UNDERSTANDING THE INTERPLANETARY APPROACH TO NAMING

IPFS AND IPNS

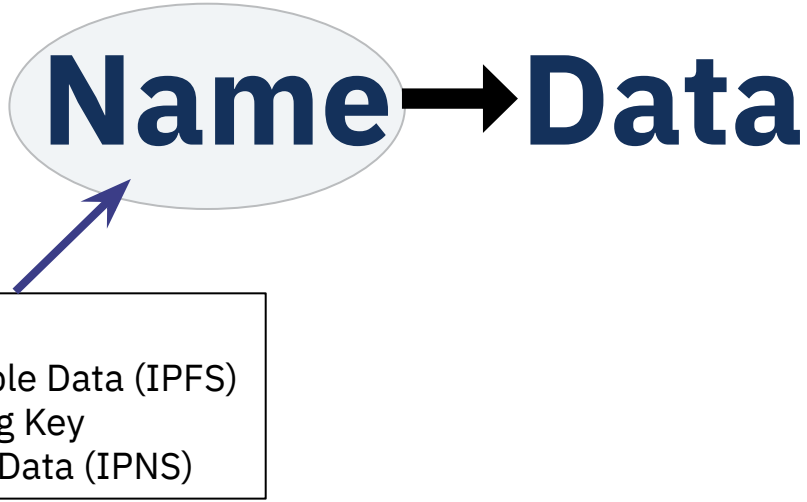
Adin Schmahmann
Go IPFS

Naming Data:

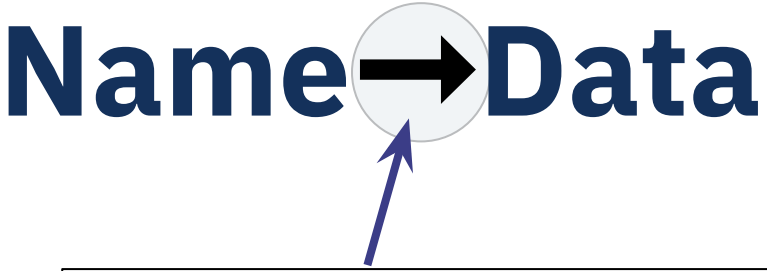
OVERVIEW

Name→Data

Naming Data:
OVERVIEW

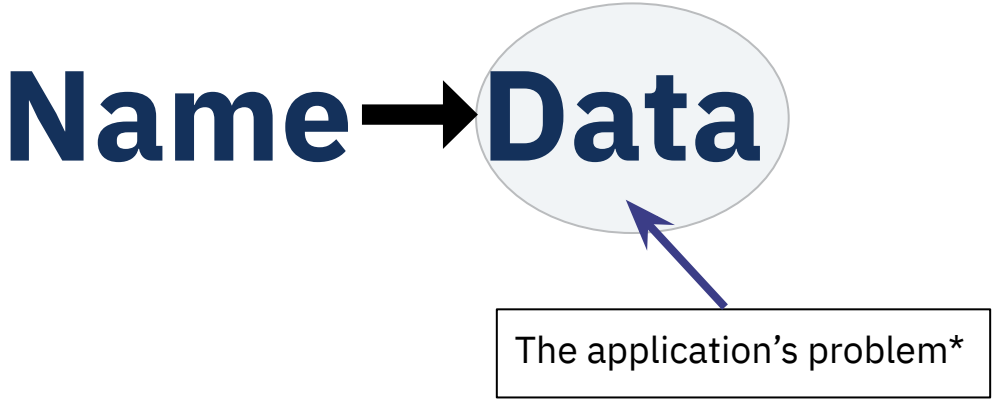


Naming Data:
OVERVIEW



Many Different Key-Value Mapping Schemes
- Database
- DHT
- DNS
- MDNS
- Persistent PubSub

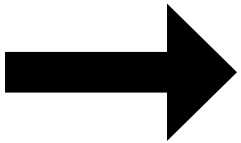
Naming Data:
OVERVIEW



Naming Data:

