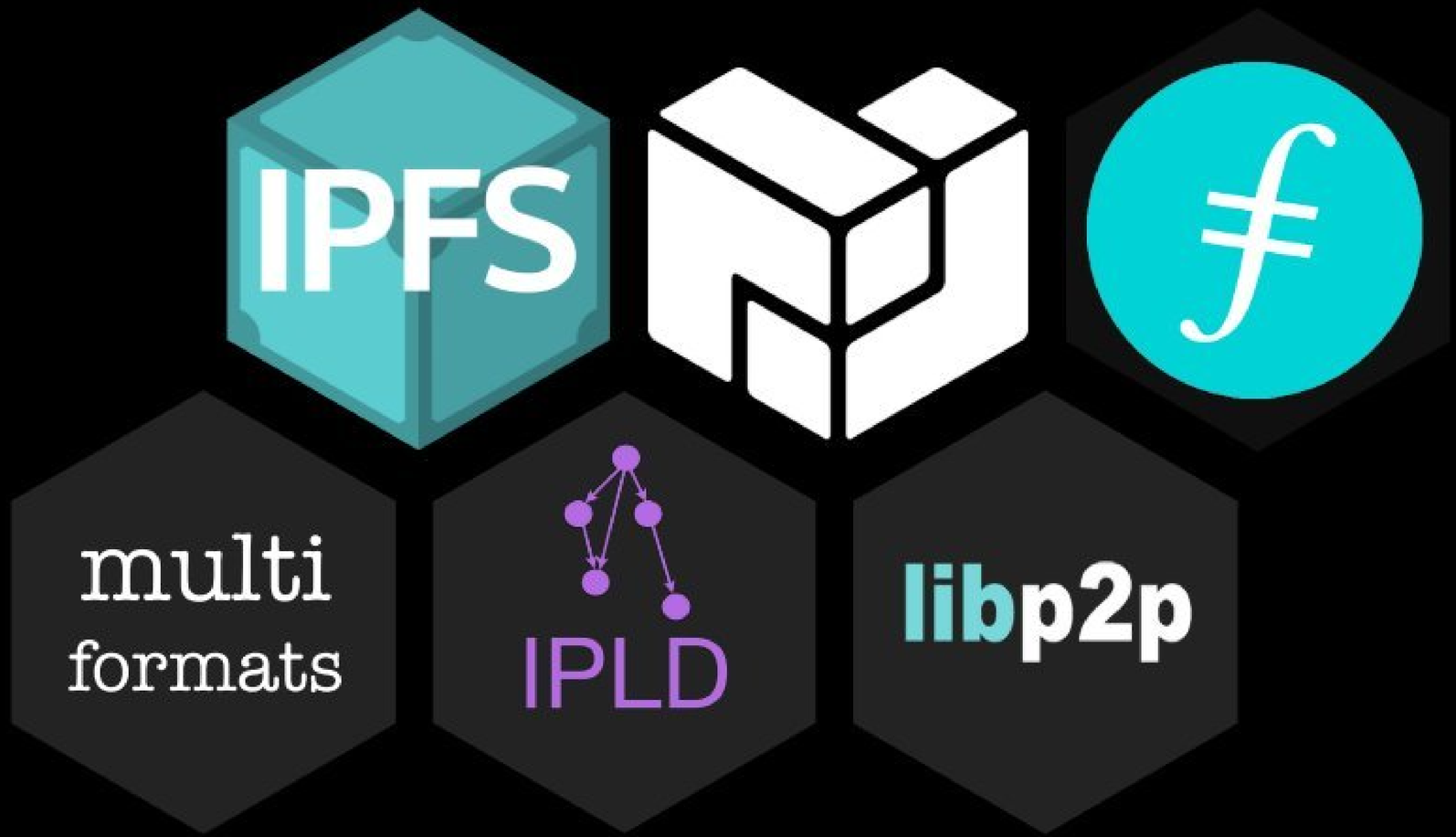
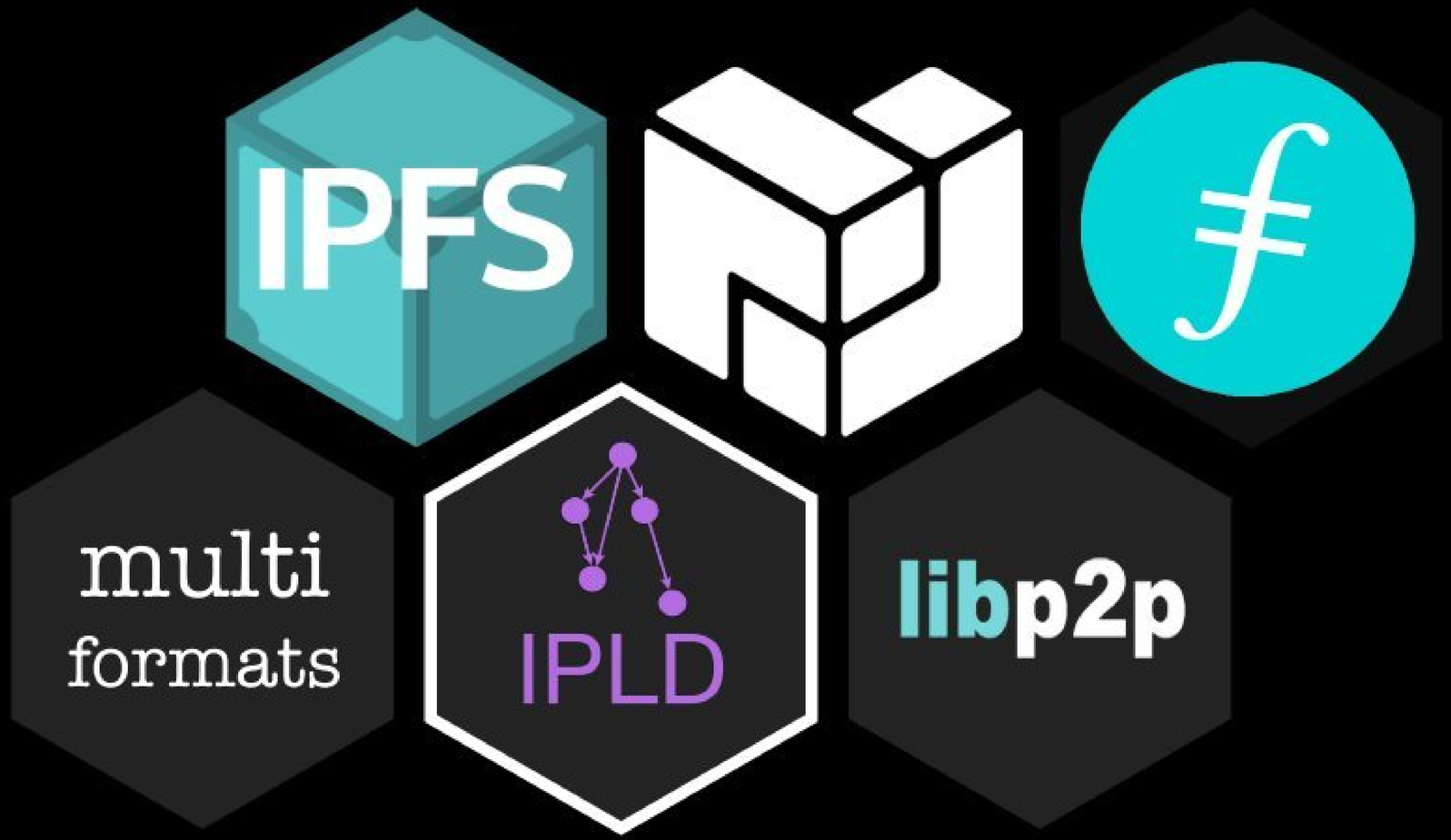
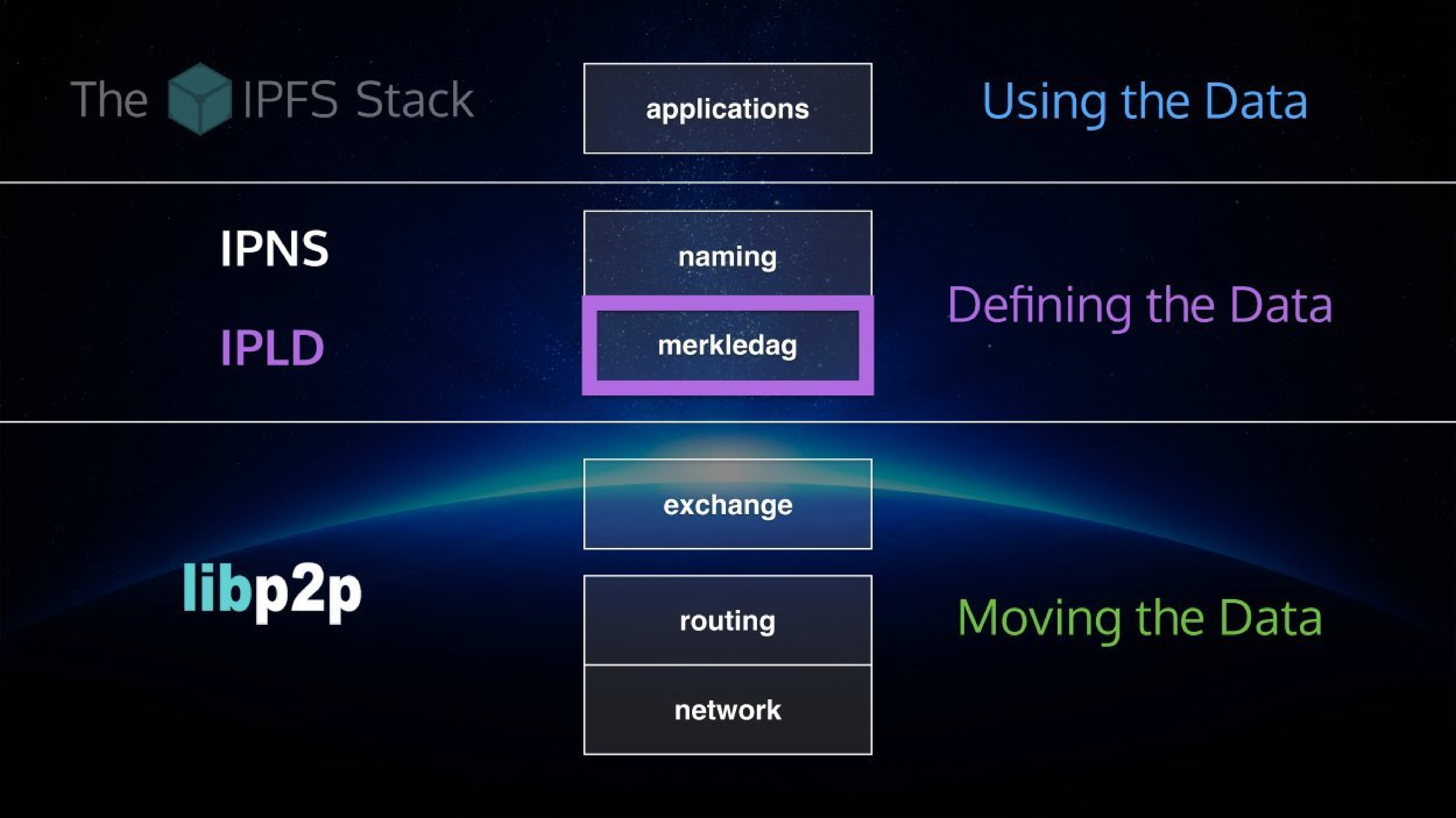
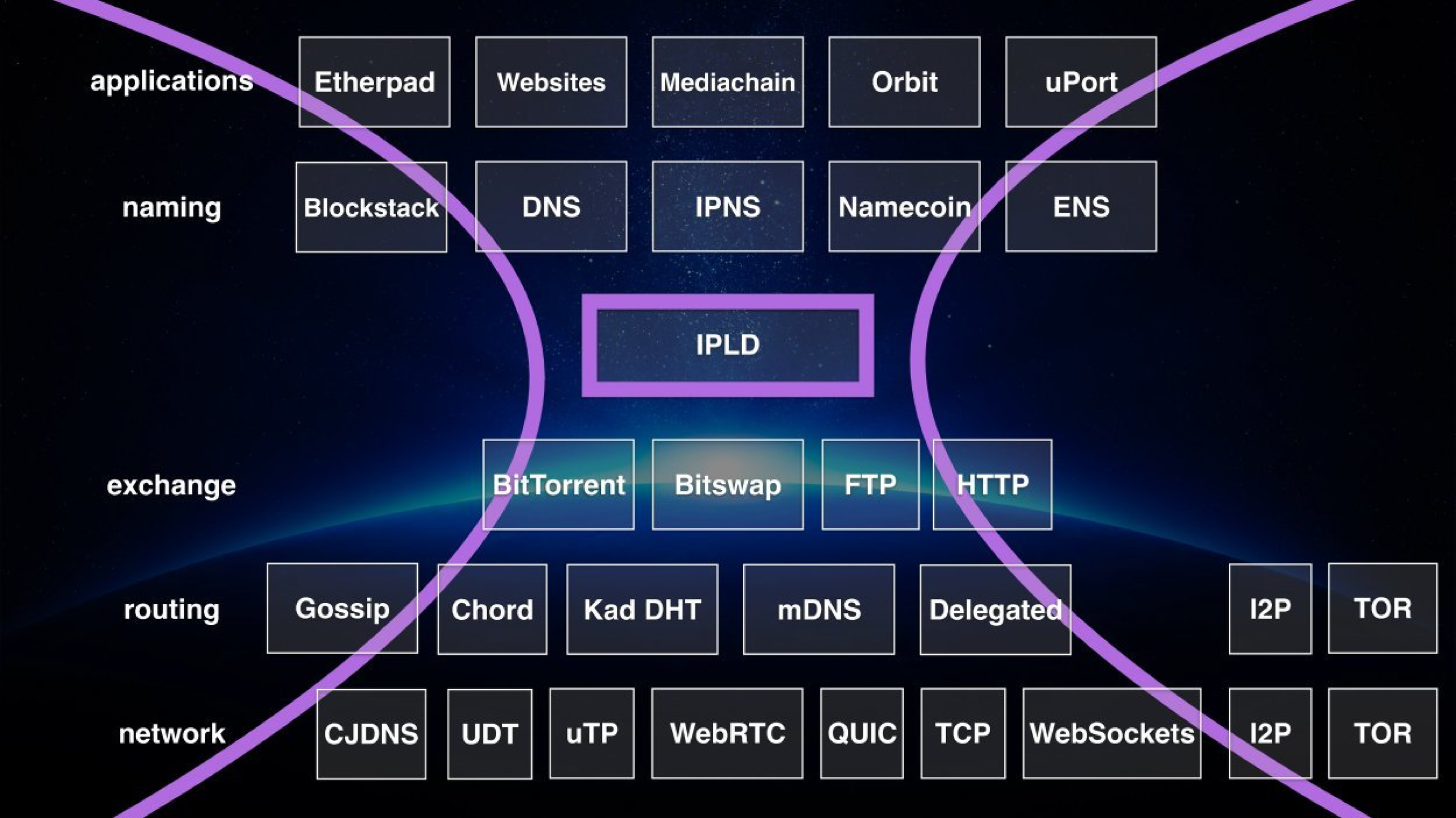
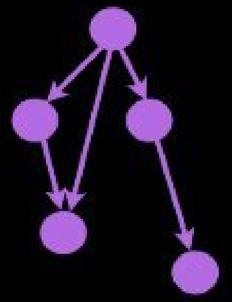

