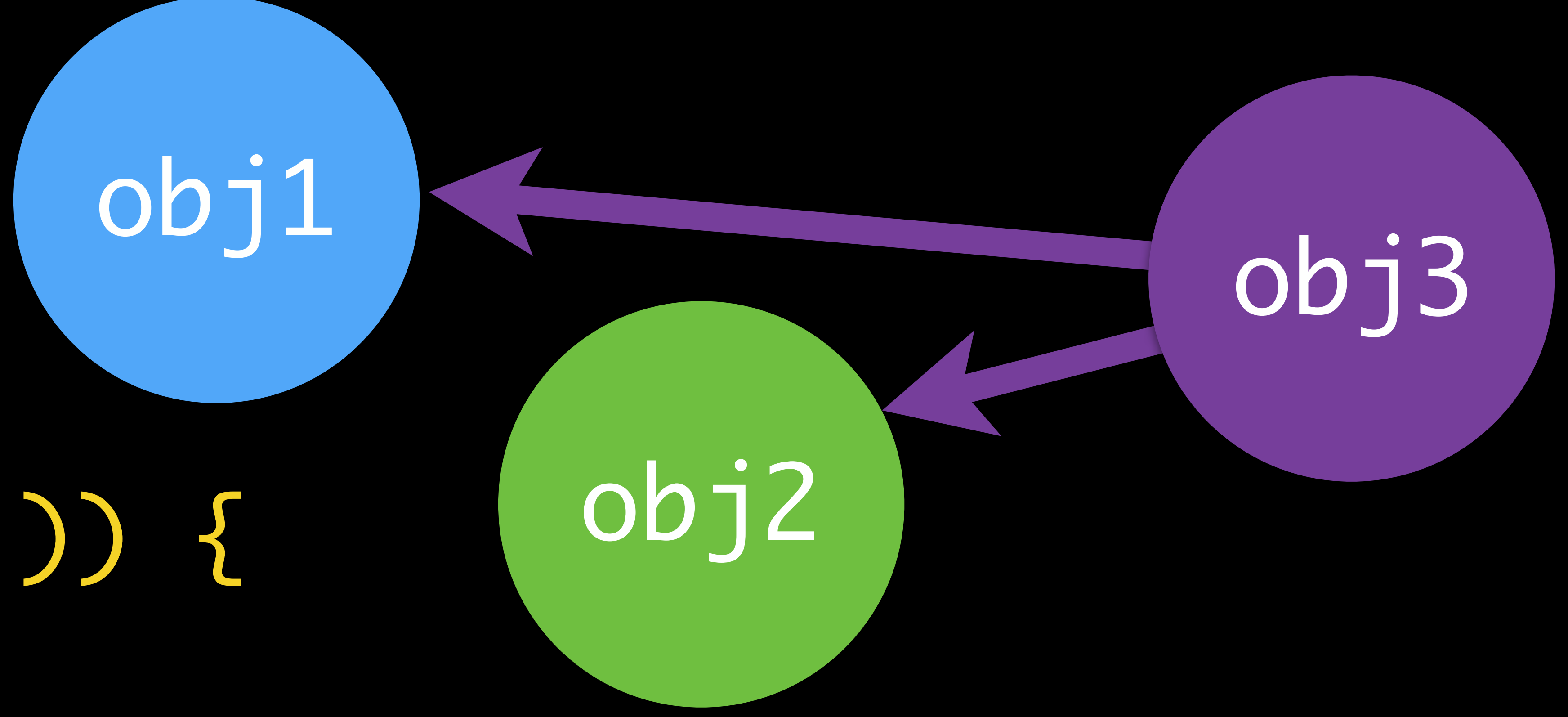
```
{
  "data": "<content>",
  "files": [
    <subfile-link>,
    <subfile-link>,
    <subfile-link>,
    ...
  ]
}
```

```
> function catFile(link) {
  var obj = ipfs.resolve(link)
  var out = obj.data || ""
  for (var file of (obj.files || [])) {
    out += catFile(file)
  }
  return out
}
```



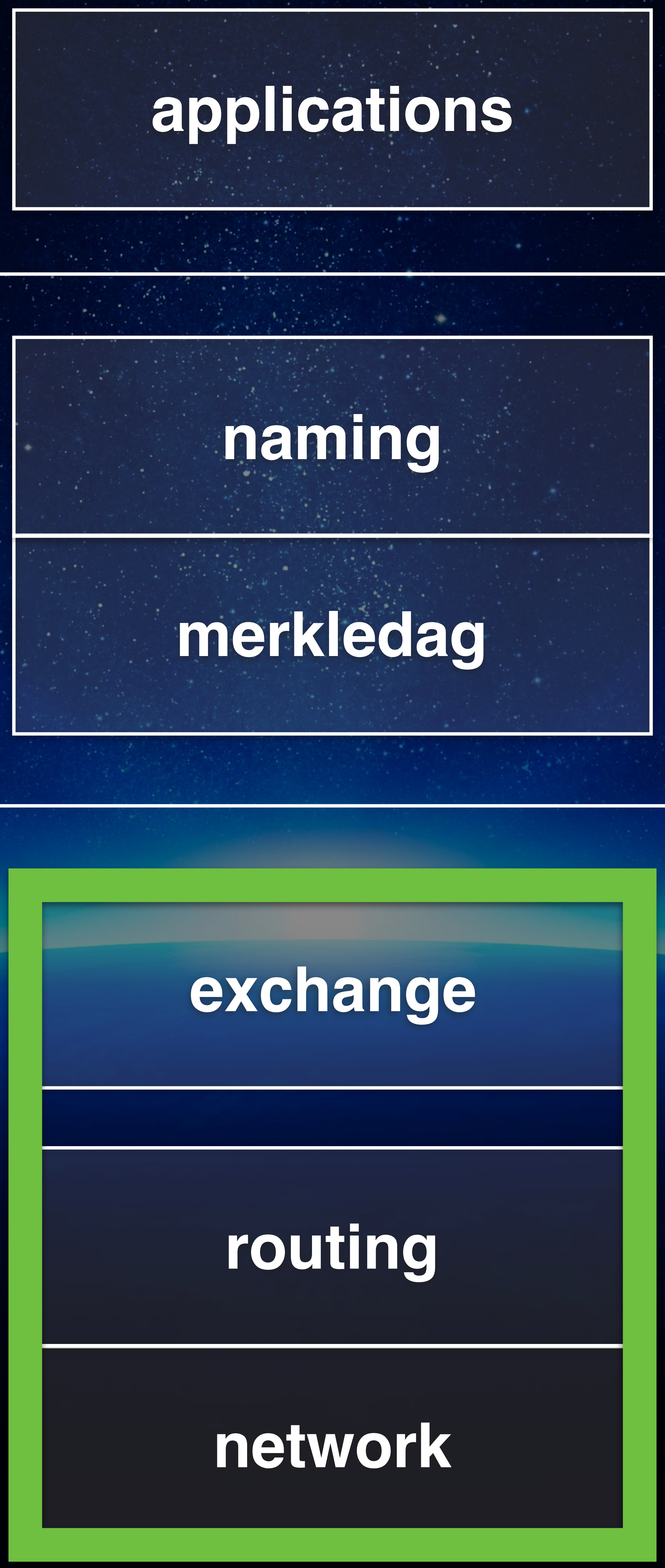
```
> catFile("QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg")
"Hello "

> catFile("QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV")
"World\n"
```

```
> function catFile(link) {
  var obj = ipfs.resolve(link)
  var out = obj.data || ""
  for (var file of (obj.files || [])) {
    out += catFile(file)
  }
  return out
}
```



```
> catFile("QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg")
"Hello "

> catFile("QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV")
"World\n"

> catFile("QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw")
"Hello World\n"
```