IPFS

Name Hash



Kademlia
Bitswap

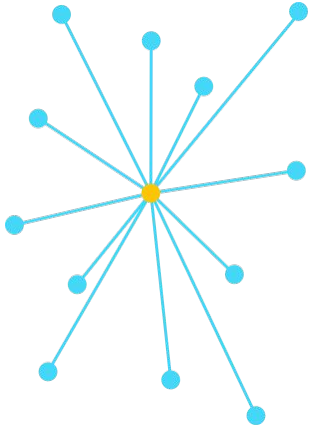
Data IPLD

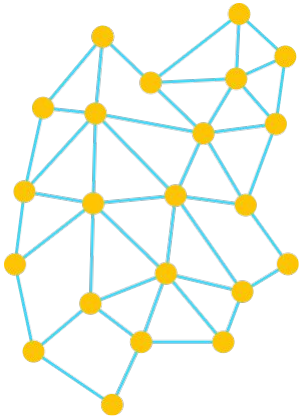
IPFS:

WHY IMMUTABILITY?

Why Immutability?:

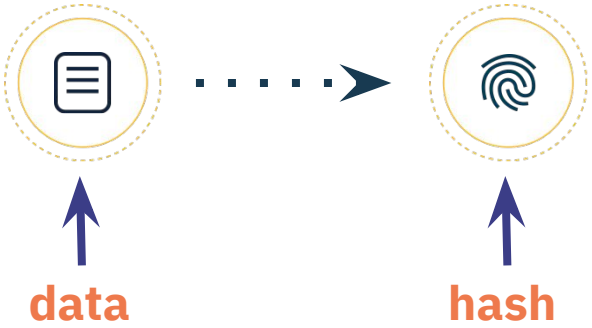
NO TRUST





Why Immutability?:

INTEGRITY CHECKING

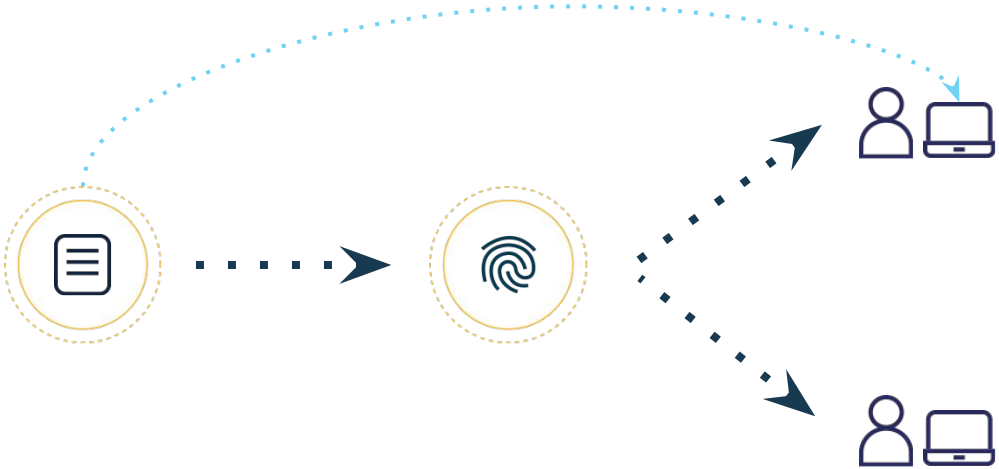


Why Immutability?:

INTEGRITY CHECKING

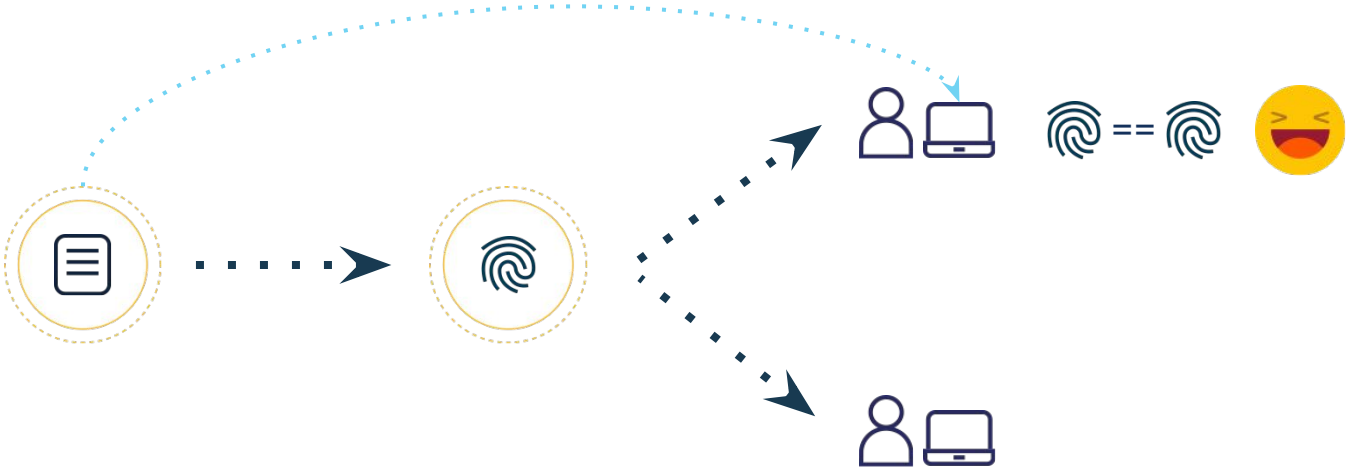
Why Immutability?:

INTEGRITY CHECKING



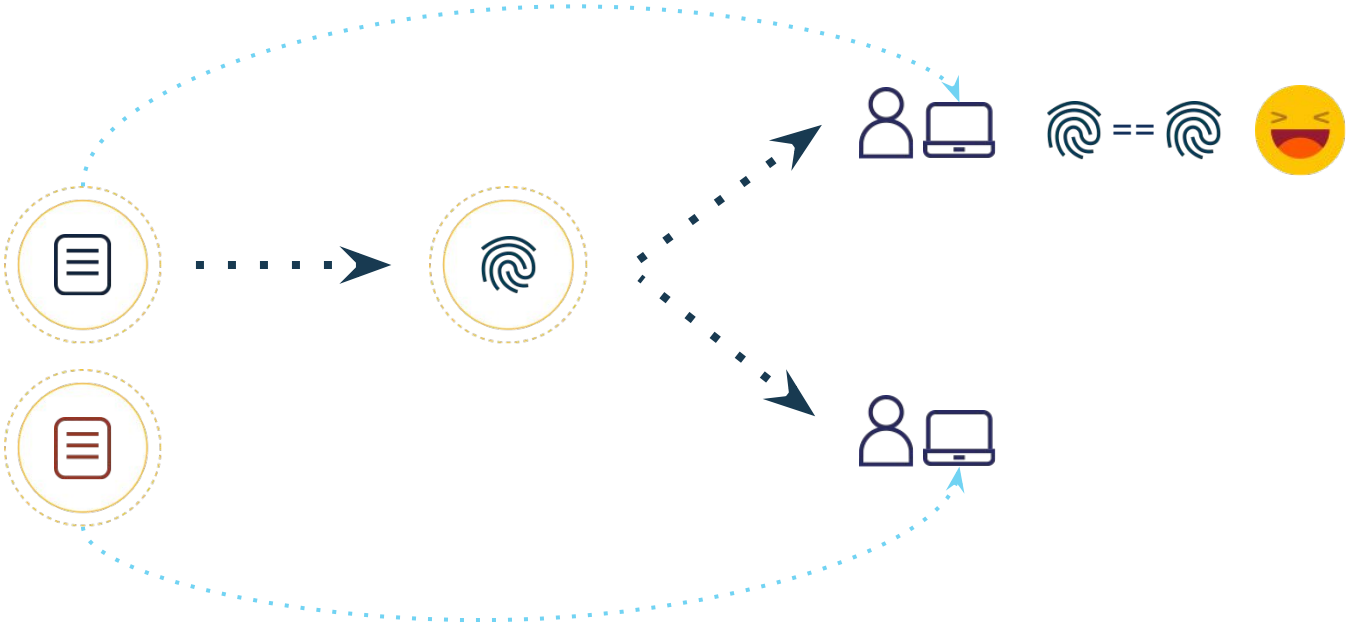
Why Immutability?:

INTEGRITY CHECKING



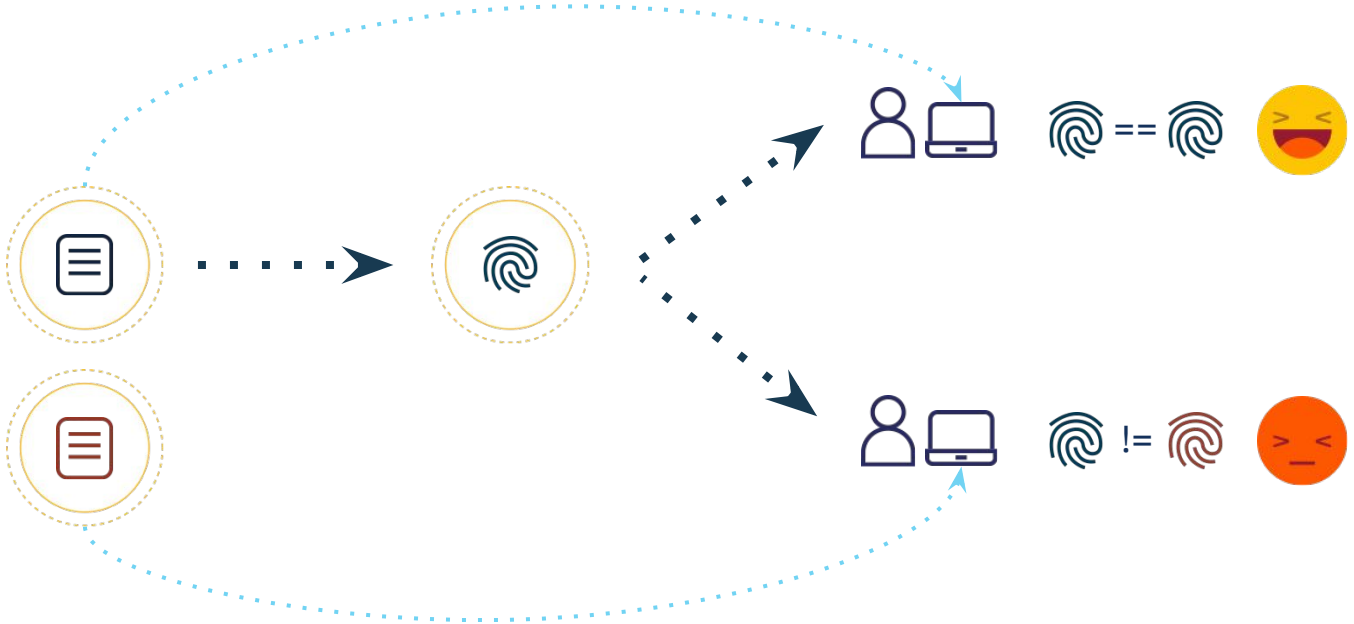
Why Immutability?:

INTEGRITY CHECKING



Why Immutability?:

INTEGRITY CHECKING



Why Immutability?:

IMMUTABLE CONTENT 💎

1. Verifiability

Why Immutability?:

VERIFIABILITY

abc.com/poodle.jpg

24h

. . . . . ➤

later

abc.com/poodle.jpg

Why Immutability?:

VERIFIABILITY

Location Addressing

abc.com/poodle.jpg

VS

Content Addressing

Why Immutability?:

FETCH FROM ANYONE

abc.com/poodle.jpg

xyz.com/poodle.jpg

Why Immutability?:

IMMUTABLE CONTENT 💎

1. Verifiability
2. Caching & Deduping

Why Immutability?:

CACHING & DEDUPING

24h

later

Why Immutability?:

CACHING & DEDUPING

VS

Why Immutability:

ANATOMY OF A CID

Why Immutability?:

WHAT DOES A CID LOOK LIKE?