IPLD: Enter the Merkle Forest


IPFS
multi
formats
IPLD
libp2p


IPFS
multi
formats
IPLD
libp2p

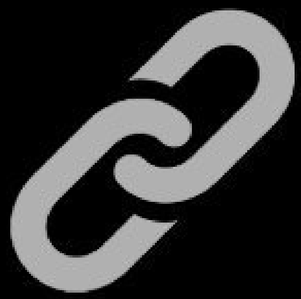

The IPFS Stack
applications
Using the Data
IPNS
IPLD
naming
merkledag
Defining the Data
libp2p
exchange
routing
network
Moving the Data

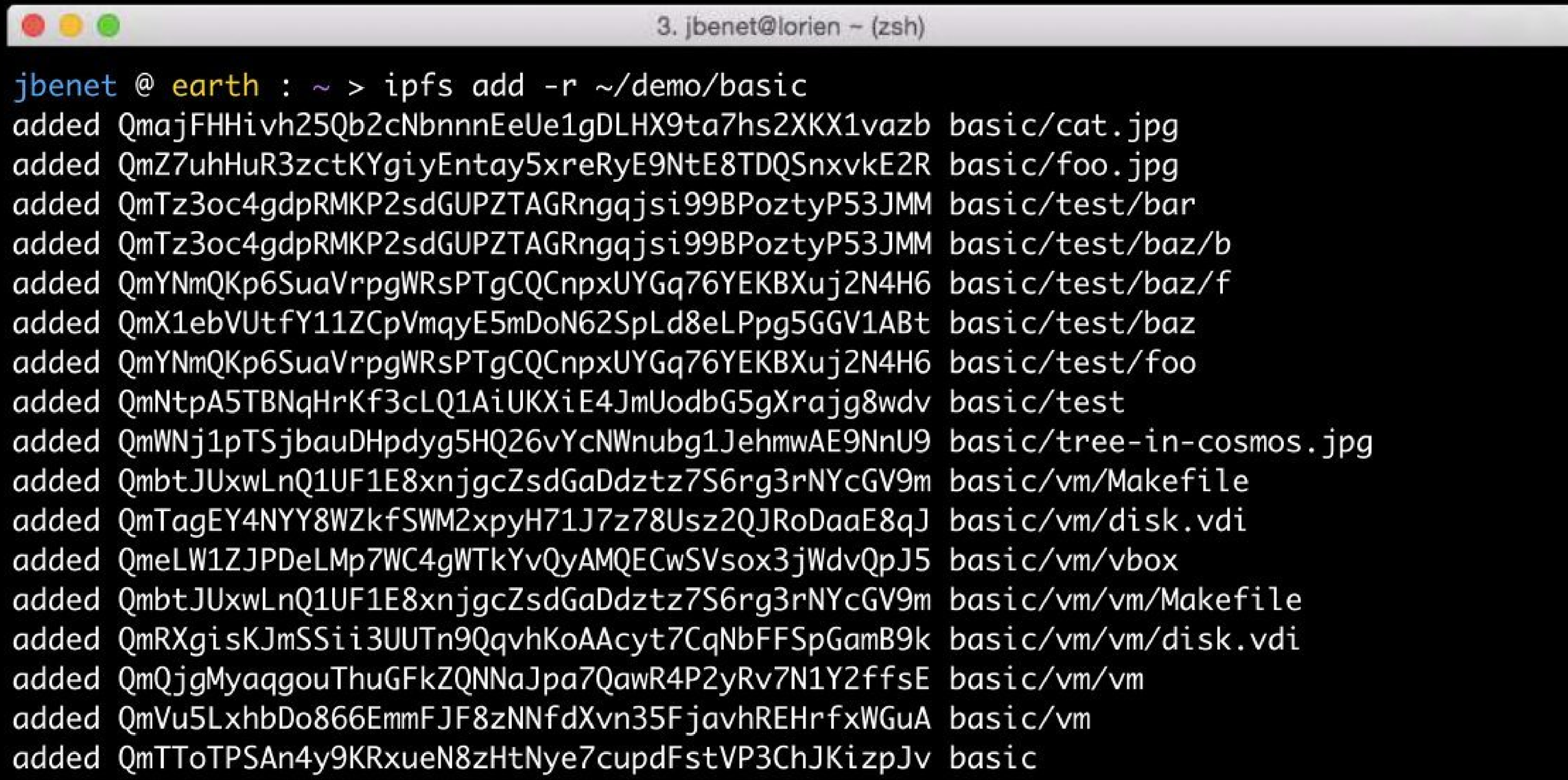

applications
Etherpad
Websites
Mediachain
Orbit
uPort
naming
Blockstack
DNS
IPNS
Namecoin
ENS
IPLD
exchange
BitTorrent
Bitswap
FTP
HTTP
routing
Gossip
Chord
Kad DHT
mDNS
Delegated
I2P
TOR
network
CJDNS
UDT
uTP
WebRTC
QUIC
TCP
WebSockets
I2P
TOR

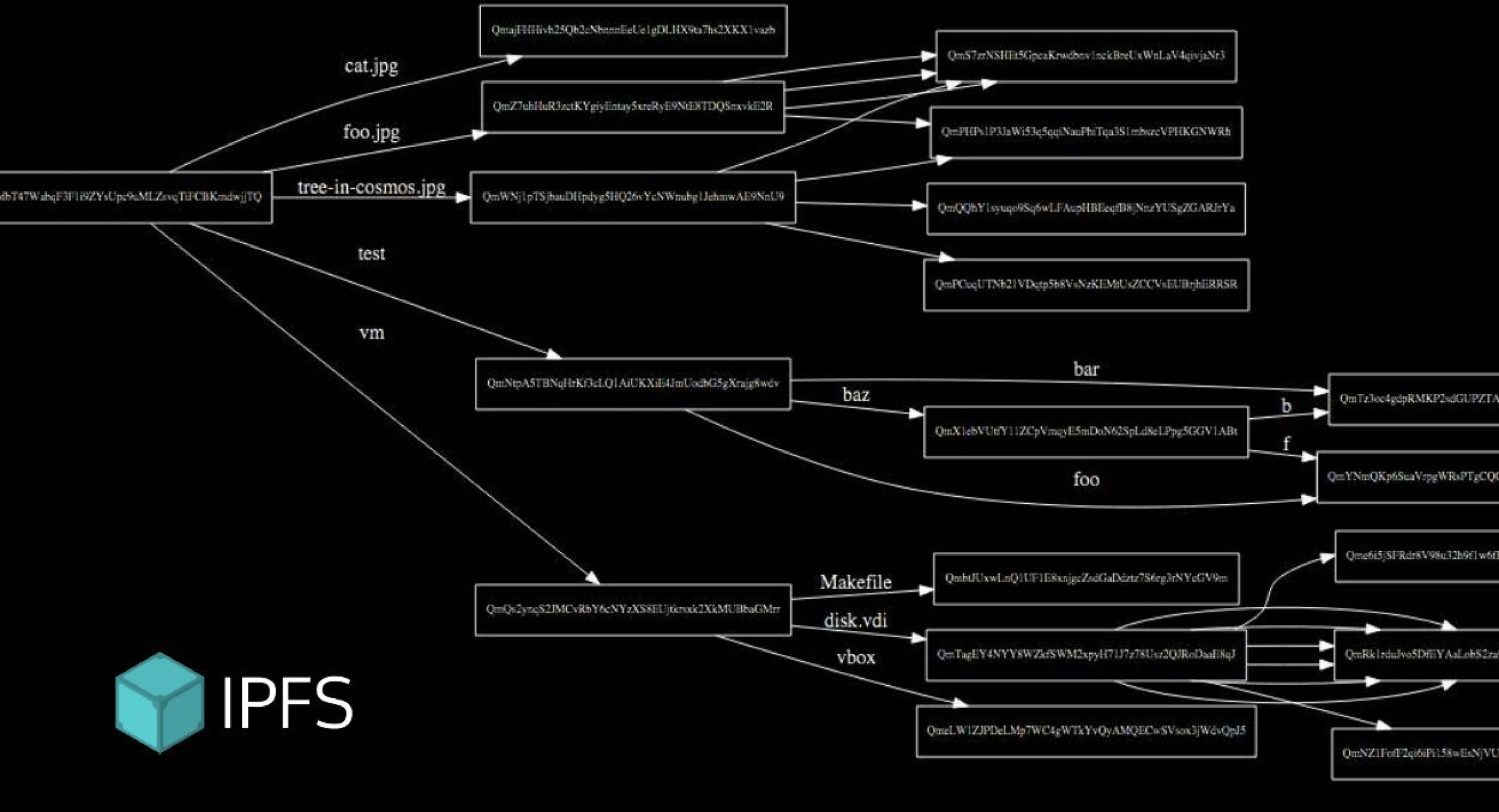


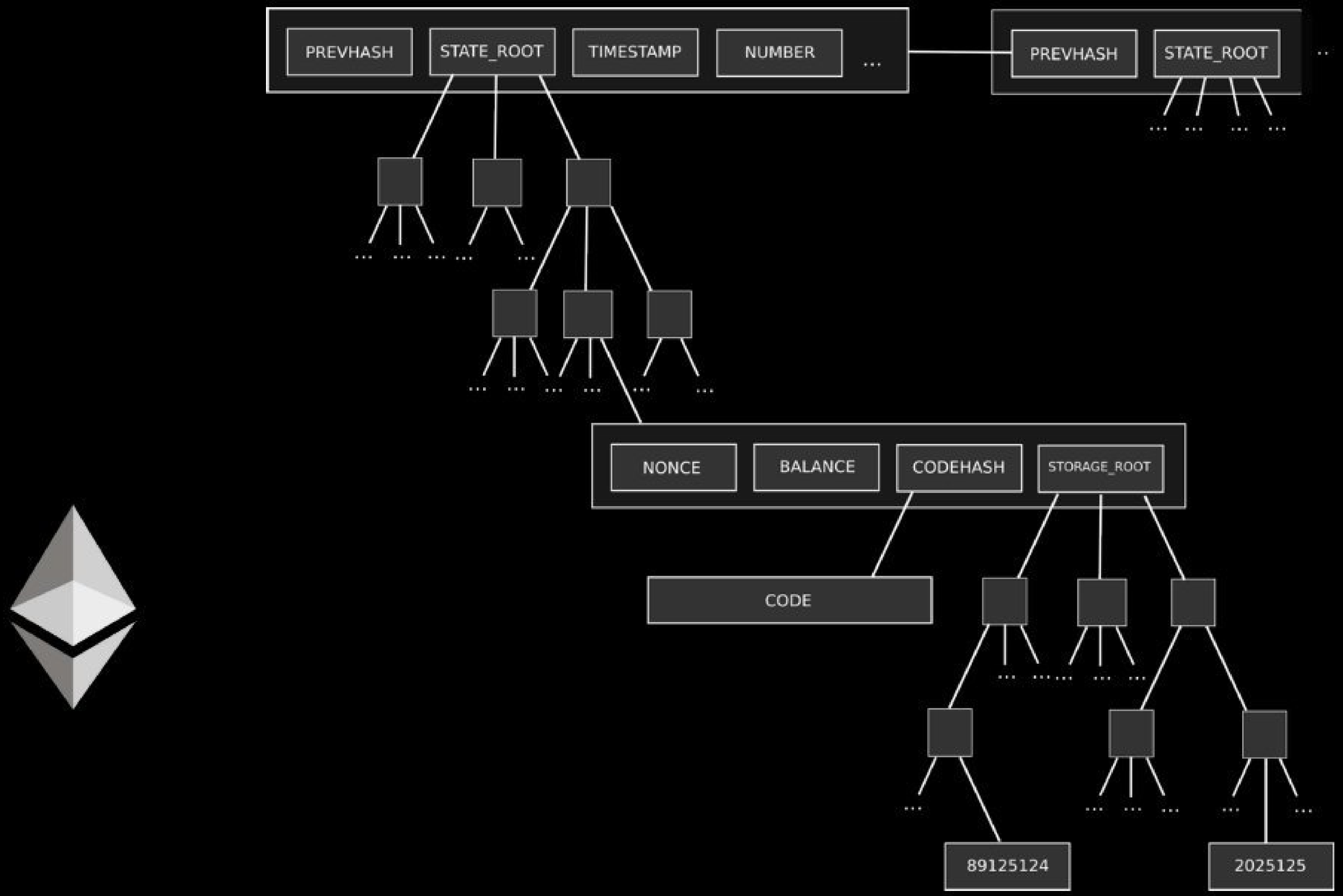
# IPLD

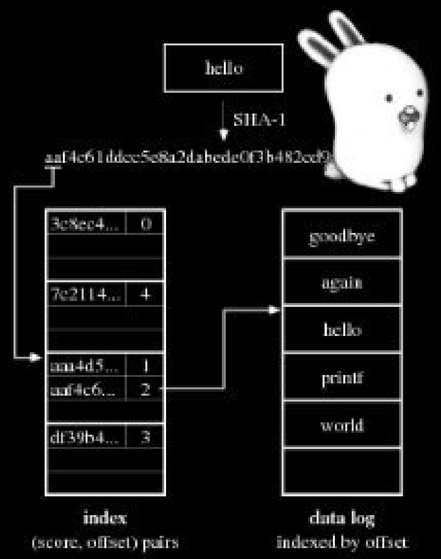
intuition

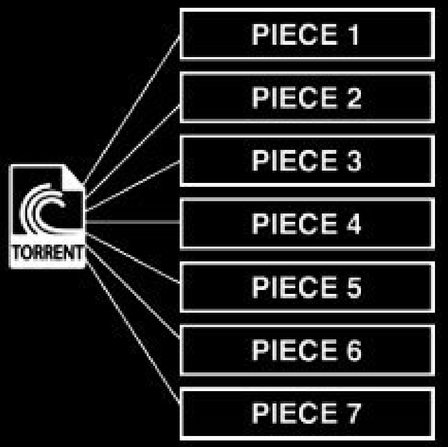


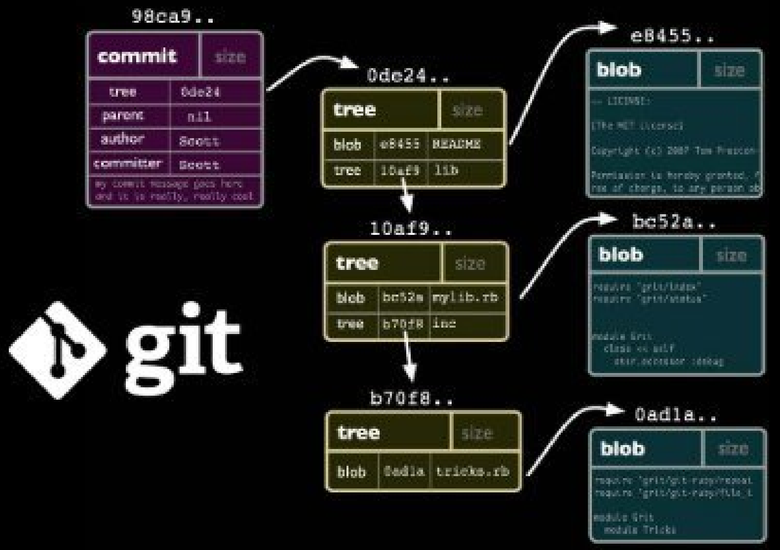

3. jbenet@lorien ~ (zsh)
jbenet @ earth : ~ > ipfs add -r ~/demo/basic
added QmajFHHivh25Qb2cNbnnnEeUe1gDLHX9ta7hs2XKX1vazb basic/cat.jpg
added QmZ7uhHuR3zctKYgiyEntay5xreRyE9NtE8TDQSnxvkE2R basic/foo.jpg
added QmTz3oc4gdpRMKP2sdGUPZTAGRngqjsi99BPoztyP53JMM basic/test/bar
added QmTz3oc4gdpRMKP2sdGUPZTAGRngqjsi99BPoztyP53JMM basic/test/baz/b
added QmYNmQKp6SuaVrpgWRsPTgCQCnpxUYGq76YEKBXuj2N4H6 basic/test/baz/f
added QmX1ebVUtfY11ZCpVmqyE5mDoN62SpLd8eLPpg5GGV1ABt basic/test/baz
added QmYNmQKp6SuaVrpgWRsPTgCQCnpxUYGq76YEKBXuj2N4H6 basic/test/foo
added QmNtpA5TBNqHrKf3cLQ1AiUKXiE4JmUodbG5gXrajg8wdv basic/test
added QmWNj1pTSjbauDHpdyg5HQ26vYcNWnubg1JehmwAE9NnU9 basic/tree-in-cosmos.jpg
added QmbtJUxwLnQ1UF1E8xnjgcZsdGaDdztz7S6rg3rNYcGV9m basic/vm/Makefile
added QmTagEY4NYY8WZkfSWM2xpyH71J7z78Usz2QJRoDaaE8qJ basic/vm/disk.vdi
added QmeLW1ZJPDeLMp7WC4gWTkYvQyAMQECwSVsox3jWdvQpJ5 basic/vm/vbox
added QmbtJUxwLnQ1UF1E8xnjgcZsdGaDdztz7S6rg3rNYcGV9m basic/vm/vm/Makefile
added QmRXgisKJmSSii3UUTn9QqvhKoAAcyt7CqNbFFSpGamB9k basic/vm/vm/disk.vdi
added QmQjgMyaqgouThuGFkZQNNaJpa7QawR4P2yRv7N1Y2ffsE basic/vm/vm
added QmVu5LxhbDo866EmmFJF8zNNfdXvn35FjavhREHrfxWGuA basic/vm
added QmTToTPSAn4y9KRxueN8zHtNye7cupdFstVP3ChJKizpJv basic