0. IPFS
1. multi formats
2. IPLD
3. libp2p

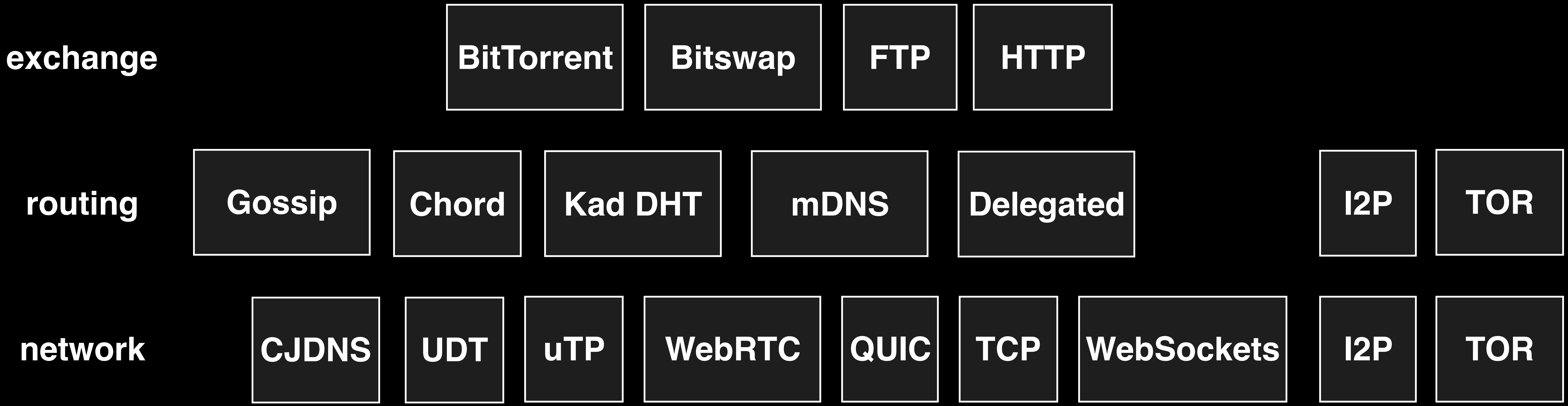


The IPFS Stack	applications	Using the Data
IPNS IPLD	naming merkledag	Defining the Data
libp2p	exchange routing network	Moving the Data