Binary:

<cid-version><multicodec><multihash>

String:

<base>**base**(<cid-version><multicodec><multihash>)

Demo:

CID EXPLORER
cid.ipfs.io

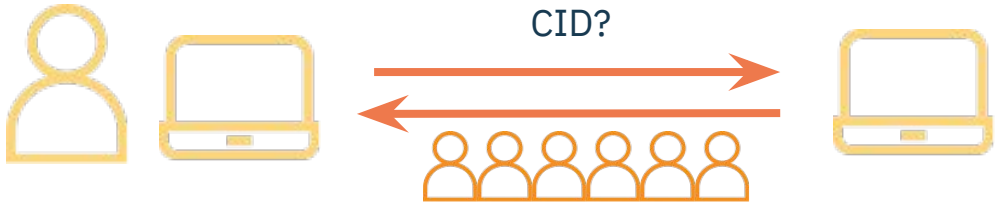
Naming Data:

FINDING IMMUTABLE DATA

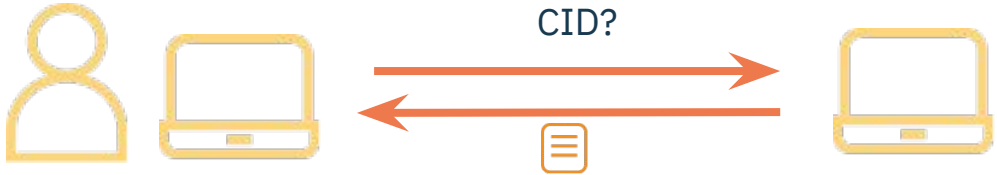
Finding Immutable Data:

Finding Providers and Data

Providers



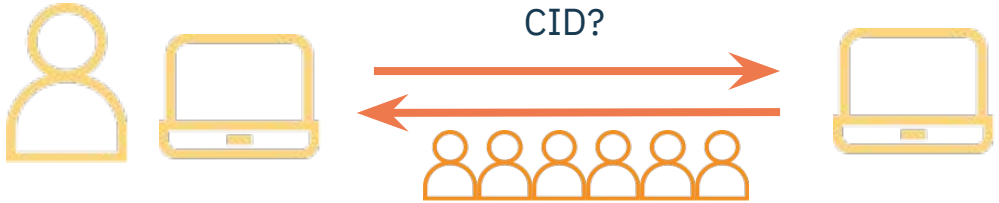
Data



Finding Immutable Data:

Finding Providers

Providers



Less pressure on shared network resources (e.g. DHT) to store  vs 

Don’t need to know in advance where the data is stored

Finding Immutable Data:

Finding Data

Data

Just use TCP or some existing transport?