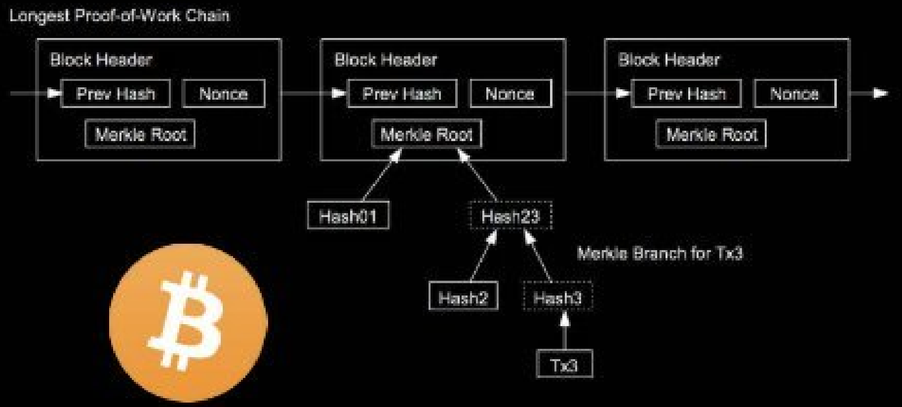

QmajHHHvh25Qb2cNbnnnEieUe1gDLHX9ta7hs2XXX1vazb
cat.jpg
QmS7zrNSHEt5GpcaKrwdbrv1nckBreUxWnLaV4qivjaNr3
QmZ7uhHuR3zctKYgivEntay5xreRyE9NtE8TDQ8nxvkE2R
foo.jpg
QmPHPx1P3JaWi53q5qqtNauPhiTqa3S1mbszcVPHKGNWRh
tree-in-cosmos.jpg
QmWNj1pTSjbazDHpdyg5HQ26vYcNWnubg1JehmwAE9NrU9
QmQQhY1syuqo8Sq6wLFAcpHBEeqfB8jNnzYUSgZGARJrYa
test
QmPCucUTNh21VDqtp5b8VsNzKEMtUsZCCVsEUBrjhERRSR
vm
QmNtpA5TBNqHrKf3cLQ1AiUKXiE4JmUodbG5gXrajg8wdv
bar
baz
QmTz3oc4gdpRMKP2sdGUPZTA
QmX1ebVUtfY11ZCpVmqyE5mDoN62SpLd8eLPpg5GGV1ABt
b
f
foo
QmYNmQKp6SuaVrpgWRsPTgCQC
QmQs2ynqS2JMCvRbY6cNYzXS8EUjt6rnxk2XkMUBbaGMrr
Makefile
QmbtJUxwLnQ1UF1E8xnjgcZsdGaDztz7S6rg3rNYcGV9m
disk.vdi
vbox
QmTagEY4NYYSWZcfSWM2xpyH71J7z78Usz2QJRolDaaE8qJ
Qme6i5jSFRdr8V98u32h9f1w6f
QmRk1rduJvo5DfEYAaLnbS2ra
QmeLW1ZJPDeLMp7WC4gWTkYvQyAMQECwSVsox3jWdvQpJ5
QmNZ1FotF2qi6tPi158wEsNjVU
IPFS

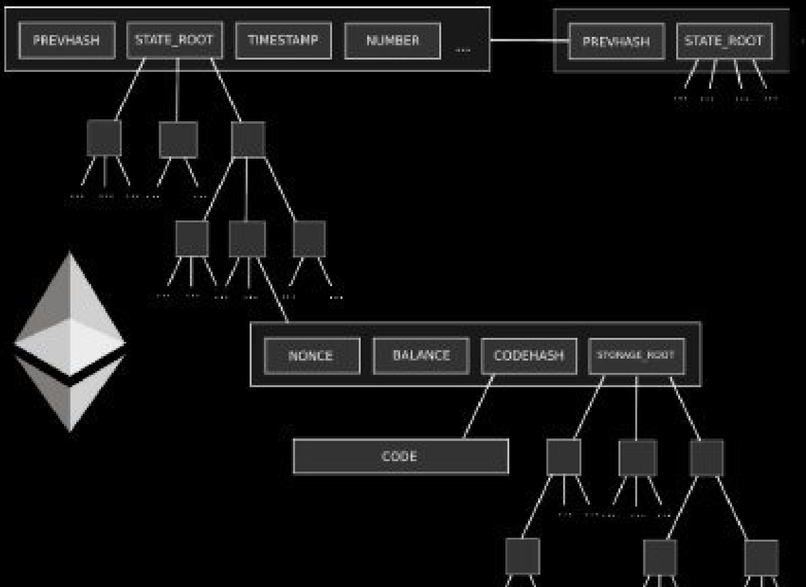

PREVHASH
STATE_ROOT
TIMESTAMP
NUMBER
PREVHASH
STATE_ROOT
NONCE
BALANCE
CODEHASH
STORAGE_ROOT
CODE
89125124
2025125

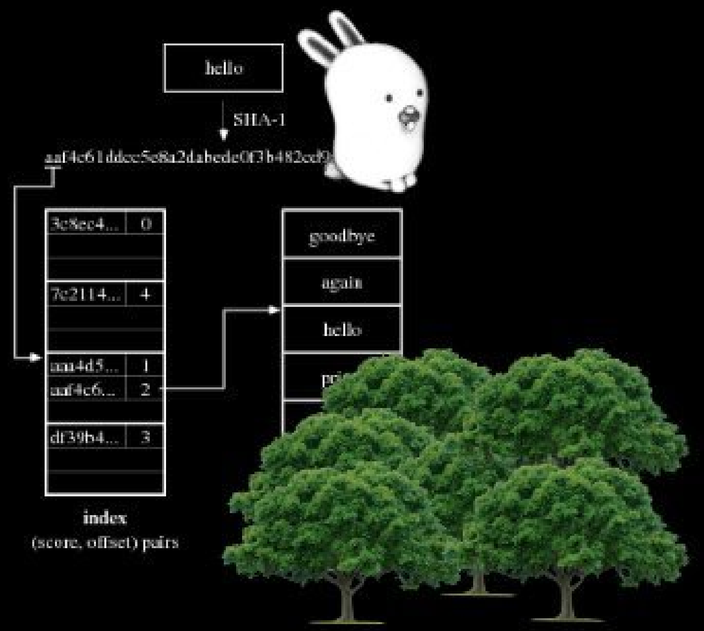

hello
SHA-1
aaf4c61ddcc5e8a2dabede0f3b482cd9
3c8ec4... 0
7c2114... 4
aaa4d5... 1
aaf4c6... 2
df39b4... 3
goodbye
again
hello
printf
world
index
(score, offset) pairs
data log
indexed by offset

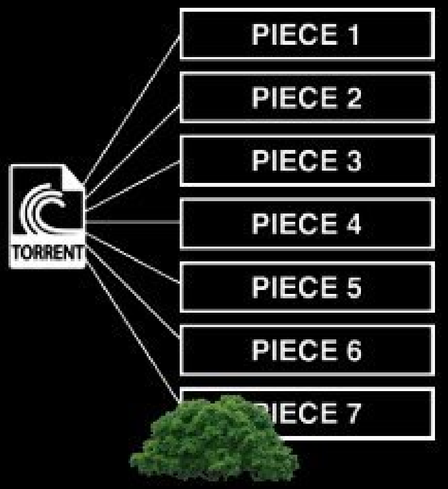

TORRENT
PIECE 1
PIECE 2
PIECE 3
PIECE 4
PIECE 5
PIECE 6
PIECE 7

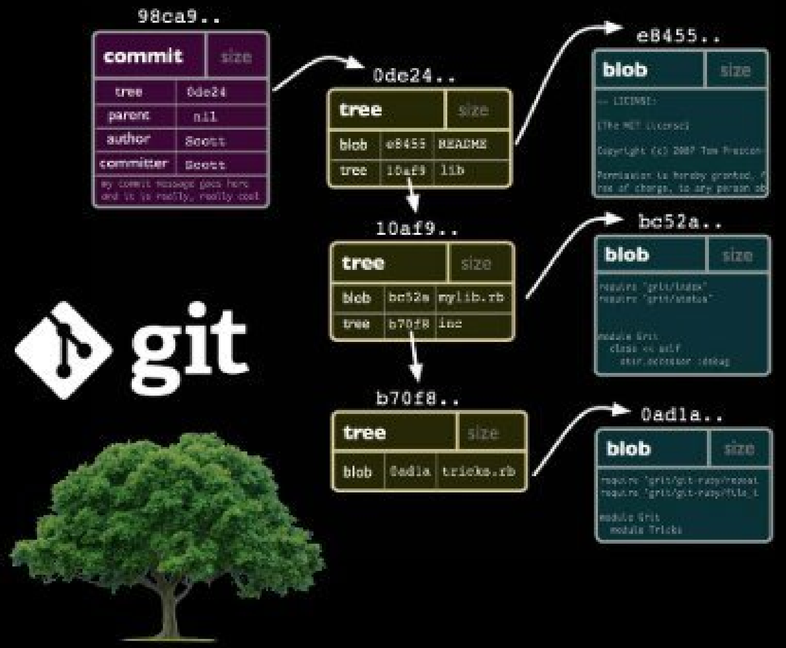

98ca9..
commit
size
tree 0dc24
parent nil
author Scott
committer Scott
0dc24..
tree
size
blob e8455 README
tree 10af9 lib
e8455..
blob
size
10af9..
tree
size
blob bc52a mylib.rb
tree b70f8 inc
bc52a..
blob
size
b70f8..
tree
size
blob 0ad1a tricks.rb
0ad1a..
blob
size
git

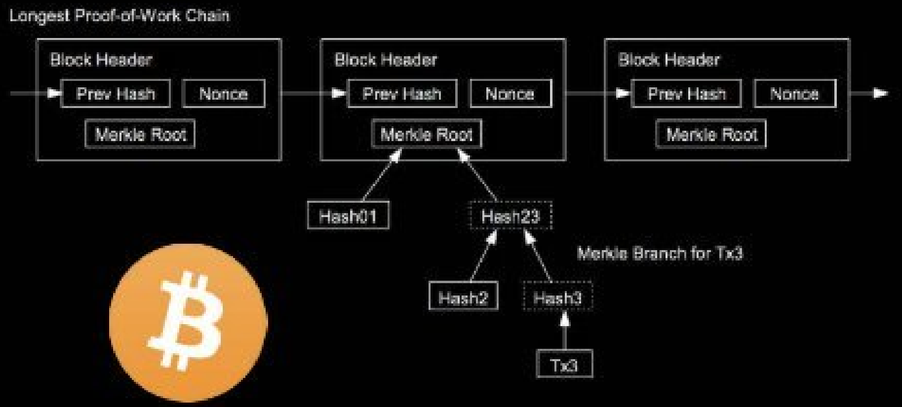

Longest Proof-of-Work Chain
Block Header
Prev Hash
Nonce
Merkle Root
Block Header
Prev Hash
Nonce
Merkle Root
Block Header
Prev Hash
Nonce
Merkle Root
Hash01
Hash23
Merkle Branch for Tx3
Hash2
Hash3
Tx3