libp2p a collection of peer-to-peer protocols

for finding peers, and connecting to them

for finding content, and transferring it

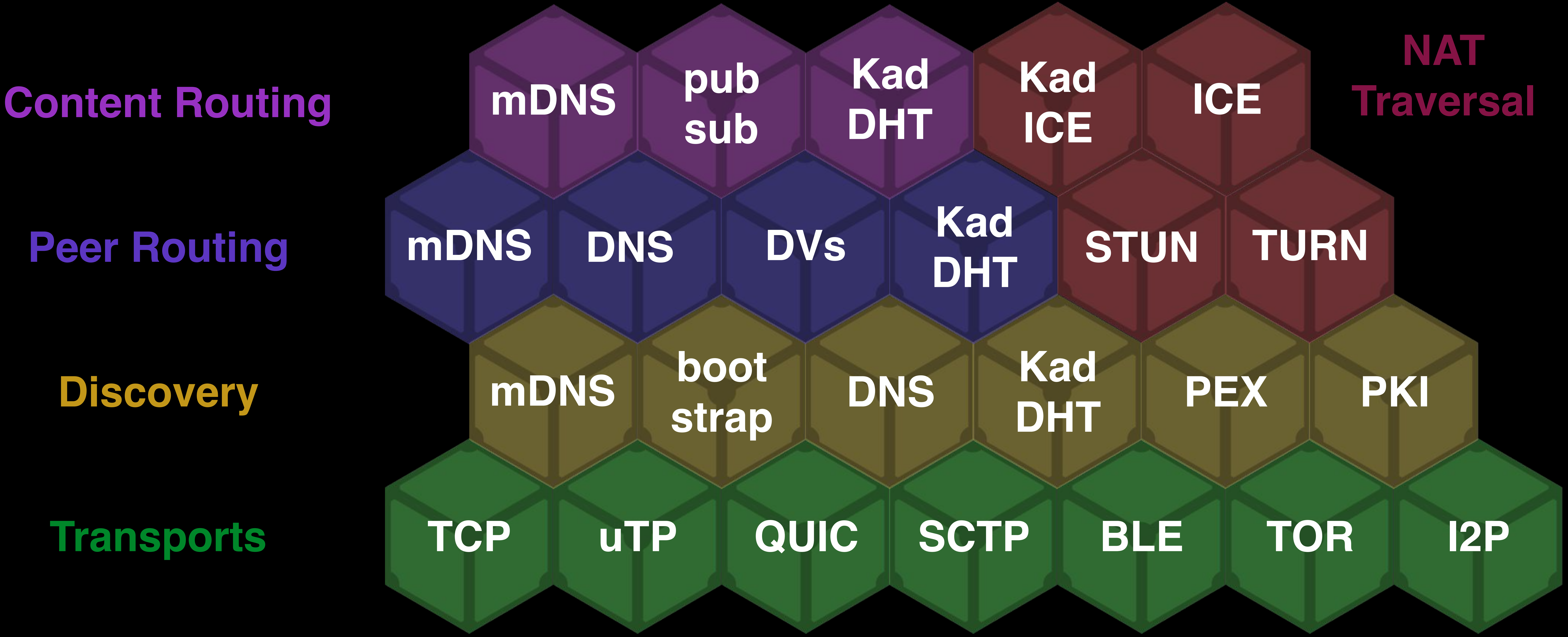


exchange: BitTorrent, Bitswap, FTP, HTTP

routing: Gossip, Chord, Kad DHT, mDNS, Delegated, I2P, TOR

network: CJDNS, UDT, uTP, WebRTC, QUIC, TCP, WebSockets, I2P, TOR

libp2p a collection of peer-to-peer protocols



Content Routing: mDNS, pub sub, Kad DHT

NAT Traversal: Kad ICE, ICE, STUN, TURN

Peer Routing: mDNS, DNS, DVs, Kad DHT

Discovery: mDNS, boot strap, DNS, Kad DHT, PEX, PKI

Transports: TCP, uTP, QUIC, SCTP, BLE, TOR, I2P libp2p

Content Routing

Peer Routing

Discovery

Transports

NAT Traversal

libp2p

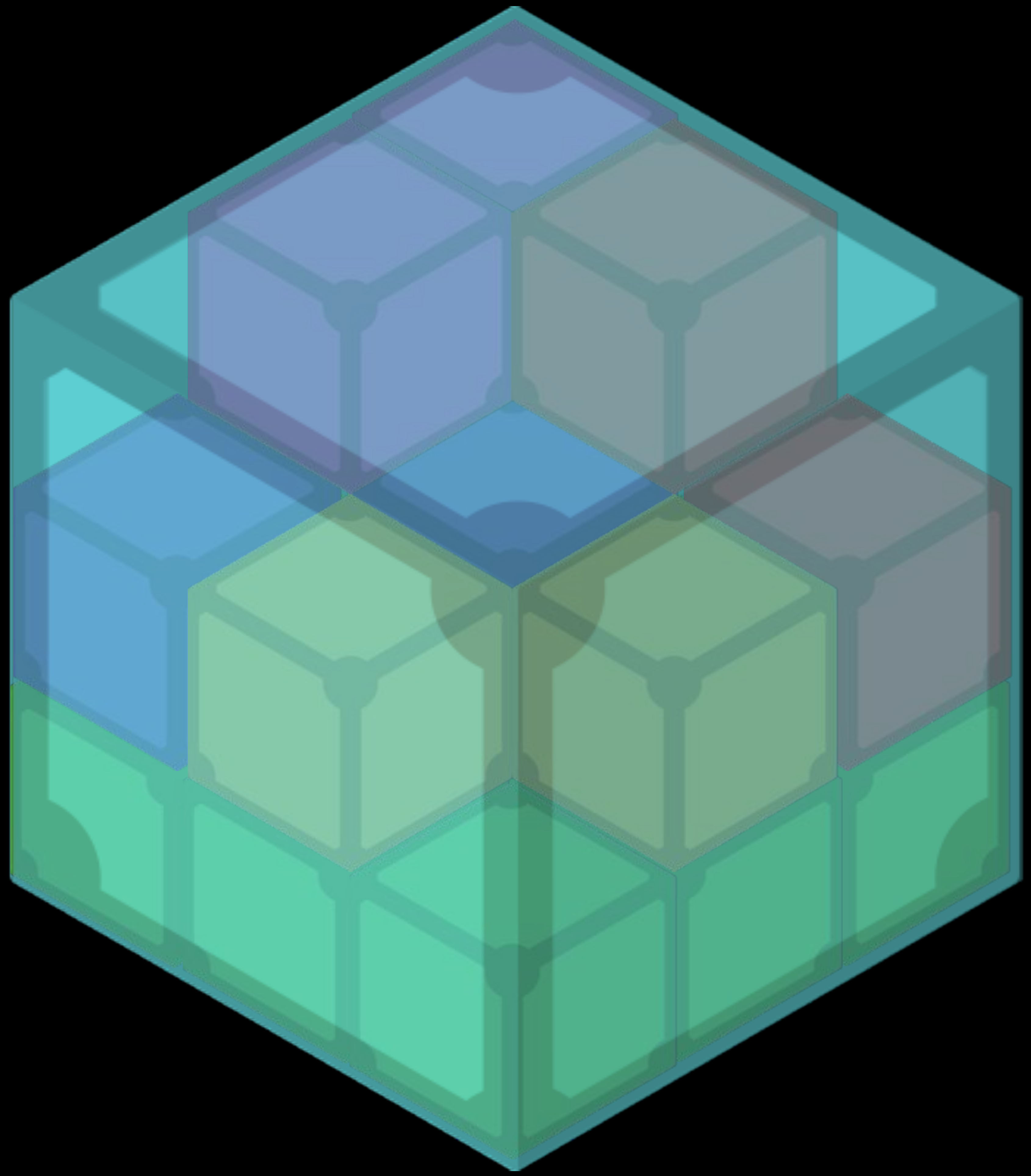
Content Routing

Peer Routing

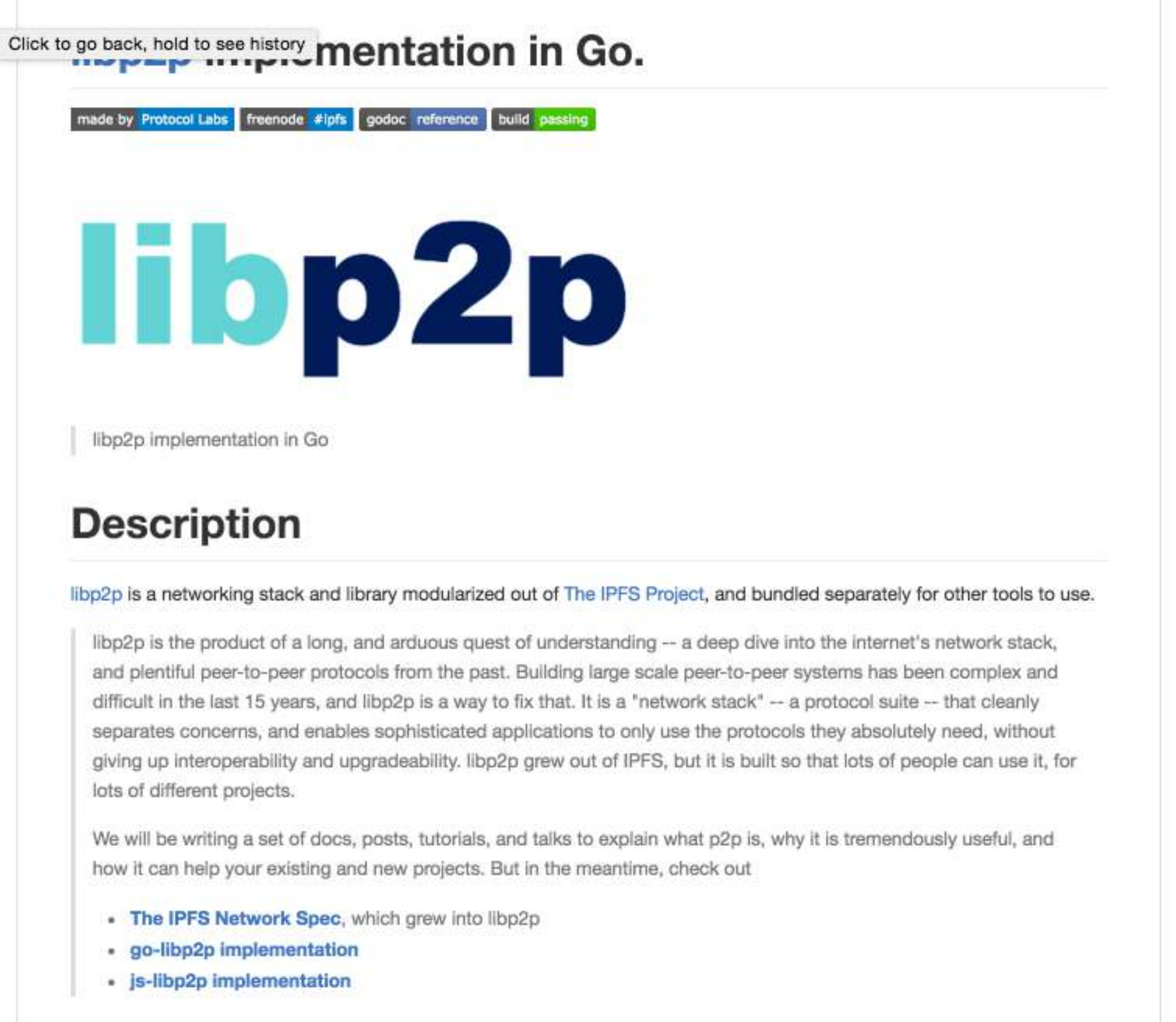
Discovery

Transports

NAT Traversal



libp2p



libp2p implementation in Go.