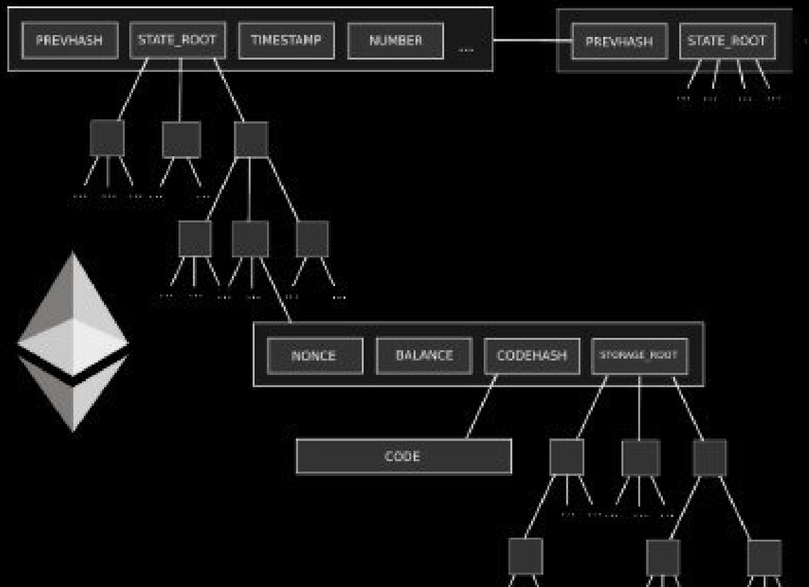

PREVHASH
STATE_ROOT
TIMESTAMP
NUMBER
...
PREVHASH
STATE_ROOT
NONCE
BALANCE
CODEHASH
STORAGE ROOT
CODE

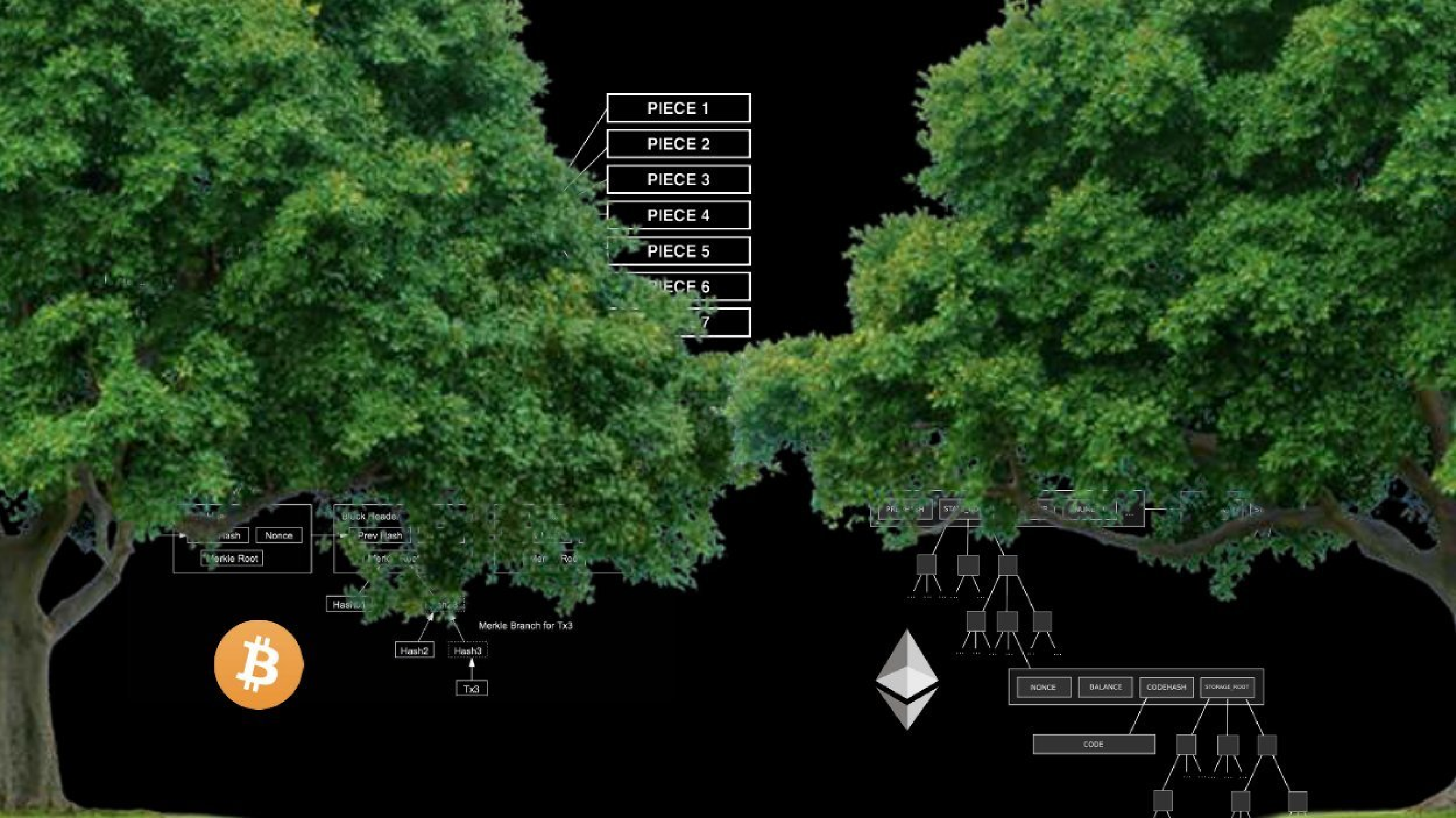

hello
SHA-1
aaf4c61ddcc5e8a2dabede0f3b482cd9
3c8ec4... 0
7c2114... 4
aaa4d5... 1
aaf4c6... 2
df39b4... 3
index
(score, offset) pairs
goodbye
again
hello


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

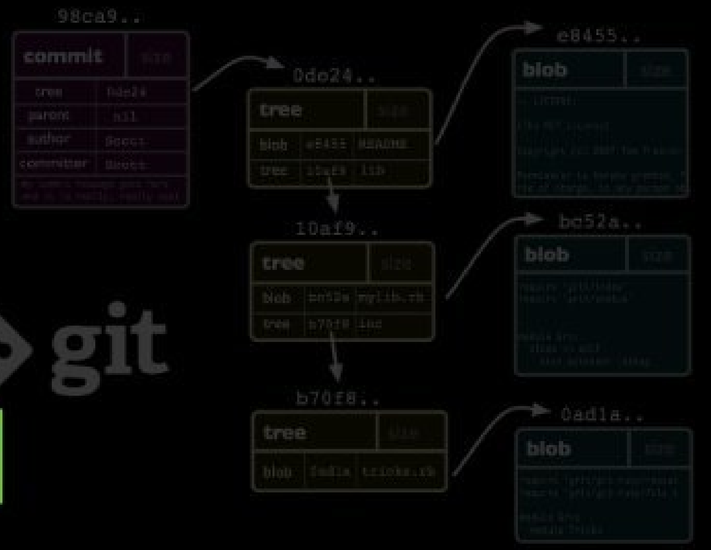

98ca9..
commit
size
tree 0dc24
parent nil
author Scott
committer Scott
0dc24..
tree
size
blob e8455 README
tree 10af9 lib
e8455..
blob
size
10af9..
tree
size
blob bc52a mylib.rb
tree b70f8 inc
bc52a..
blob
size
b70f8..
tree
size
blob 0ad1a tricks.rb
0ad1a..
blob
size
git


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

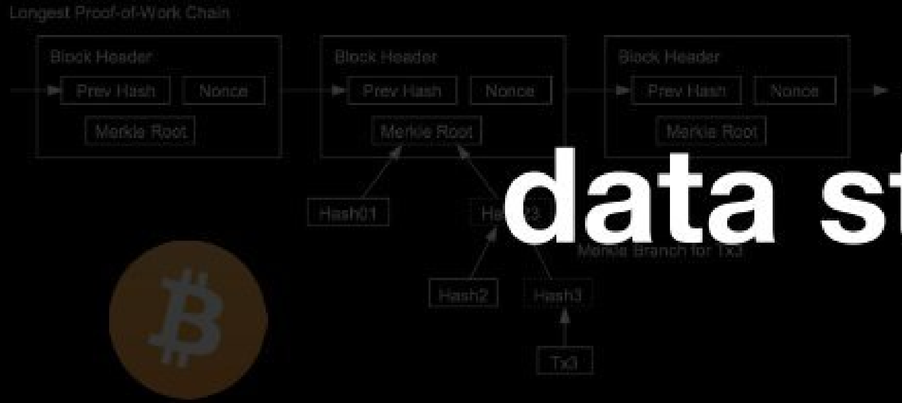

PREVHASH
STATE_ROOT
TIMESTAMP
NUMBER
...
PREVHASH
STATE_ROOT
NONCE
BALANCE
CODEHASH
STORAGE ROOT
CODE

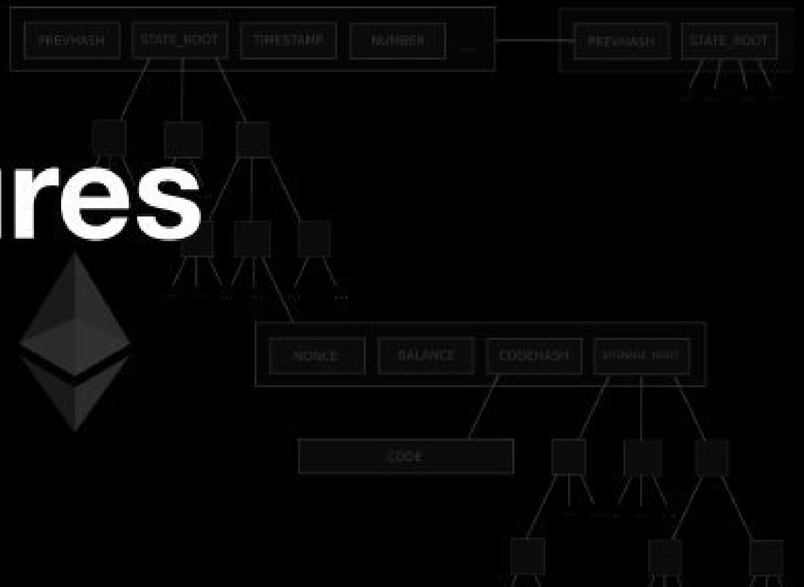

PIECE 1
PIECE 2
PIECE 3
PIECE 4
PIECE 5
7
Nonce
Merkle Root
Prev Hash
Merkle Branch for Tx3
Hash2
Hash3
Tx3
NONCE
BALANCE
CODEHASH
CODE






# distributed
# authenticated
# hash-linked

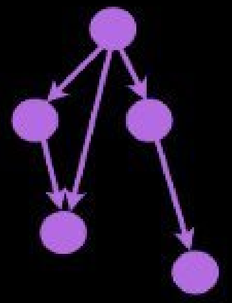



# data structures

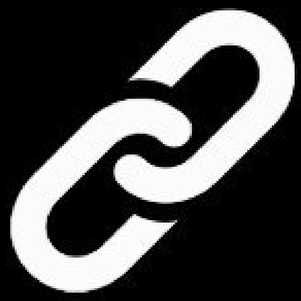
# IPFS is like a forest of linked merkle-trees

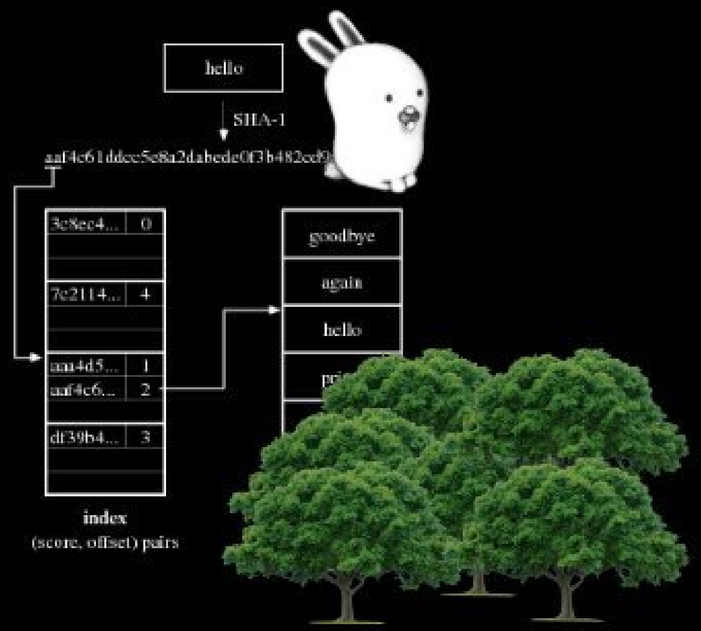

IPLD: Enter the Merkle Forest

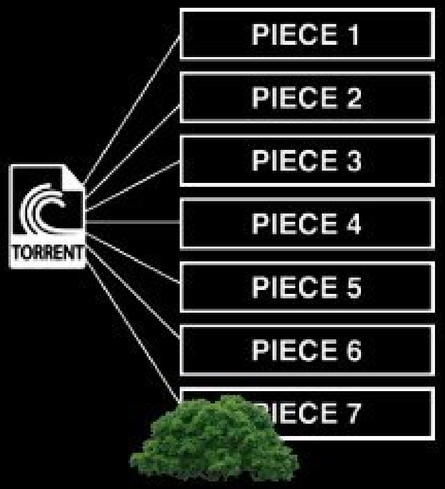


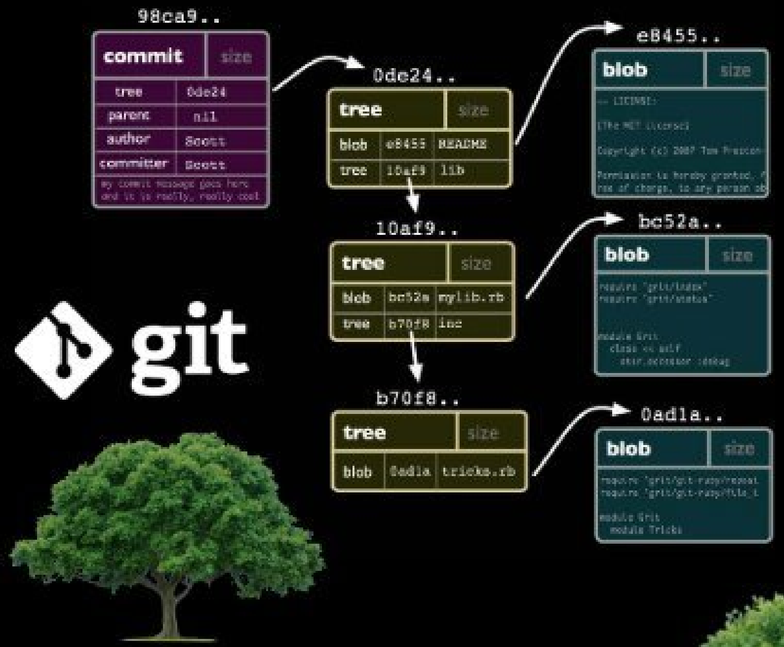
# IPLD

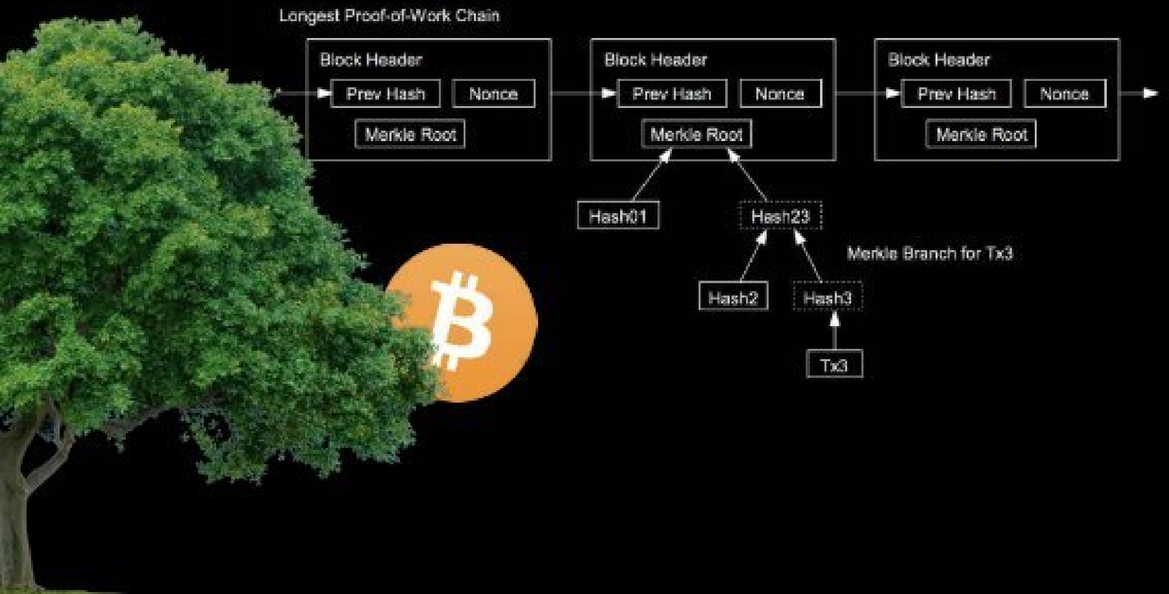
a common hash-chain format
for distributed data structures

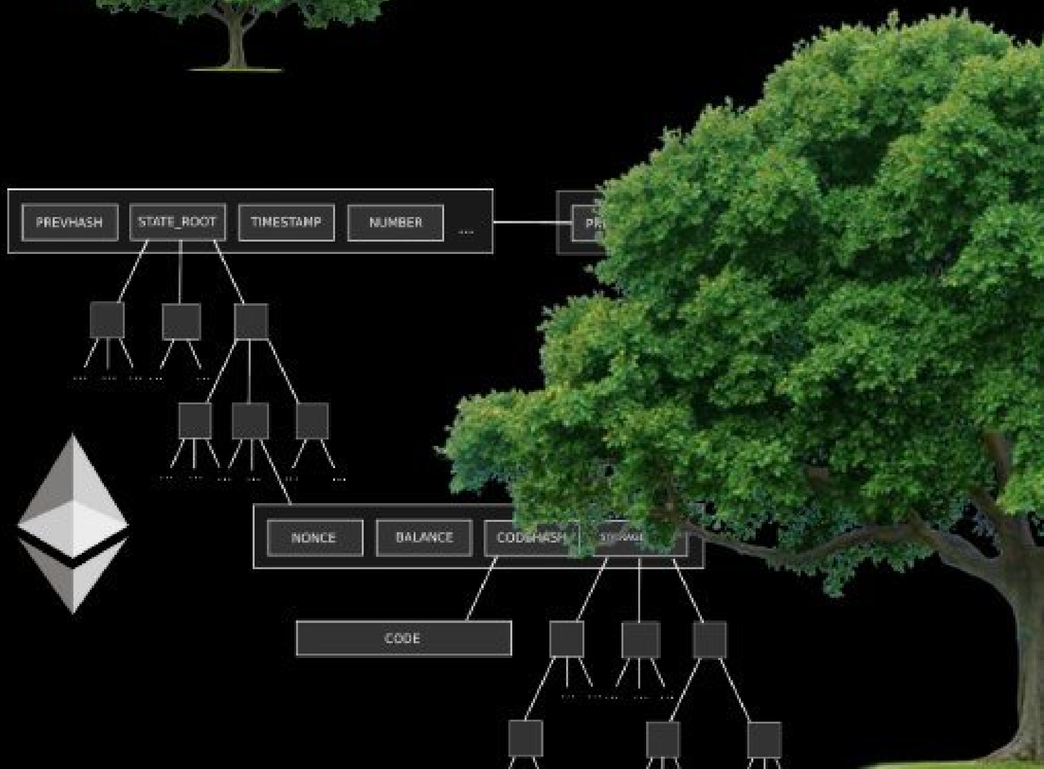


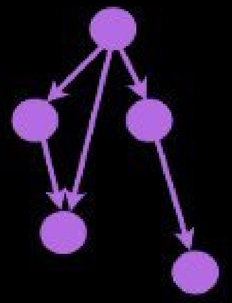

hello
SHA-1
aaf4c61ddcc5e8a2dabede0f3b482cd9
3c8ec4... 0
7c2114... 4
aaa4d5... 1
aaf4c6... 2
df39b4... 3
index
(score, offset) pairs
goodbye
again
hello


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


98ca9..
commit
size
tree 0dc24
parent nil
author Scott
committer Scott
0dc24..
tree
size
blob e8455 README
tree 10af9 lib
e8455..
blob
size
10af9..
tree
size
blob bc52a mylib.rb
tree b70f8 inc
bc52a..
blob
size
b70f8..
tree
size
blob 0ad1a tricks.rb
0ad1a..
blob
size
git


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

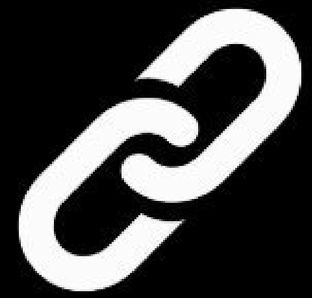

PREVHASH
STATE_ROOT
TIMESTAMP
NUMBER
...
NONCE
BALANCE
CODEHASH
STORAGE
CODE



# IPLD

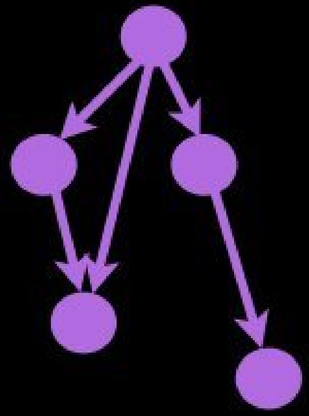
## how it works

# IPLD



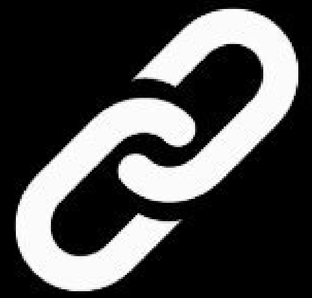
- **merkle-links** secure, immutable
- **merkle-paths** /ipld/Qmabc…xyz/foo/bar.jpg

- **universal** nestable URIs
- **serialization** JSON, PB, XML, RLP



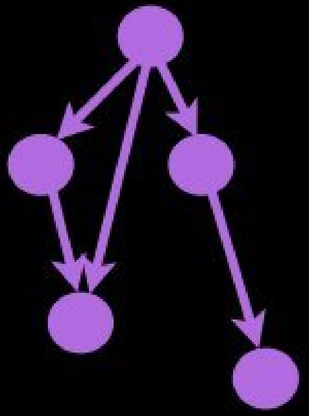
- **canonical** hashing safe

# IPLD components

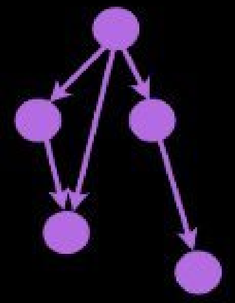


- **CID** a format for hash-links
- **Data Model** for universal resolution

- **Serialization Formats** per-data struct support
- **Tools & Libraries** to work with IPLD



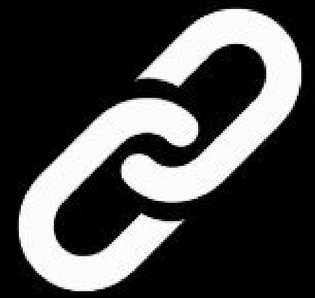
- **IPLD Selector** for selecting subgraphs
- **IPLD Transformations** for computing



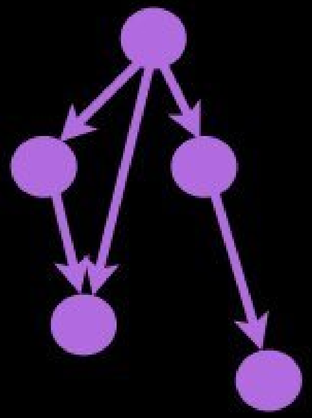
# IPLD

## how: CIDs

# Background: Multihash



- Allows you to use multiple hash functions



# multiformats - self describing values

protocol agility, interop, avoid lock in

multihash - cryptographic hashes
multiaddr
multibase
multicodec
multistream
multikey

0x08e11fc41466fcda0af7dee0905605d9
e4aada4961542da952c8bb93080cc6f9

0x95a1b32bd70332e24f63f3802aae5f5e
1fa4622cc72750e0073bbbb6dcf6fce7

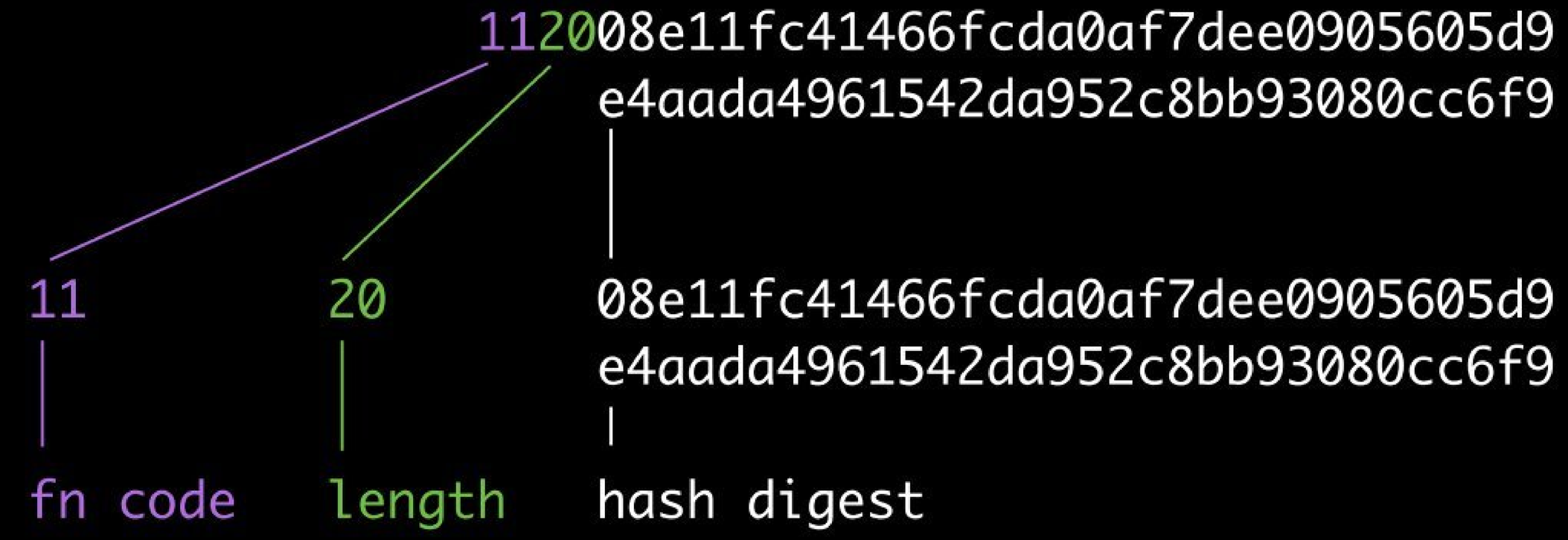
0xcaadb37a46daeda4e0d5e61574a9aaca
211d513806a026e6cc4461f7ba7867f9

0x08fbea061a5dea457d69fe5c12575c1d
9d30c49f575936f6e1c6d4ea0ab078df

```
256         0x08e11fc41466fcda0af7dee0905605d9
              e4aada4961542da952c8bb93080cc6f9

256         0x95a1b32bd70332e24f63f3802aae5f5e
              1fa4622cc72750e0073bbbb6dcf6fce7

256         0xcaadb37a46daeda4e0d5e61574a9aaca
              211d513806a026e6cc4461f7ba7867f9

256         0x08fbea061a5dea457d69fe5c12575c1d
              9d30c49f575936f6e1c6d4ea0ab078df
```

```
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
```


1120 08e11fc41466fcda0af7dee0905605d9
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

multiformats / multihash

Unwatch 23 Unstar 133 Fork 30

Code Issues 12 Pull requests 5 Projects 0 Pulse Graphs Settings

Branch: master multihash / hashtable.csv Find file Copy path

ivilata replace ``sha3`` name with ``sha3-512``, add other SHA-3 functions (#11) 45ac2d8 on Jan 19, 2016

2 contributors

13 lines (12 sloc) 172 Bytes Raw Blame History

Search this file...

| | code | name |
|---|---|---|
| 1 | 0x11 | sha1 |
| 2 | 0x12 | sha2-256 |
| 3 | 0x13 | sha2-512 |
| 4 | 0x14 | sha3-512 |
| 5 | 0x15 | sha3-384 |
| 6 | 0x16 | sha3-256 |
| 7 | 0x17 | sha3-224 |
| 8 | 0x18 | shake-128 |
| 9 | 0x19 | shake-256 |
| 10 | 0x40 | blake2b |
| 11 | 0x41 | blake2s |

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

## Multibase Table v1.0.0-RC (semver)

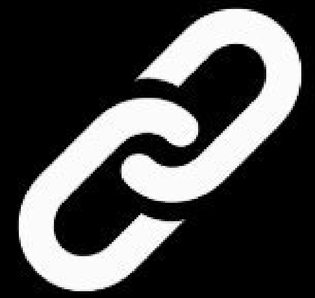
The current multibase table is here:

```
encoding      codes   name
base1         1       unary tends to be 11111
base2         0       binary has 1 and 0
base8         7       highest char in octal
base10        9       highest char in decimal
base16        F, f    highest char in hex
base32        B, b    rfc4648 - no padding - highest letter
base32pad     C, c    rfc4648 - with padding
base32hex     V, v    rfc4648 - no padding - highest char
base32hexpad  T, t    rfc4648 - with padding
base32z       h       z-base-32 - used by Tahoe-LAFS - highest letter
base58flickr  Z       highest char
base58btc     z       highest char
base64        m       rfc4648 - no padding
base64pad     M       rfc4648 - with padding - MIME encoding
base64url     u       rfc4648 - no padding
base64urlpad  U       rfc4648 - with padding
```

These encodings are being considered:

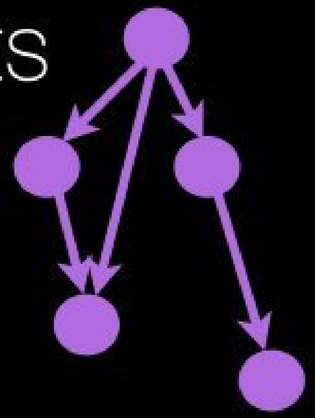
```
base128
base256ascii  X       ascii
base-emoji    😎      base emoji
base65536     🆒      base65536
utf8
utf16
```

# CID: Content IDentifier



- **CID** is a format for hash-links (merkle-links)
- Uses **Multihash** for multiple hash fn support

- Uses **Multibase** for multiple encodings
- Uses **Multicodec** for multiple serialization formats

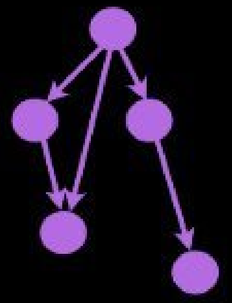


# CID format

```
<cidv1>   ::= <mbp><version><mcp><mh>

<mb>      ::= <multibase-prefix>
<version> ::= <cid-version>
<mcp>     ::= <multicodec-packed-code>
<mh>      ::= <multihash>
```

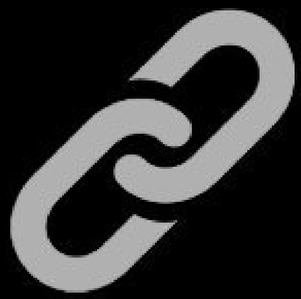
# CID format

```
<cidv1>   ::= <mbp><version><mcp><mh>

<mb>      ::= <multibase-prefix>
<version> ::= <cid-version>
<mcp>     ::= <multicodec-packed-code>
<mh>      ::= <multihash>
```