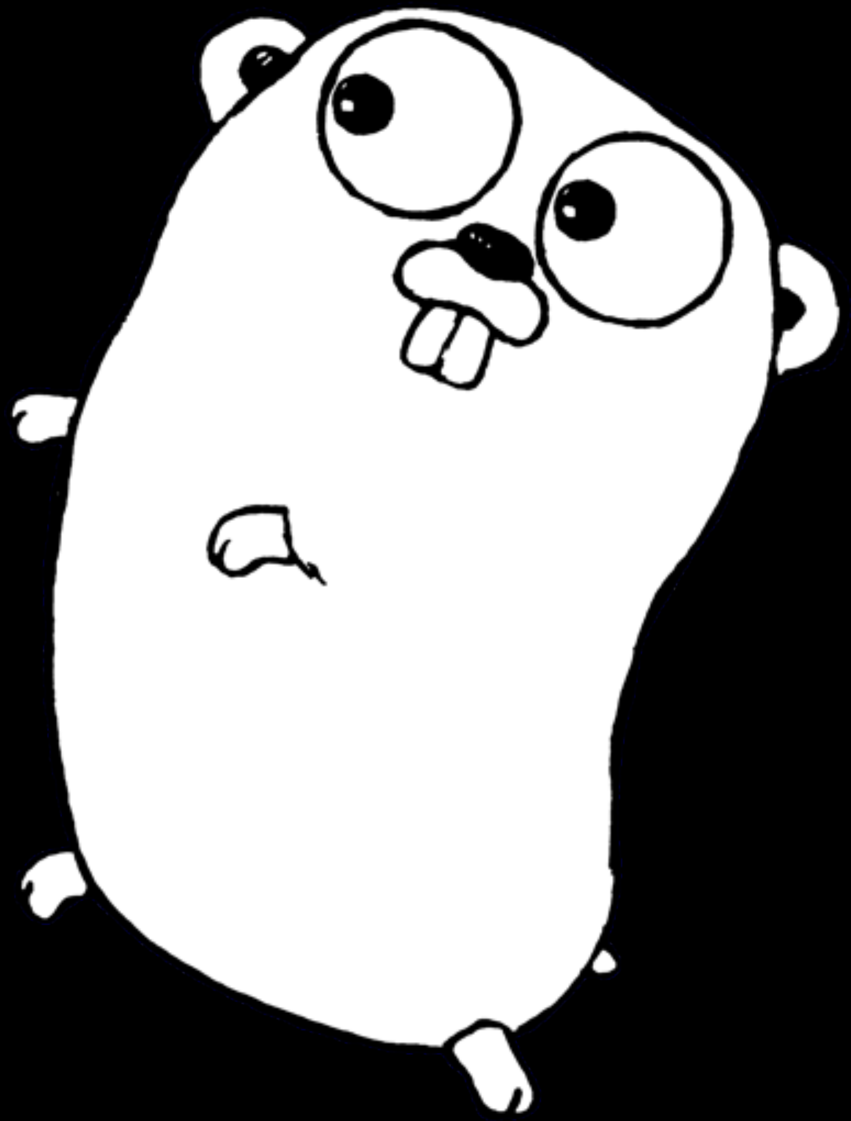
libp2p

> libp2p implementation in Go

Description

libp2p is a networking stack and library modularized out of The IPFS Project, and bundled separately for other tools to use.

> libp2p is the product of a long, and arduous quest of understanding -- a deep dive into the internet's network stack, and plentiful peer-to-peer protocols from the past. Building large scale peer-to-peer systems has been complex and difficult in the last 15 years, and libp2p is a way to fix that. It is a "network stack" -- a protocol suite -- that cleanly separates concerns, and enables sophisticated applications to only use the protocols they absolutely need, without giving up interoperability and upgradeability. libp2p grew out of IPFS, but it is built so that lots of people can use it, for lots of different projects.
>
> We will be writing a set of docs, posts, tutorials, and talks to explain what p2p is, why it is tremendously useful, and how it can help your existing and new projects. But in the meantime, check out
>
> - **The IPFS Network Spec**, which grew into libp2p
> - **go-libp2p implementation**
> - **js-libp2p implementation**