# IPLD

## code example



```
> 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
```

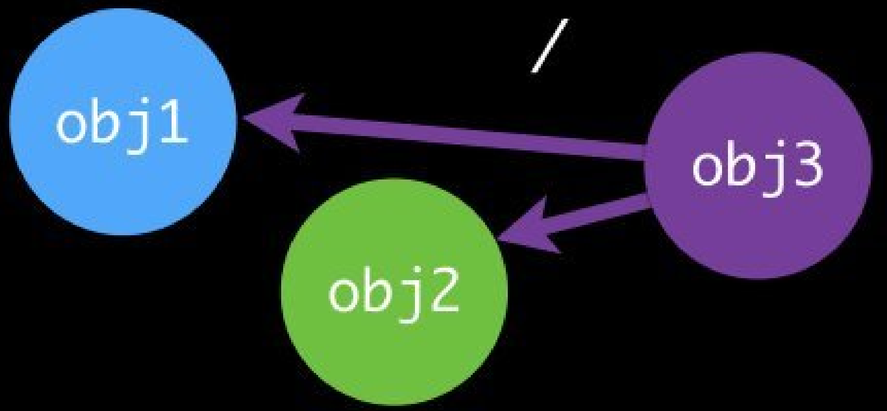
```
> var obj3 = {
  "files": [
    { "/": "QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg" },
    { "/": "QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV" },
  ]
}
> var obj3Data = ipld.marshal(obj3)
> var obj3Hash = ipld.multihash(obj3Data)

> obj3Hash
QmdhMzs1tkLYwC3jimzUABEt1xzkrokkanywe1y1QFcAhw
```

___ HASH LINKS!

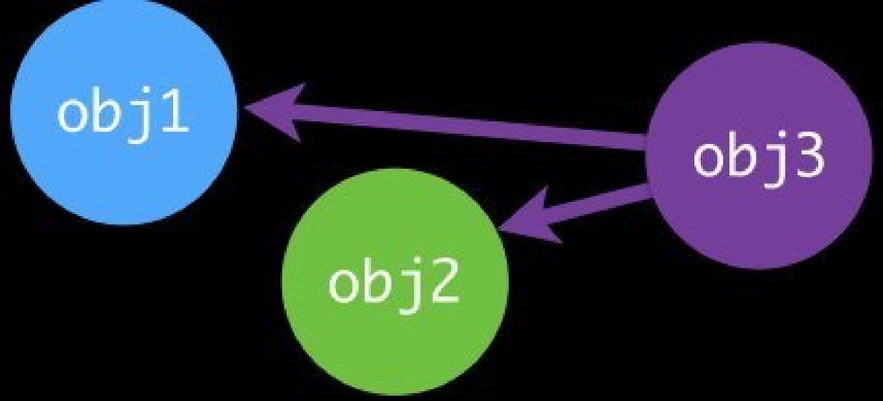
```
> var obj3 = {
  "files": [
    { "/": "QmUUuaDDWvRG23zyzBQVv43etRqmbGCRNhgZYu9qvZ88Bg" },
    { "/": "QmSVuc2kjbtCFQ9ur8fnyKUKvSyLZMTBVbZugJWChydAHV" },
  ]
}
> var obj3Data = ipld.marshal(obj3)
> var obj3Hash = ipld.multihash(obj3Data)
```

___ HASH LINKS!

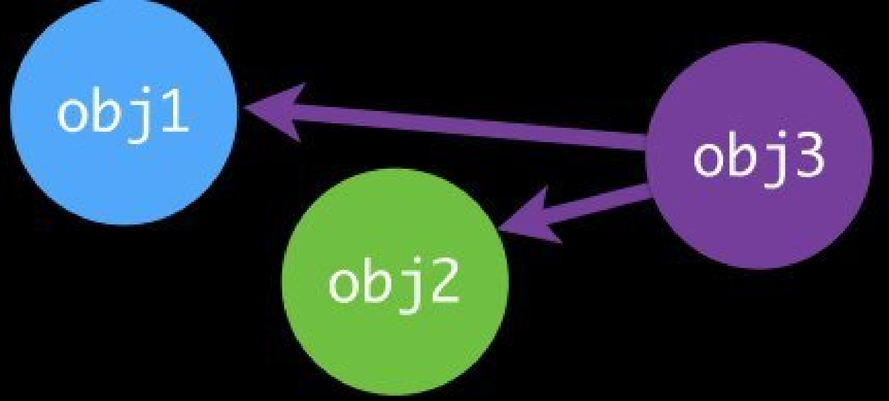
HASH LINKS!
/



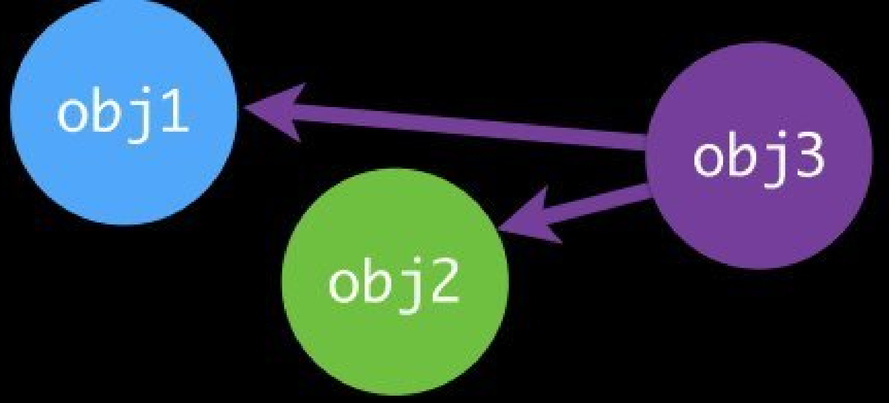
```
> var ipfs = require('ipfs')

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



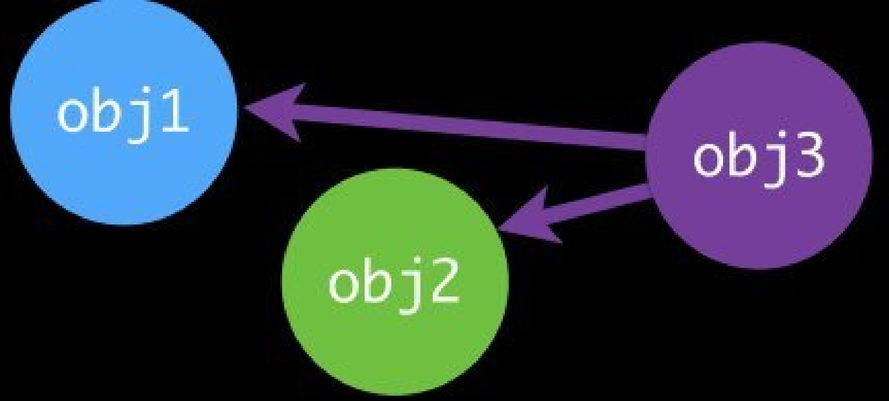
```
> var ipfs = require('ipfs')

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

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

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



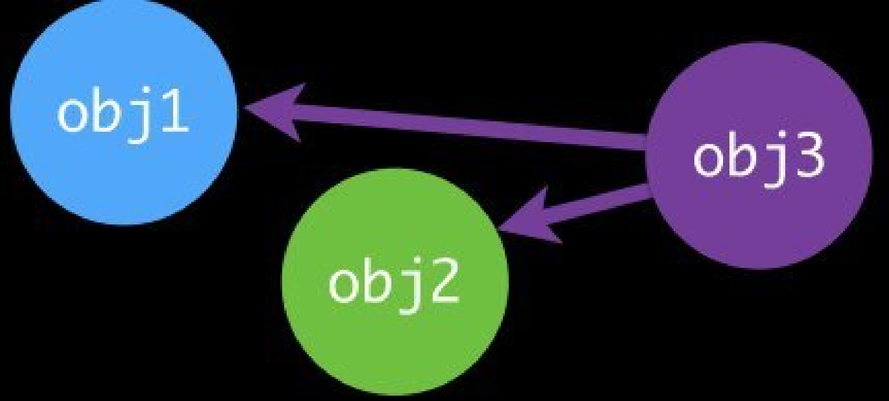
```
> var ipfs = require('ipfs')

> 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 "
```



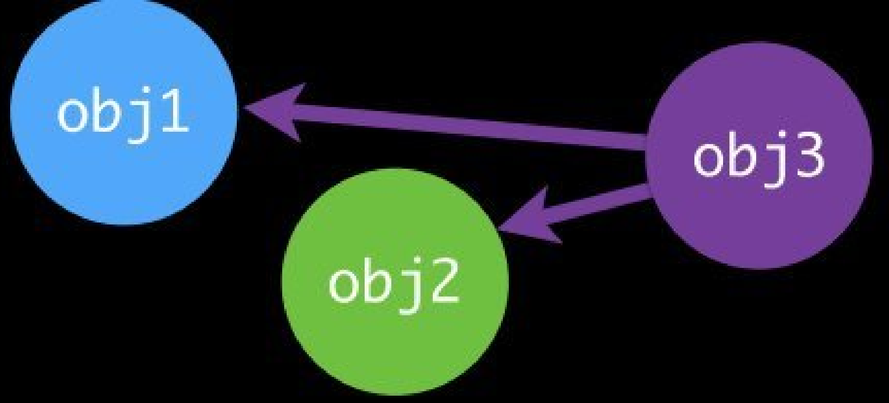
```
> var ipfs = require('ipfs')

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

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



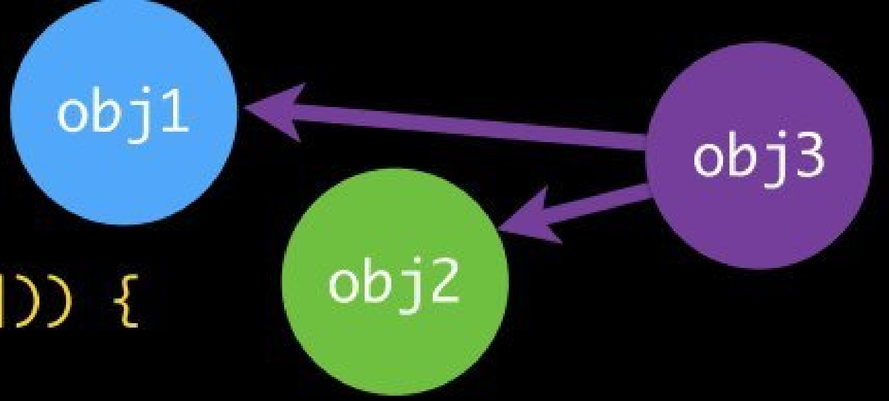
```
> var ipfs = require('ipfs')

> 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 "
```

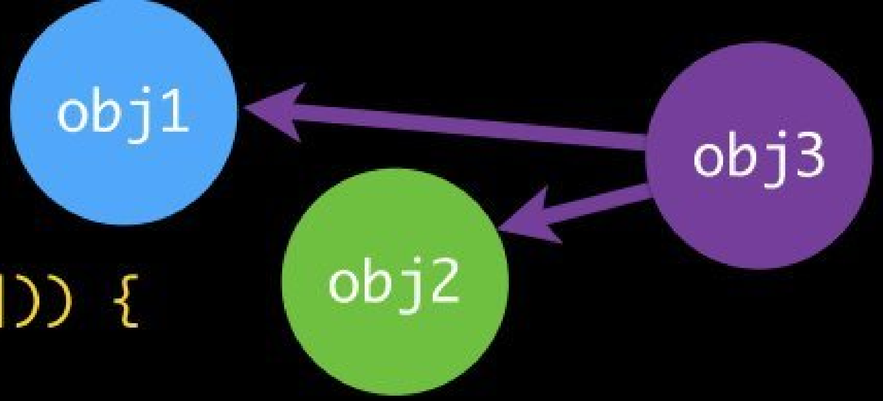


```
> function catFile(link) {
  var obj = ipfs.resolve(link)
  var out = obj.data || ""
  for (var file of (obj.files || [])) {
    out += catFile(file)
  }
  return out
}
```

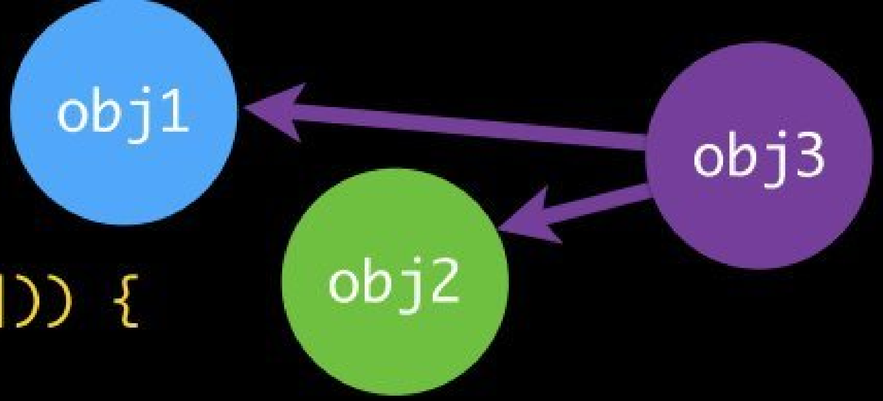



```
{
  "data": "<content>",
  "files": [
    <subfile-link>,
    <subfile-link>,
    <subfile-link>,
    ...
  ]
}
```

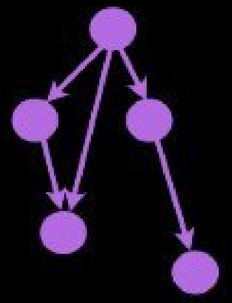
```
> function catFile(link) {
  var obj = ipfs.resolve(link)
  var out = obj.data || ""
  for (var file of (obj.files || [])) {
    out += catFile(file)
  }
  return out
}


> 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"
```



# IPLD

## more examples

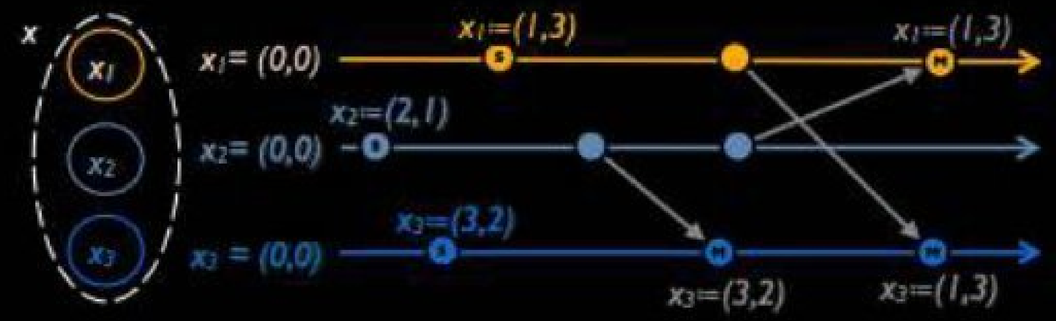



Figure 7: Integer LWW Register (state-based). Payload is a pair (value, timestamp)

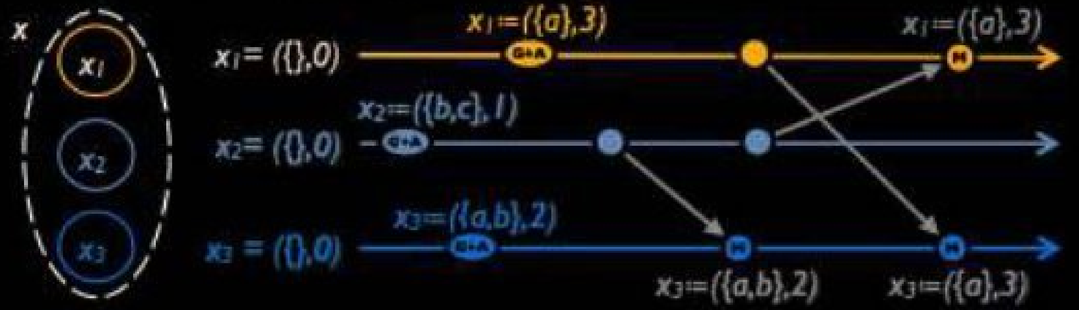



Figure 8: LWW-Set (state-based). Payload is a pair (set, timestamp)

# CRDTs

# Orbit

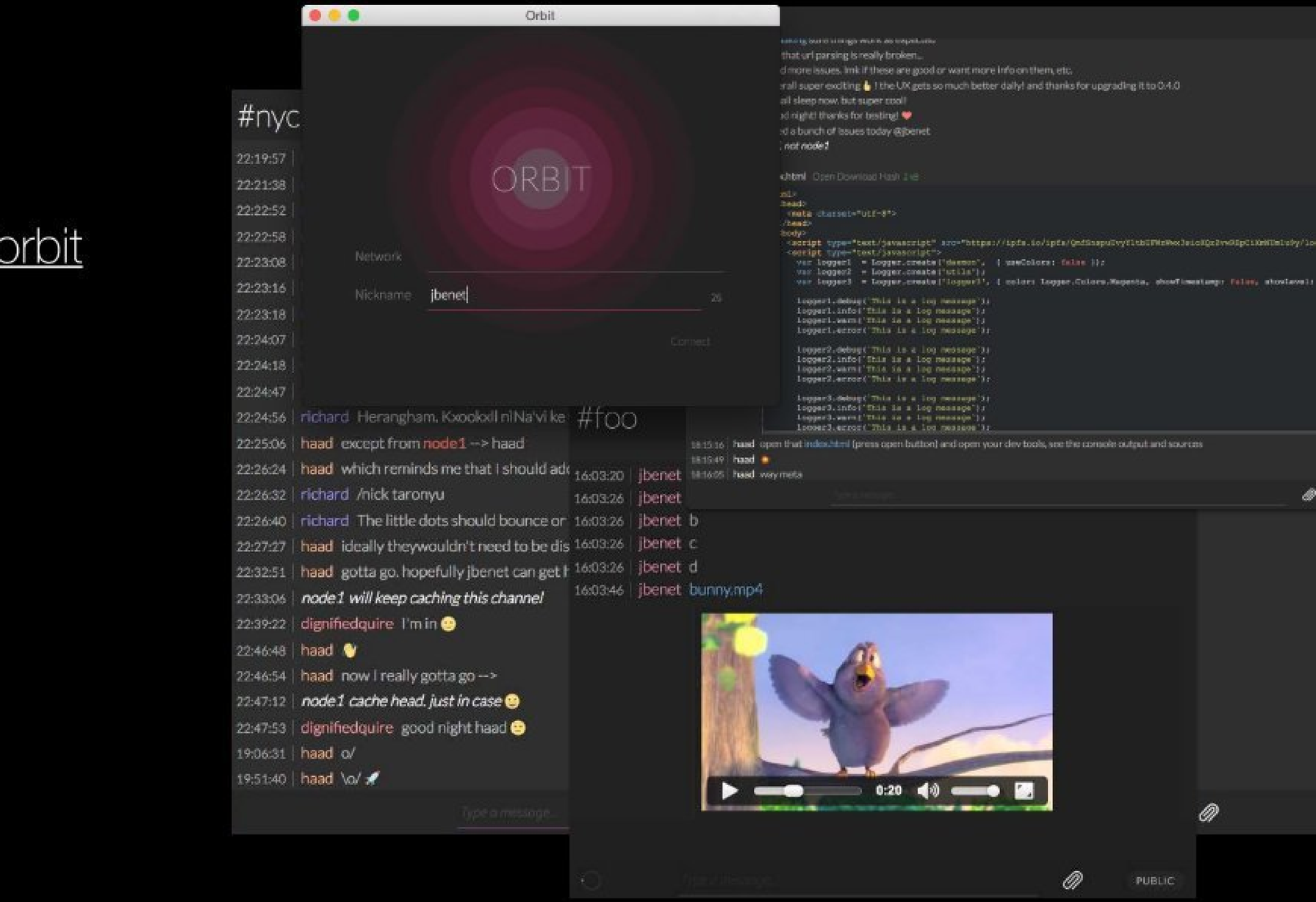
github.com/haadcode/orbit

p2p chat on IPFS

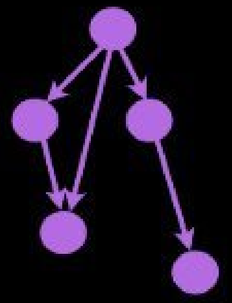
IPFS

IPLD

libp2p




IPLD: Enter the Merkle Forest



# IPLD

## even more examples

# IPLD

- The “thin-waist” of the internet of data
- Merkle Links
- **IPLD**: common format for hash chains
- **IPRS**: Distributed Records on IPFS
- Git Versioning with **Commits**
- **unixfs**: representing POSIX files

IPFS

# DNS

is used for all sorts of things

# DNS

is used for all sorts of things

```
type:   A
sub:    subdomain
value:  104.236.176.52
TTL:    180s
```

# IPRS

is used for all
sorts of things

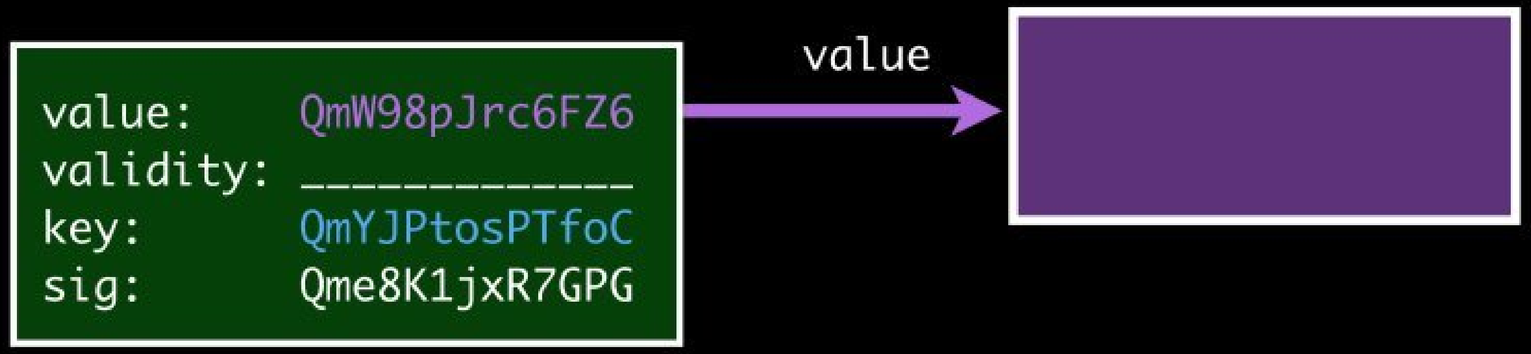
```
value:    QmW98pJrc6FZ6
validity: ____________
key:      QmYJPtosPTfoC
sig:      Qme8K1jxR7GPG
```

## IPRS

is used for all
sorts of things

## Records

are just IPLD objects
that link to other objects

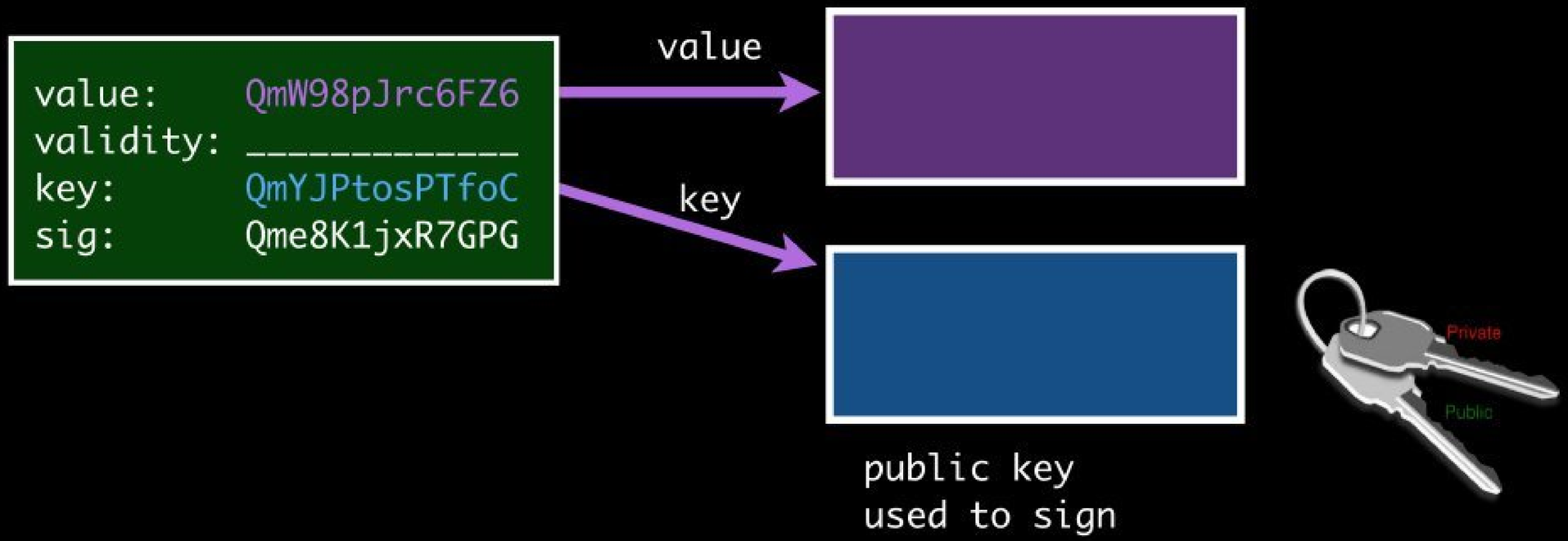


## IPRS

is used for all
sorts of things

## Records

are just IPLD objects
that link to other objects



## IPRS

is used for all
sorts of things

```
value:     QmW98pJrc6FZ6
validity:  ____________
key:       QmYJPtosPTfoC
sig:       Qme8K1jxR7GPG
```

## Records

are just IPLD objects
that link to other objects

## IPRS Validity

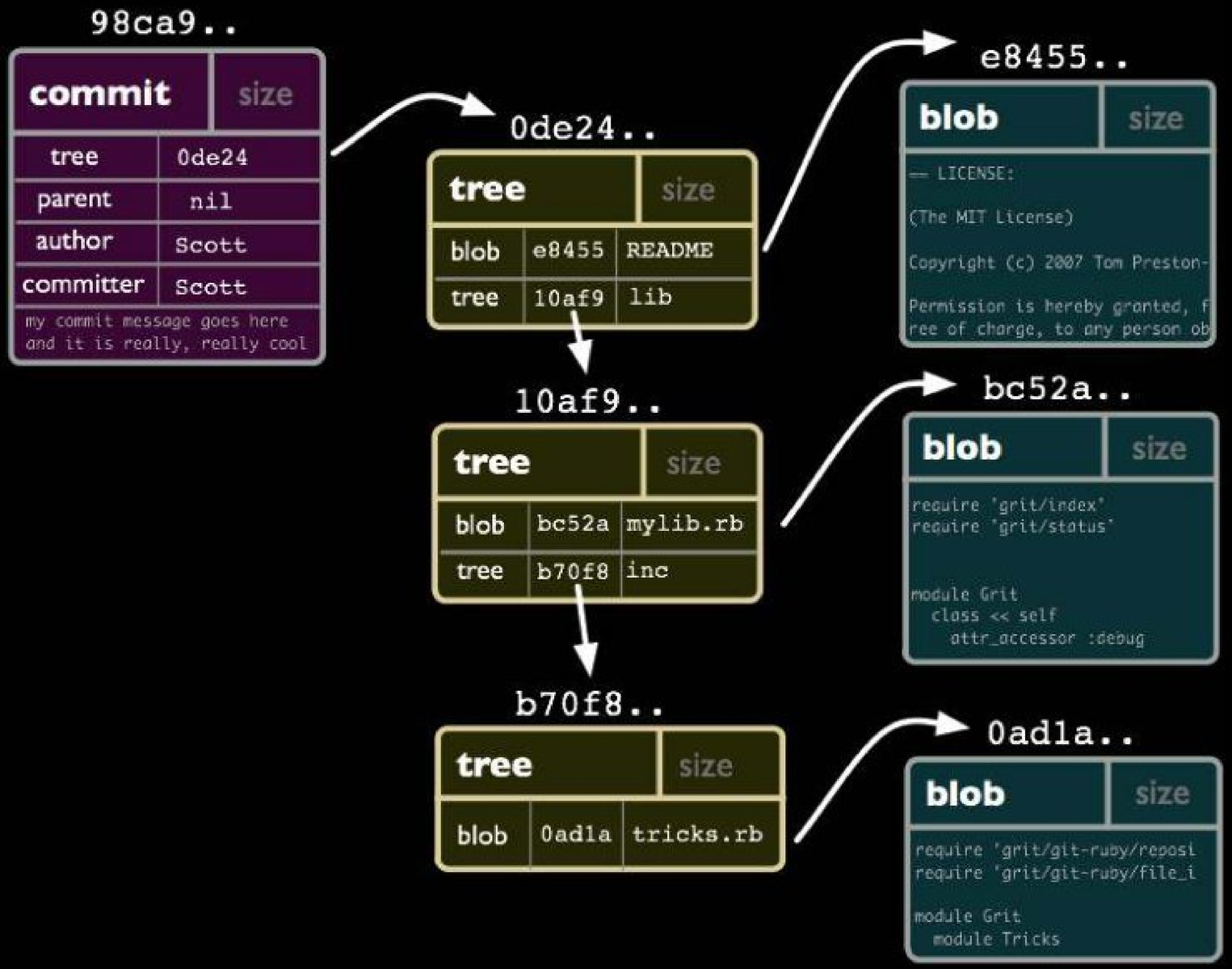
- App-specific value
- Timestamps + TTL
- Cryptographic Freshness
- Ancestry (hash chain)

# IPLD

- The “thin-waist” of the internet of data
- Merkle Links
- **IPLD**: common format for hash chains
- **IPRS**: Distributed Records on IPFS
- Git Versioning with **Commits**
- **unixfs**: representing POSIX files

IPFS

# versioning data structures - git



# versioning data structures - git