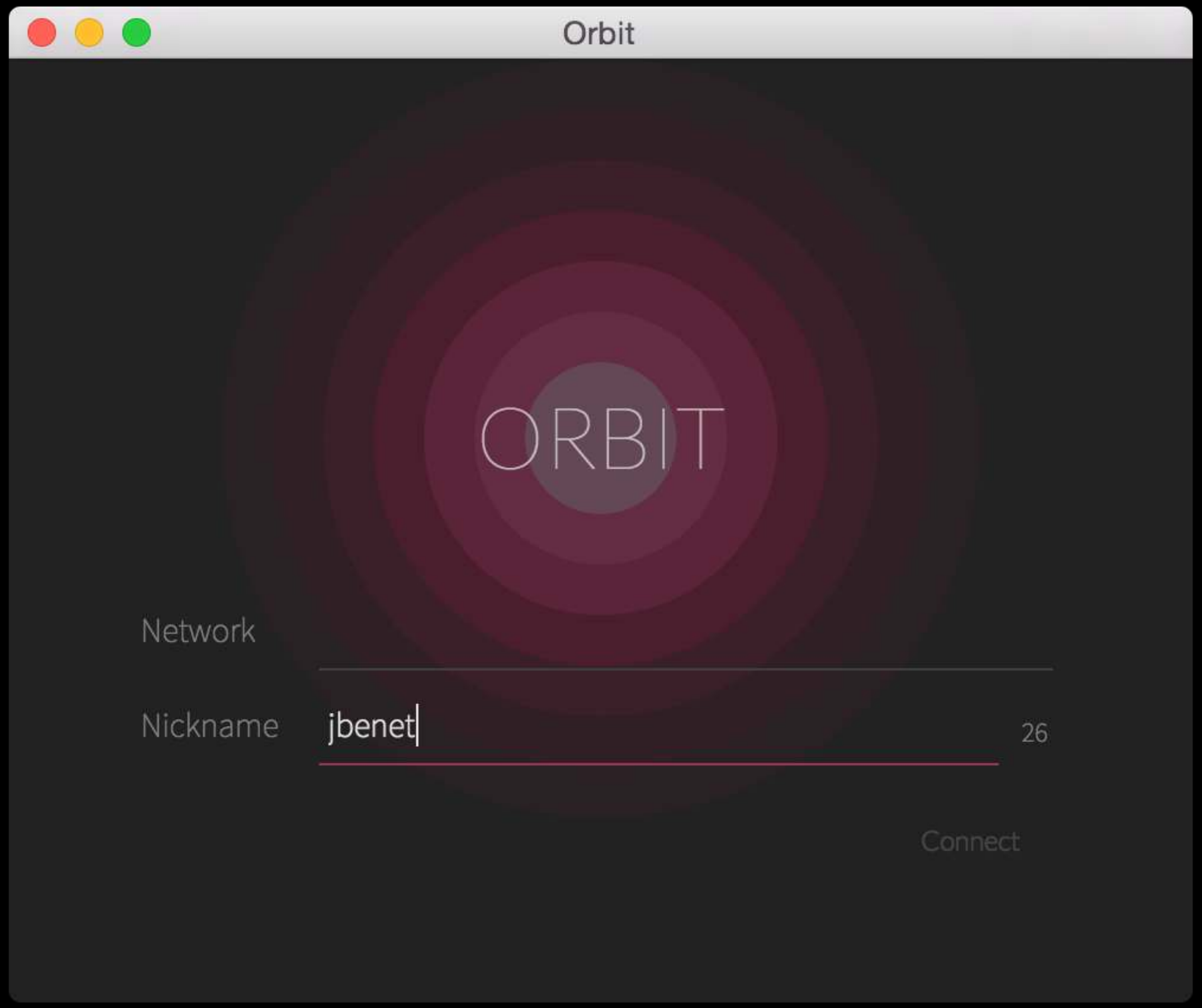


JS

Orbit

github.com/haadcode/orbit

p2p chat on IPFS



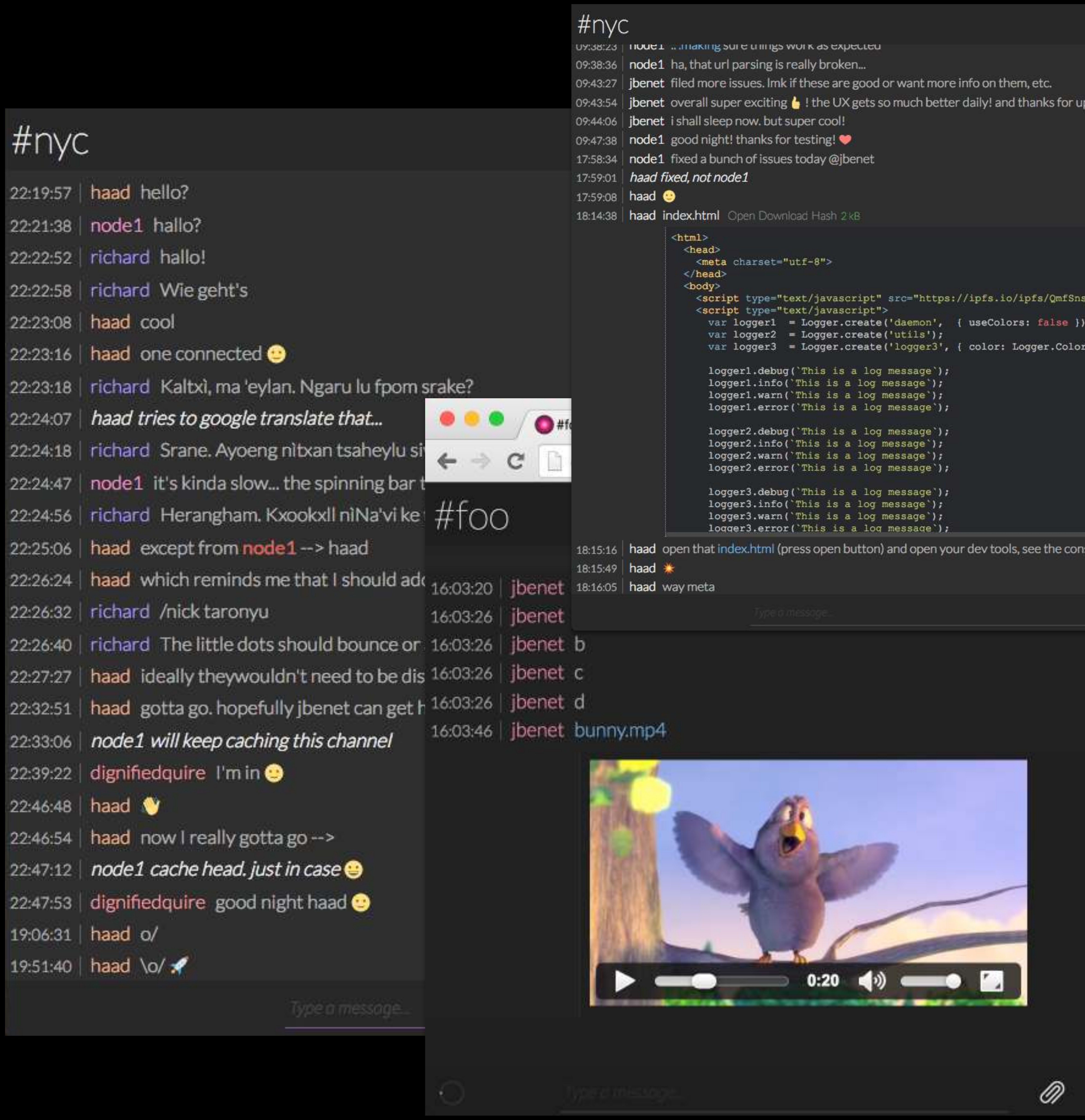
Network

Nickname jbenet 26

IPFS

IPLD

libp2p



#nyc

```
22:19:57 | haad hello?
22:21:38 | node1 hallo?
22:22:52 | richard hallo!
22:22:58 | richard Wie geht's
22:23:08 | haad cool
22:23:16 | haad one connected 🙂
22:23:18 | richard Kaltxì, ma 'eylan. Ngaru lu fpom srake?
22:24:07 | haad tries to google translate that...
22:24:18 | richard Srane. Ayoeng nìtxan tsaheylu si
22:24:47 | node1 it's kinda slow... the spinning bar t
22:24:56 | richard Herangham. Kxookxll niNa'vi ke
22:25:06 | haad except from node1 --> haad
22:26:24 | haad which reminds me that I should adc
22:26:32 | richard /nick taronyu
22:26:40 | richard The little dots should bounce or
22:27:27 | haad ideally theywouldn't need to be dis
22:32:51 | haad gotta go. hopefully jbenet can get h
22:33:06 | node1 will keep caching this channel
22:39:22 | dignifiedquire I'm in 🙂
22:46:48 | haad 👋
22:46:54 | haad now I really gotta go -->
22:47:12 | node1 cache head. just in case 🙂
22:47:53 | dignifiedquire good night haad 🙂
19:06:31 | haad o/
19:51:40 | haad \o/ 🚀
```

Type a message...

#foo

```
16:03:20 | jbenet
16:03:26 | jbenet
16:03:26 | jbenet b
16:03:26 | jbenet c
16:03:26 | jbenet d
16:03:46 | jbenet bunny.mp4
```

0:20

Type a message...

#nyc

```
09:38:23 | node1 ...making sure things work as expected
09:38:36 | node1 ha, that url parsing is really broken...
09:43:27 | jbenet filed more issues. lmk if these are good or want more info on them, etc.
09:43:54 | jbenet overall super exciting 👍 ! the UX gets so much better daily! and thanks for up
09:44:06 | jbenet i shall sleep now. but super cool!
09:47:38 | node1 good night! thanks for testing! ❤
17:58:34 | node1 fixed a bunch of issues today @jbenet
17:59:01 | haad fixed, not node1
17:59:08 | haad 🙂
18:14:38 | haad index.html Open Download Hash 2 kB
```

```
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <script type="text/javascript" src="https://ipfs.io/ipfs/QmfSns
    <script type="text/javascript">
      var logger1  = Logger.create('daemon',  { useColors: false })
      var logger2  = Logger.create('utils');
      var logger3  = Logger.create('logger3', { color: Logger.Color

      logger1.debug(`This is a log message`);
      logger1.info(`This is a log message`);
      logger1.warn(`This is a log message`);
      logger1.error(`This is a log message`);

      logger2.debug(`This is a log message`);
      logger2.info(`This is a log message`);
      logger2.warn(`This is a log message`);
      logger2.error(`This is a log message`);

      logger3.debug(`This is a log message`);
      logger3.info(`This is a log message`);
      logger3.warn(`This is a log message`);
      logger3.error(`This is a log message`);
```

```
18:15:16 | haad open that index.html (press open button) and open your dev tools, see the con
18:15:49 | haad ✴
18:16:05 | haad way meta
```

Type a message...