```
jbenet @ earth : ~ > ipld cat --fmt yml $gitcommit
---
tree: {mlink: e4647147e940e2fab134e7f3d8a40c2022cb36f3}
parents:
  - {mlink: /ipfs/QmP8AmEKHHMnaBaQsEJVGs7ZbBSQAxhPn9zP2jHQy6Uekj}
  - {mlink: /ipfs/QmfVQwj2cyhrKQV1BY1y2nAcQcjiKEkvEYJDGwWFUHcHaQ}
author:
  mlink: /ipfs/QmVoXUXmES4ujTdJ9CQAs8D3jMag2enjtBn2towYHmKxYa
  name: Juan Batiz-Benet
  email: juan@benet.ai
  time: "1435398707 -0700"
committer:
  mlink: /ipfs/QmVoXUXmES4ujTdJ9CQAs8D3jMag2enjtBn2towYHmKxYa
  name: Juan Batiz-Benet
  email: juan@benet.ai
  time: "1435398707 -0700"
message: "Merge pull request #7 from ipld\n\n(WIP) records + merkledag specs"
```

# CRDTs: lattices as IPFS objects

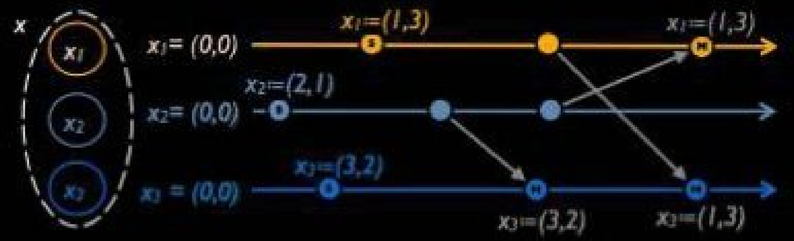



Figure 7: Integer LWW Register (state-based). Payload is a pair (value, timestamp)

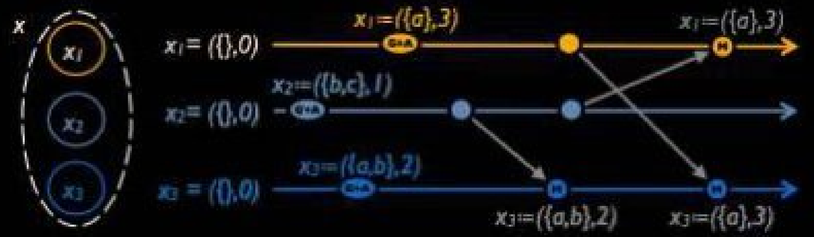
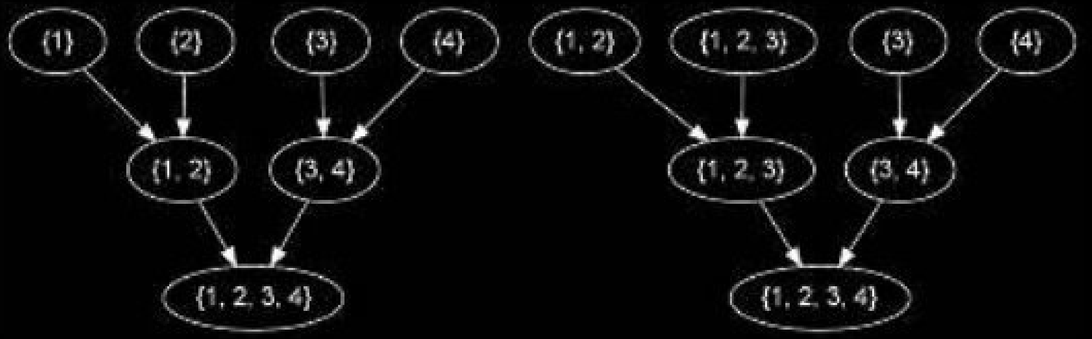



Figure 8: LWW-Set (state-based). Payload is a pair (set, timestamp)

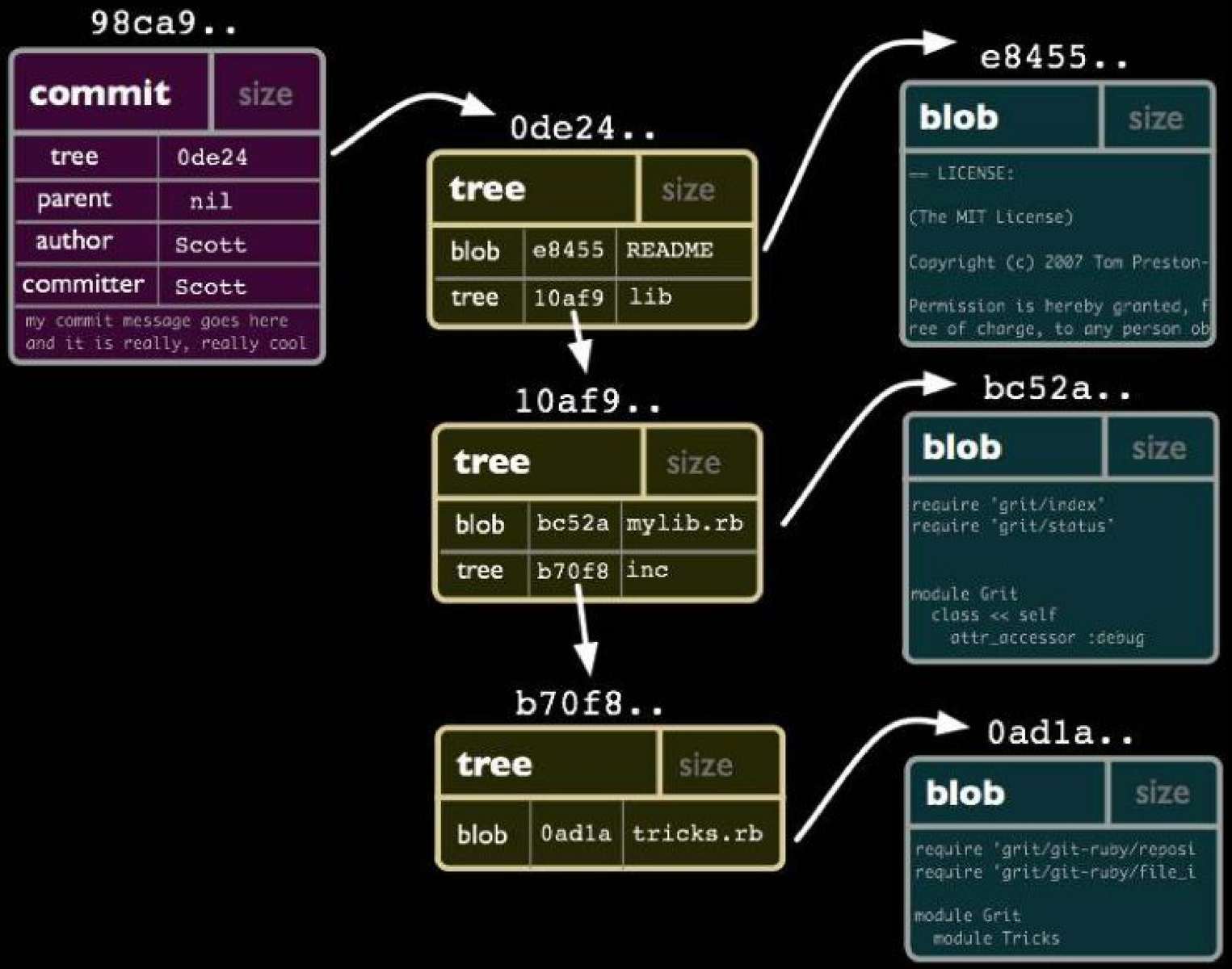


# IPLD

- The “thin-waist” of the internet of data
- Merkle Links
- **IPLD**: common format for hash chains
- **IPRS**: Distributed Records on IPFS
- Git Versioning with **Commits**
- **unixfs**: representing POSIX files

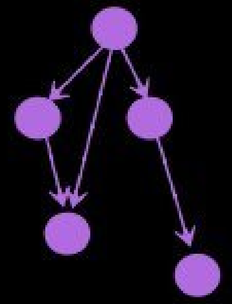
IPFS

# unixfs: representing POSIX files

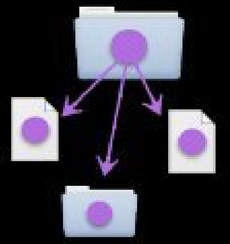


# unixfs: representing POSIX files

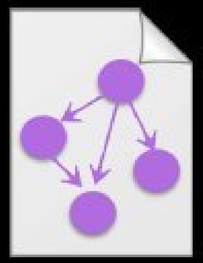
unixfs as a dag



files are dag nodes



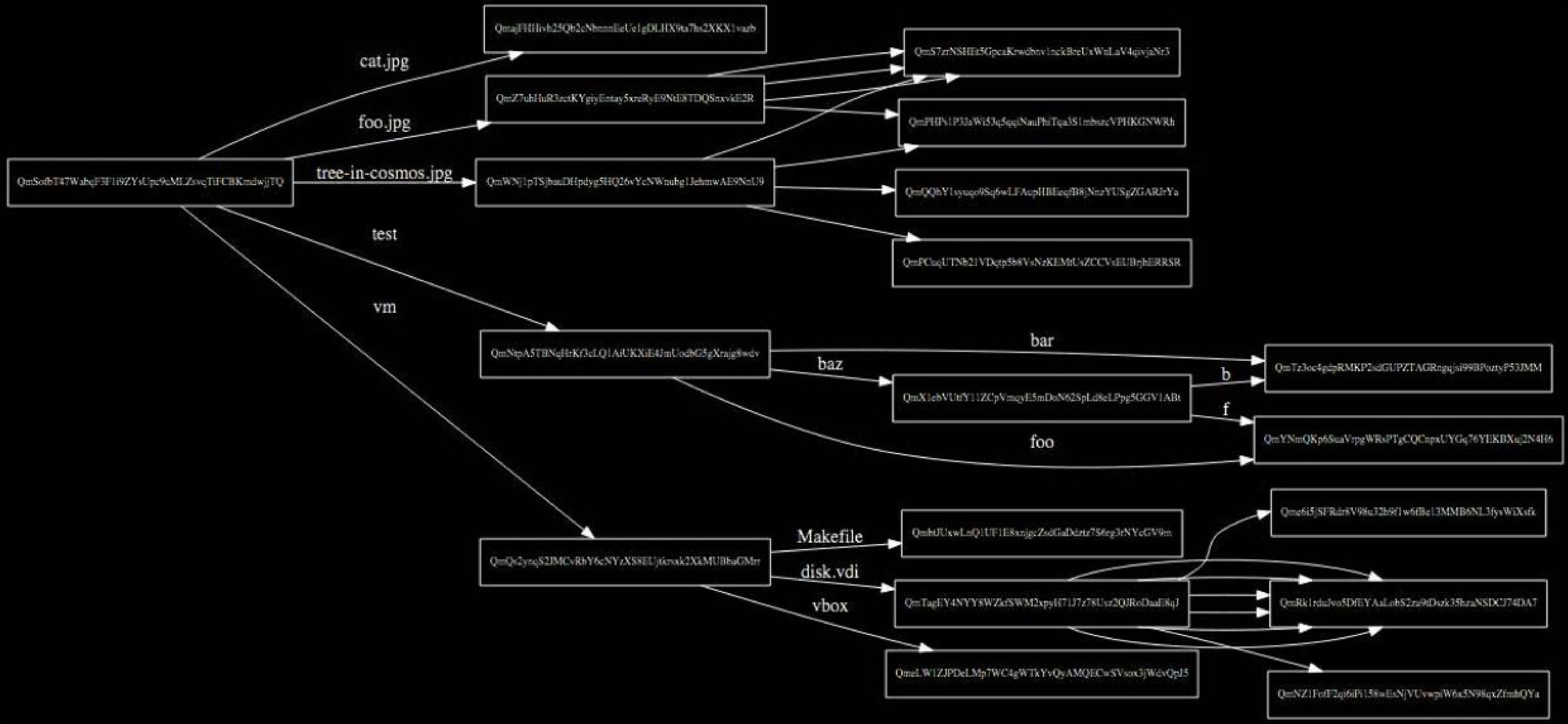
which link
to others



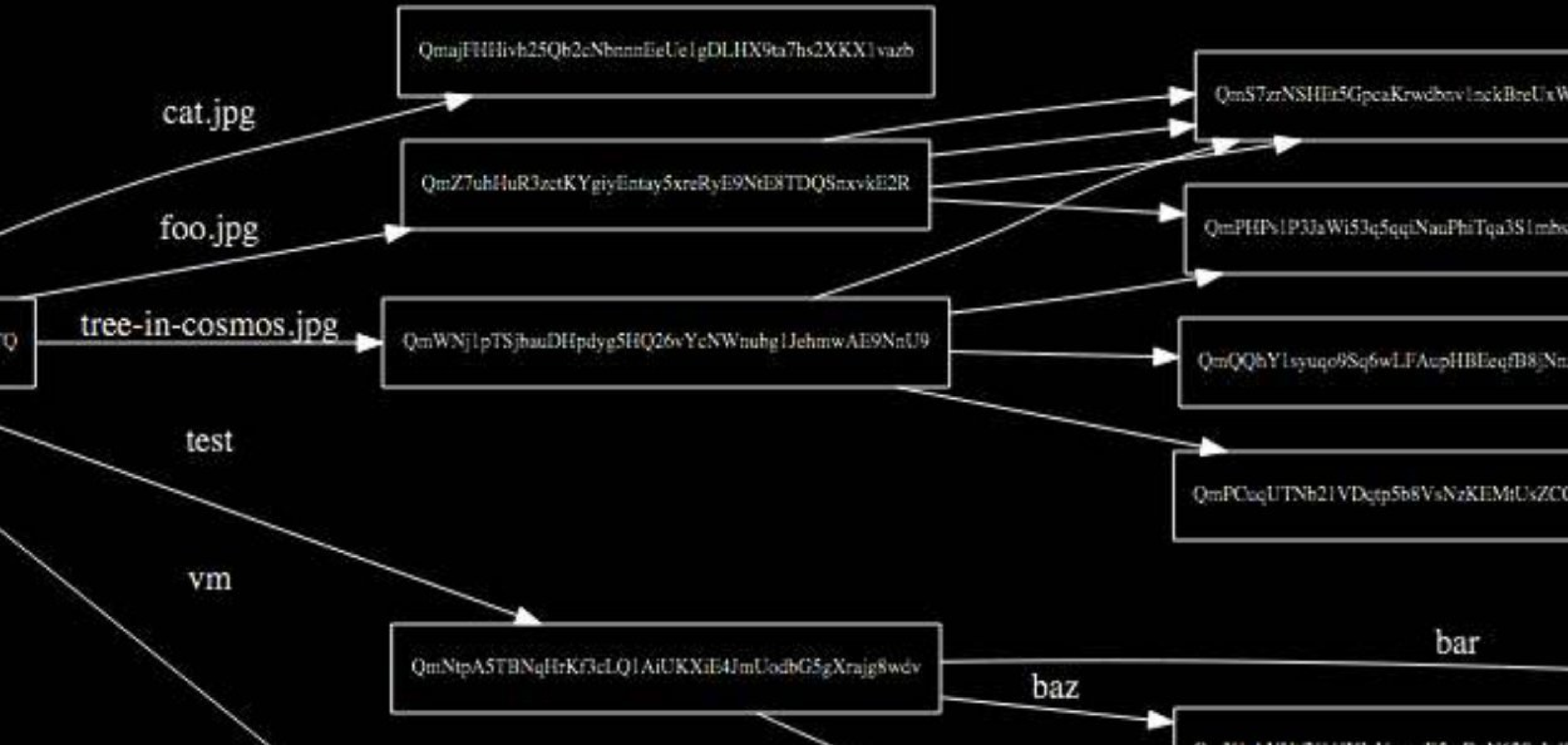
BIG files may be
split into many

directories are
also dag nodes

# unixfs: representing POSIX files



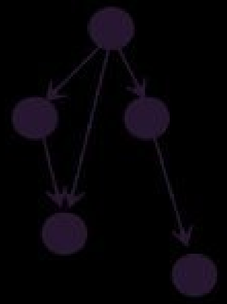
# unixfs: representing POSIX files

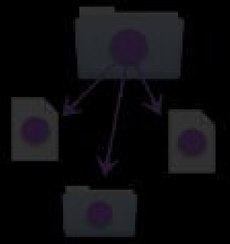


# unixfs: representing POSIX files

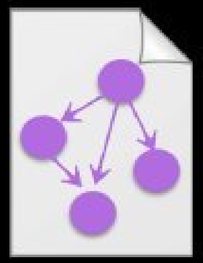
unixfs as a dag

files are dag nodes





which link to others



BIG files may be split into many

- File Importers
- Data Versioners
- Diffing Algorithms
- Computable Data
- Kolmogorov Complexity

also dag nodes