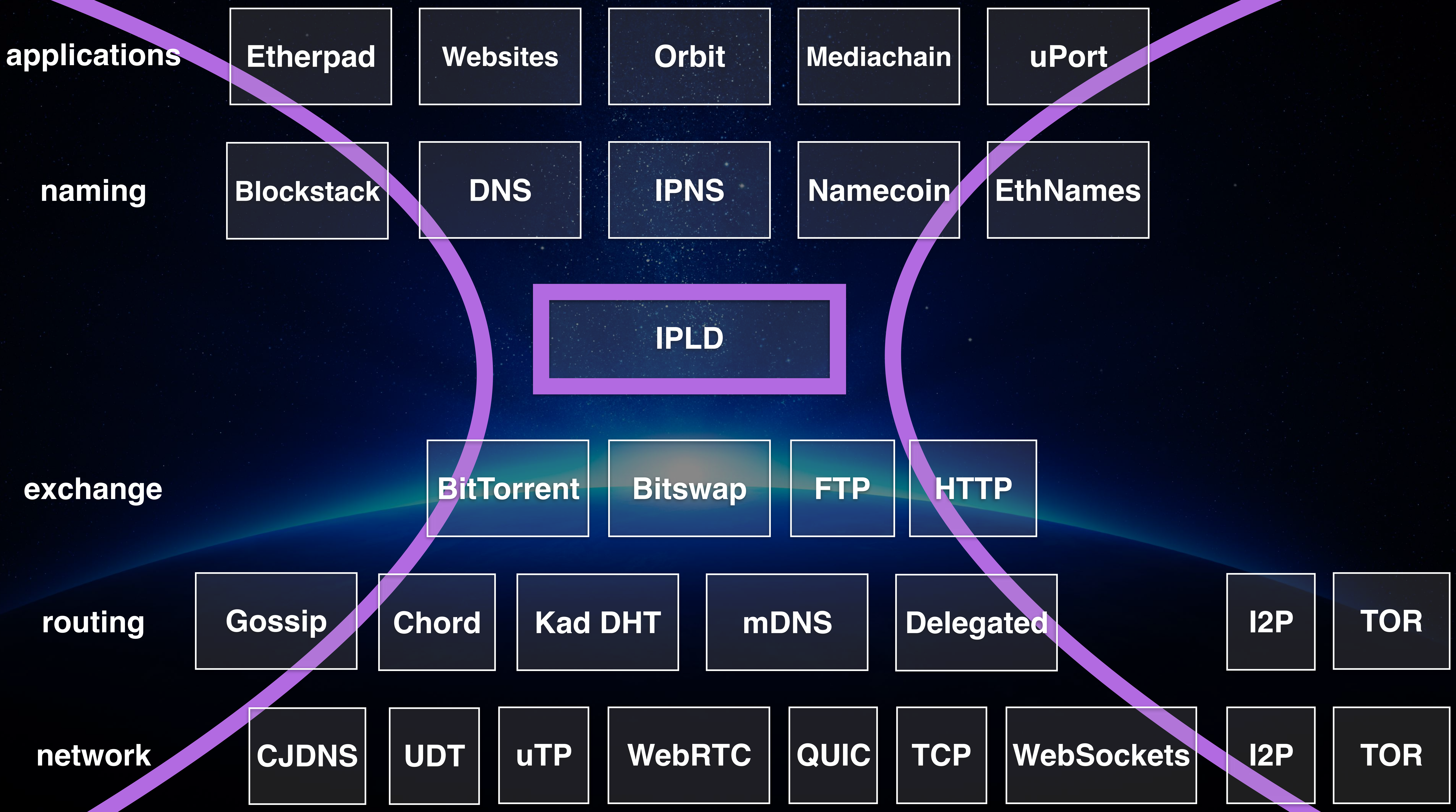
Blockchains and the Web

IPFS

https://ipfs.io
/ipns/ipfs.io

W3C®

@juanbenet
2016-06-29



applications: Etherpad, Websites, Orbit, Mediachain, uPort

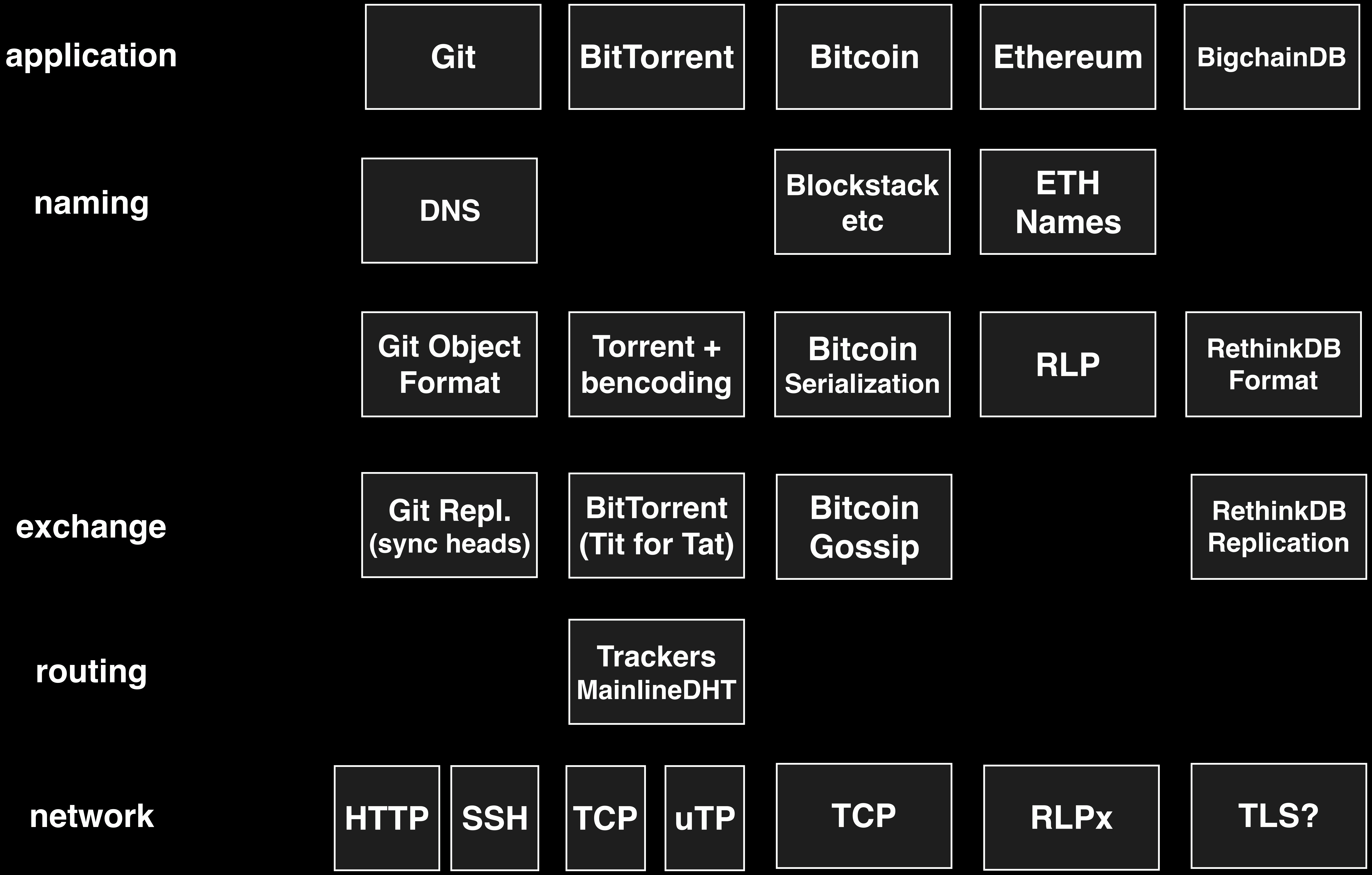
naming: Blockstack, DNS, IPNS, Namecoin, EthNames

IPLD

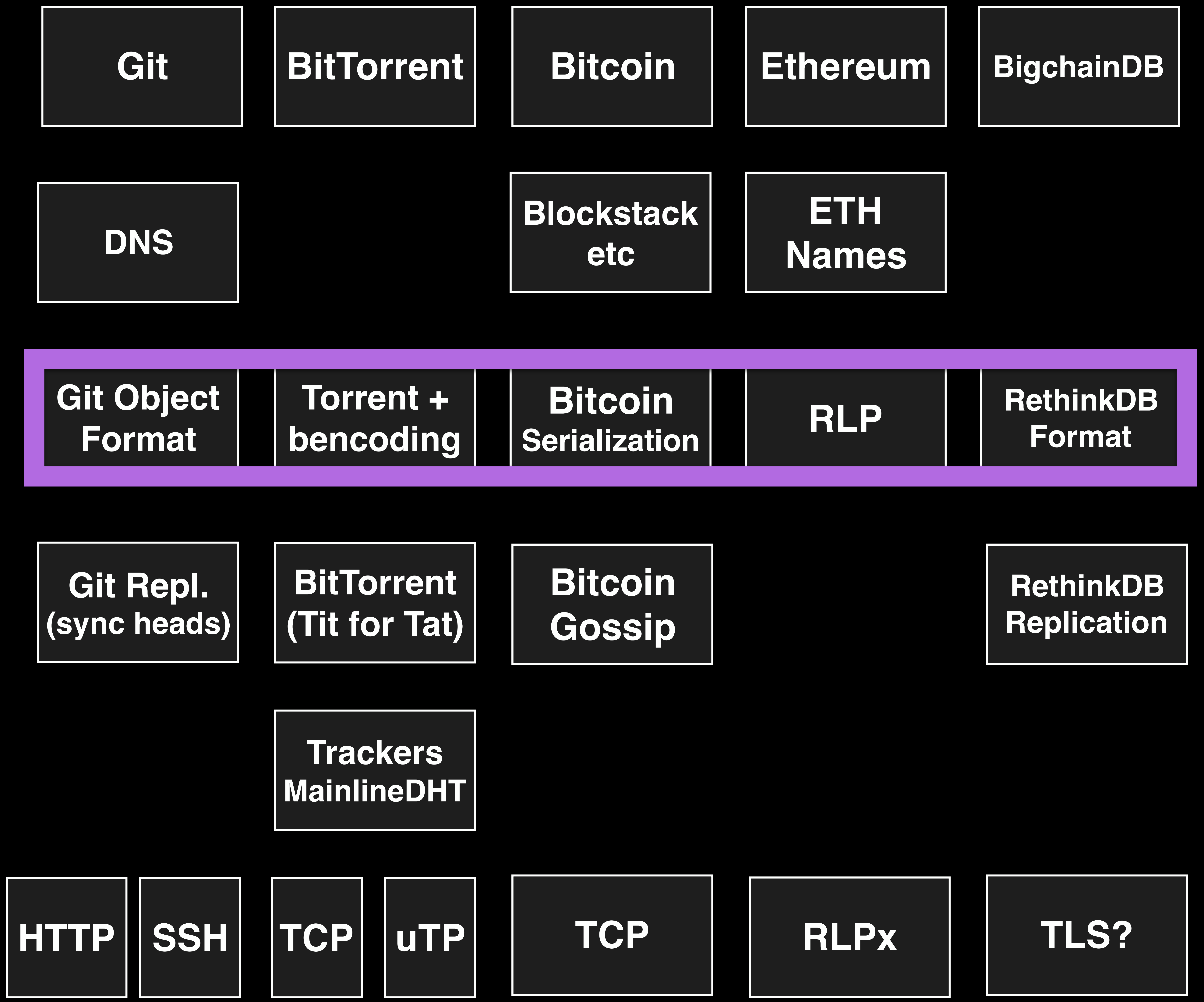
exchange: BitTorrent, Bitswap, FTP, HTTP

routing: Gossip, Chord, Kad DHT, mDNS, Delegated, I2P, TOR

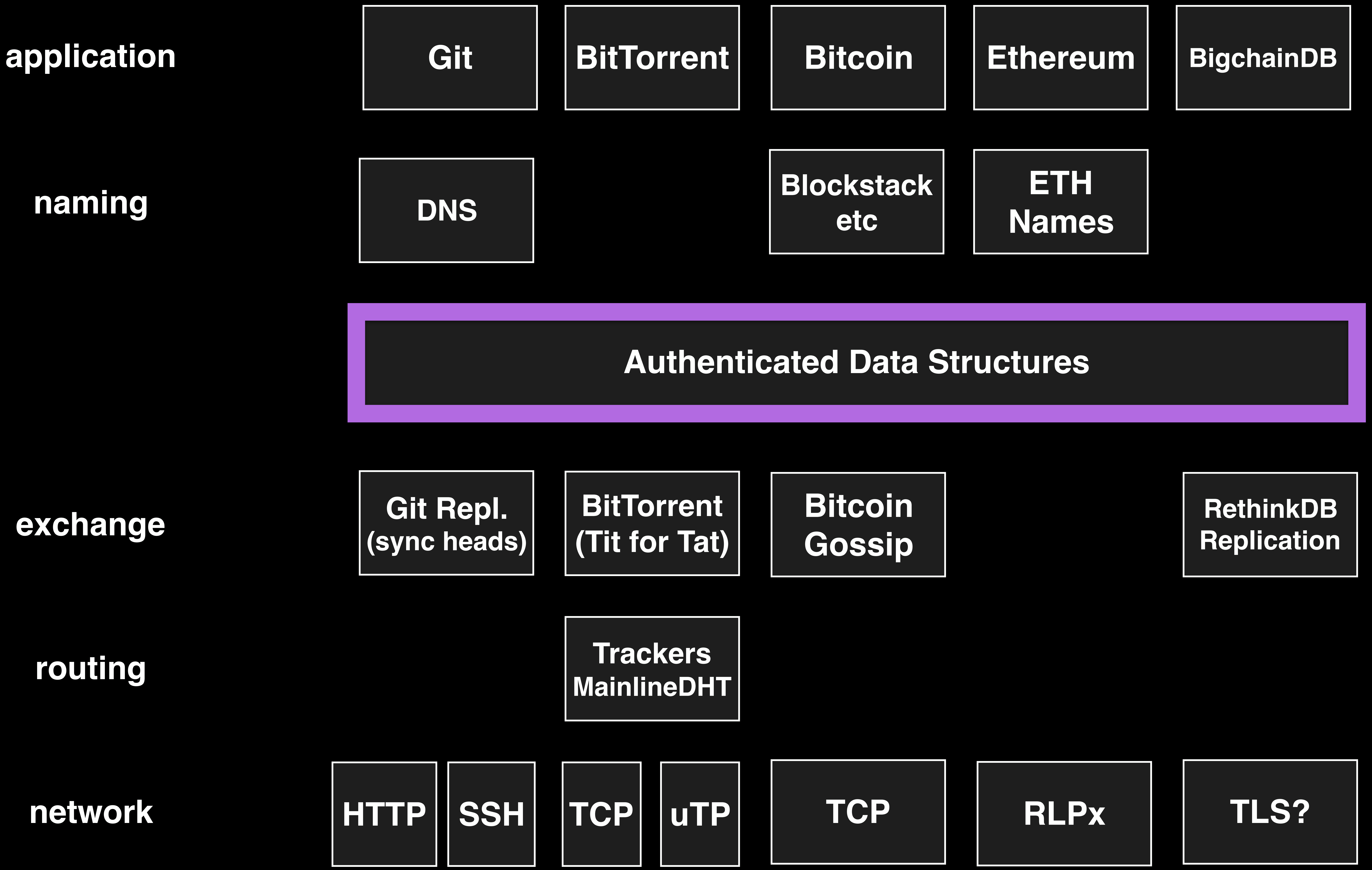
network: CJDNS, UDT, uTP, WebRTC, QUIC, TCP, WebSockets, I2P, TOR



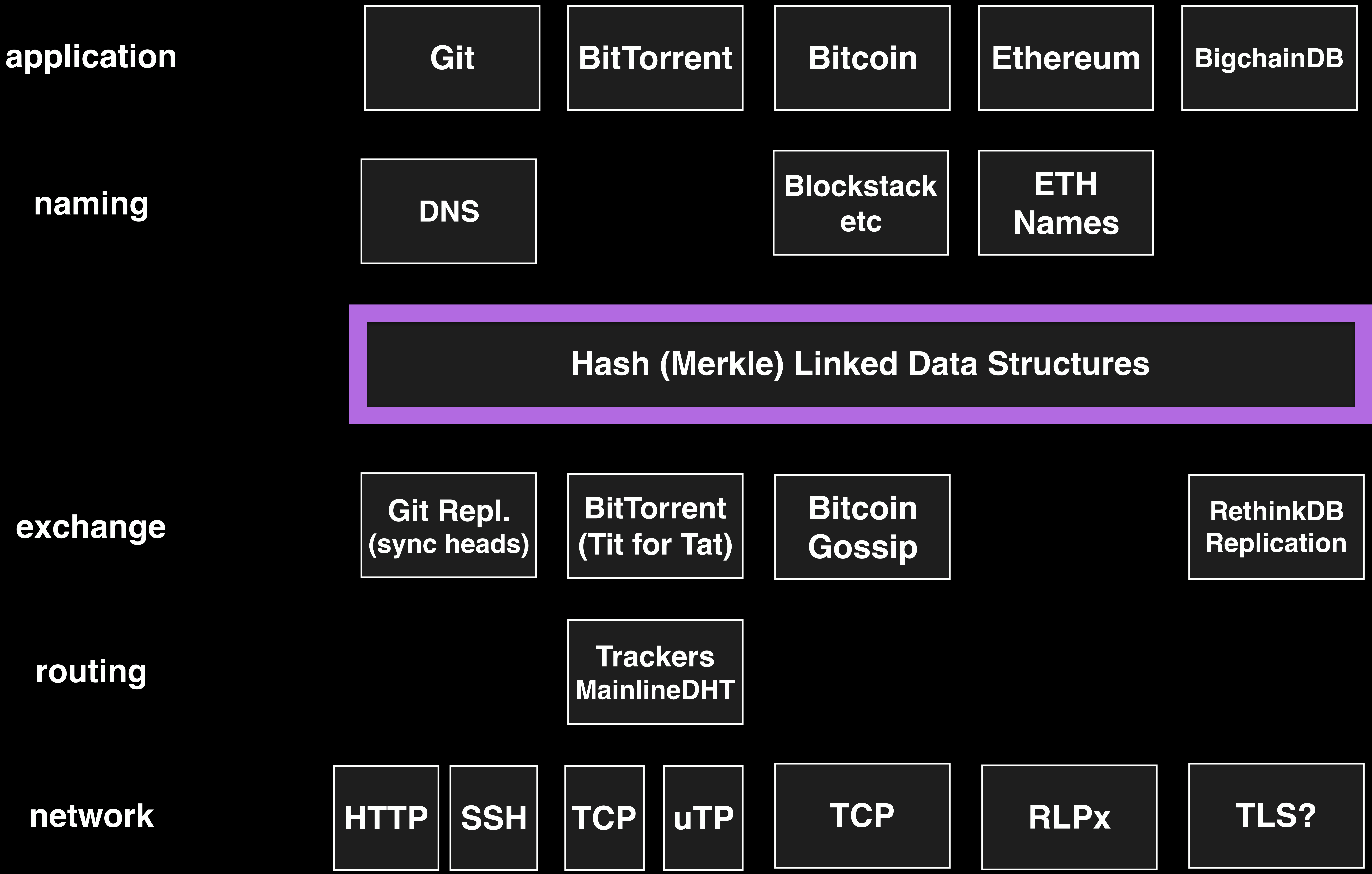
application	Git	BitTorrent	Bitcoin	Ethereum	BigchainDB
naming	DNS		Blockstack etc	ETH Names	
	Git Object Format	Torrent + bencoding	Bitcoin Serialization	RLP	RethinkDB Format
exchange	Git Repl. (sync heads)	BitTorrent (Tit for Tat)	Bitcoin Gossip		RethinkDB Replication
routing		Trackers MainlineDHT			
network	HTTP SSH	TCP uTP	TCP	RLPx	TLS?



application	Git	BitTorrent	Bitcoin	Ethereum	BigchainDB
naming	DNS		Blockstack etc	ETH Names	
	Git Object Format	Torrent + bencoding	Bitcoin Serialization	RLP	RethinkDB Format
exchange	Git Repl. (sync heads)	BitTorrent (Tit for Tat)	Bitcoin Gossip		RethinkDB Replication
routing		Trackers MainlineDHT			
network	HTTP SSH	TCP uTP	TCP	RLPx	TLS?



application	Git	BitTorrent	Bitcoin	Ethereum	BigchainDB
naming	DNS		Blockstack etc	ETH Names	
	Authenticated Data Structures				
exchange	Git Repl. (sync heads)	BitTorrent (Tit for Tat)	Bitcoin Gossip		RethinkDB Replication
routing		Trackers MainlineDHT			
network	HTTP SSH	TCP uTP	TCP	RLPx	TLS?



application	Git	BitTorrent	Bitcoin	Ethereum	BigchainDB
naming	DNS		Blockstack etc	ETH Names	
	Hash (Merkle) Linked Data Structures				
exchange	Git Repl. (sync heads)	BitTorrent (Tit for Tat)	Bitcoin Gossip		RethinkDB Replication
routing		Trackers MainlineDHT			
network	HTTP SSH	TCP uTP	TCP	RLPx	TLS?

application	Git	BitTorrent	Bitcoin	Ethereum	BigchainDB
naming		DNS	Blockstack etc	ETH Names	
		IPLD			
exchange	Git Repl. (sync heads)	BitTorrent (Tit for Tat)	Bitcoin Gossip	RethinkDB Replication	
routing		Trackers MainlineDHT			
network	HTTP SSH	TCP uTP	TCP	RLPx	TLS?