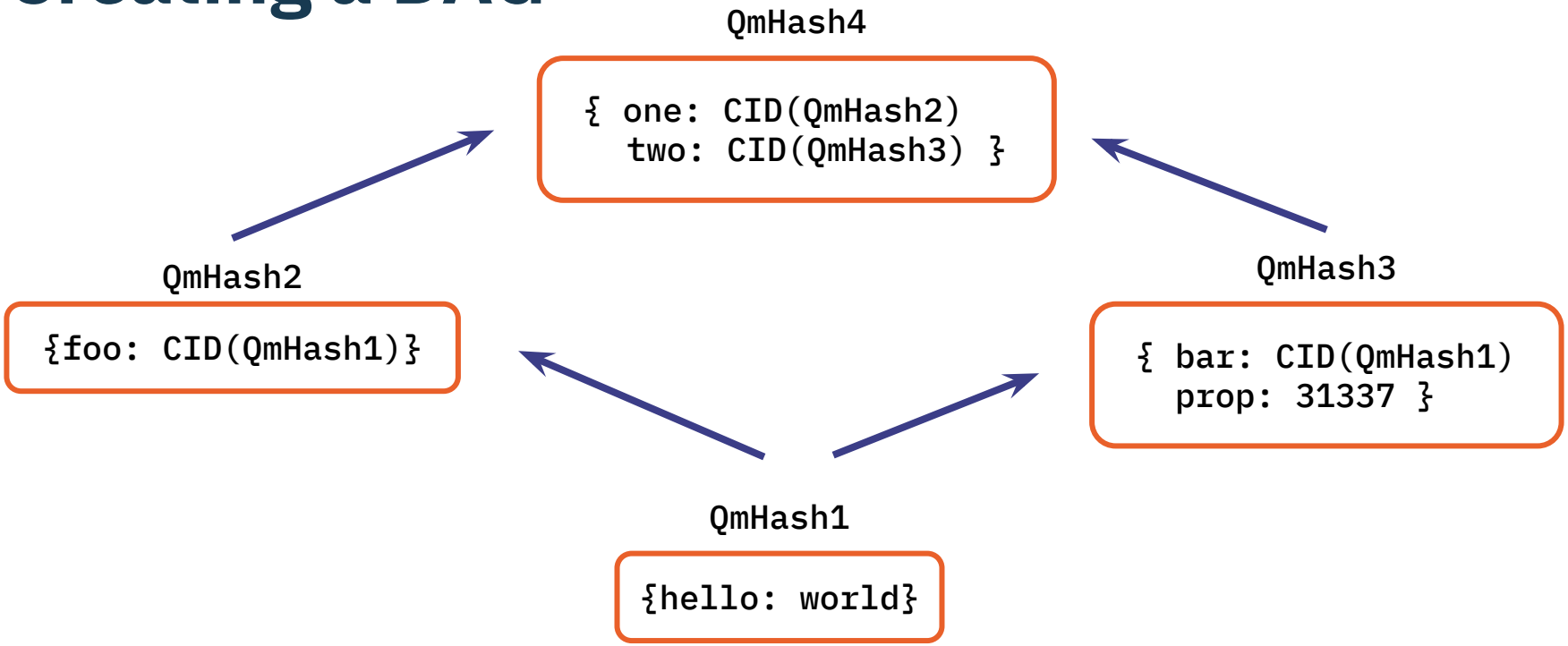
No. We want to download from multiple peers simultaneously. This means we want hashes for many sections of our data.

IPLD:

MERKLE DAG

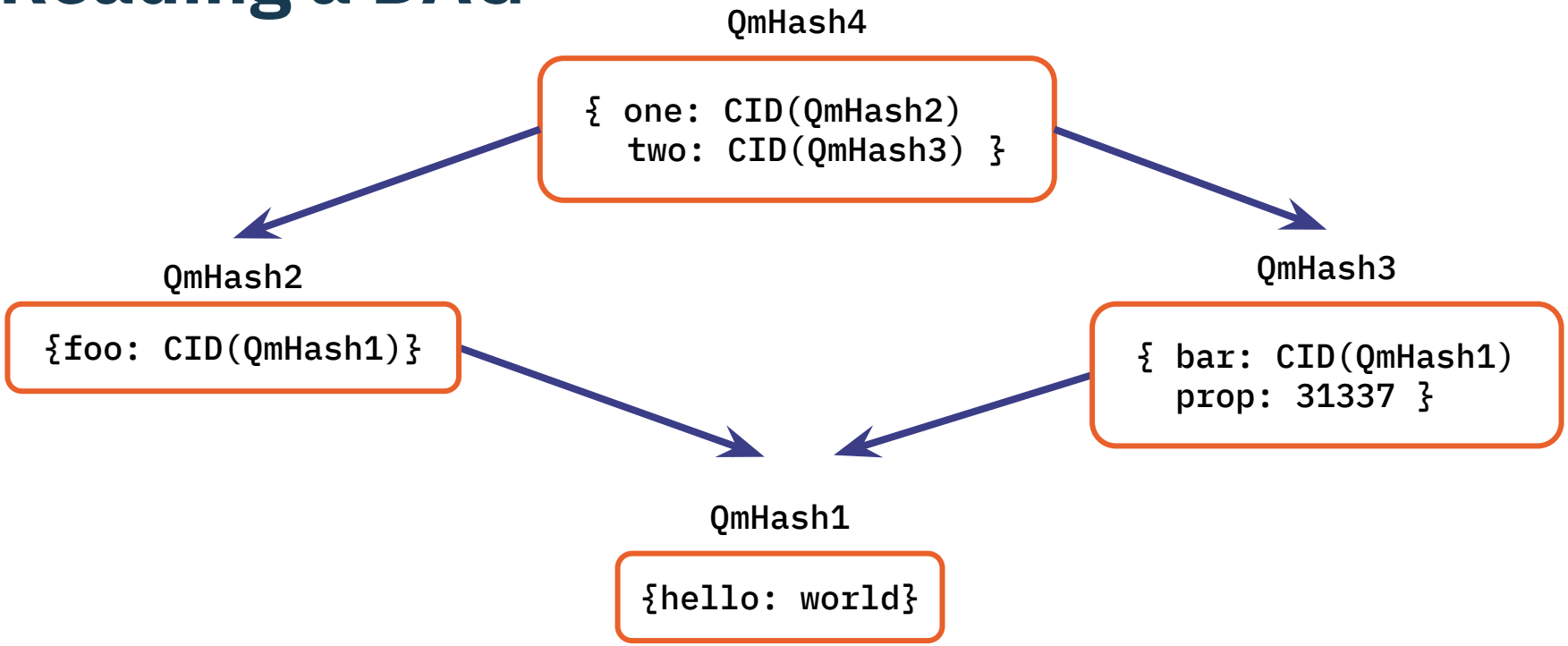
Merkle DAG:

Creating a DAG



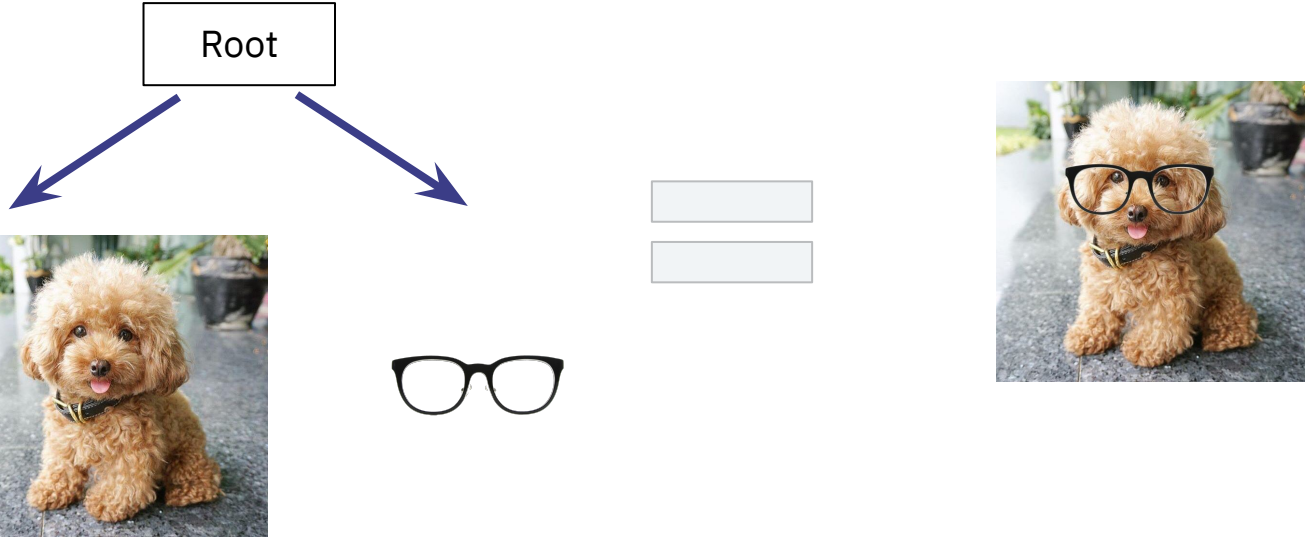
Merkle DAG:

Reading a DAG



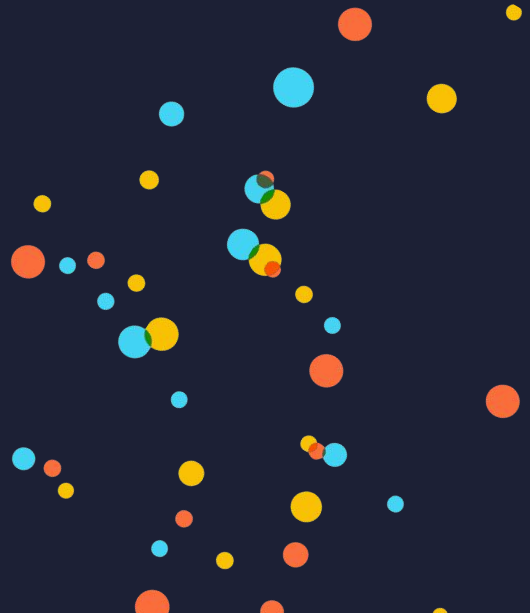
Merkle DAG:

CACHING & DEDUPING



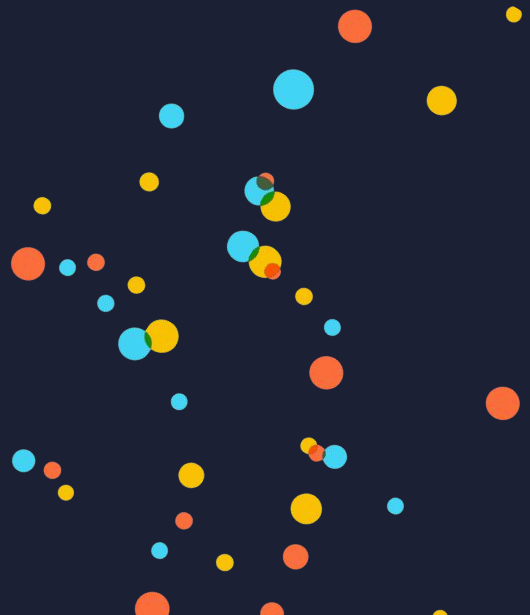
Demo:

BUILD A DAG
dag.ipfs.io



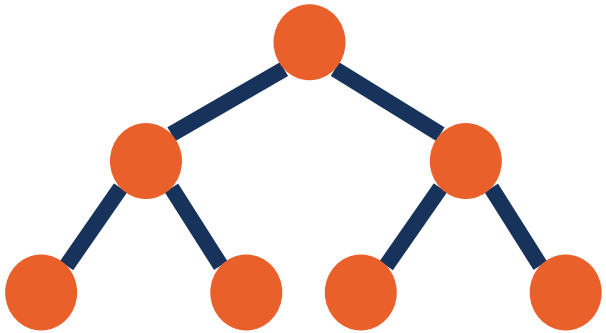
Demo:

DEDUPLICATION
dag.ipfs.io

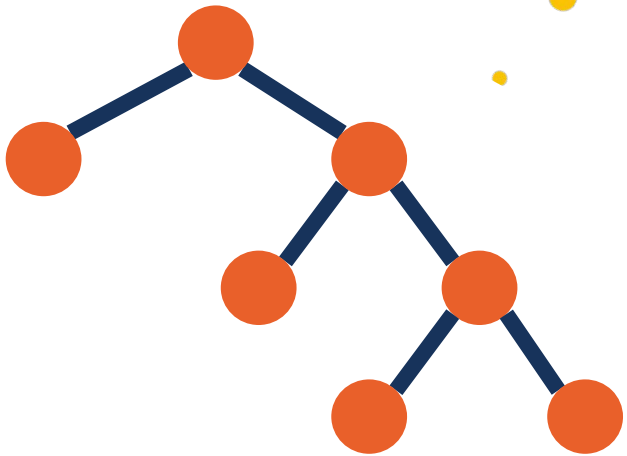


Importing files:

DAG LAYOUTS



Balanced

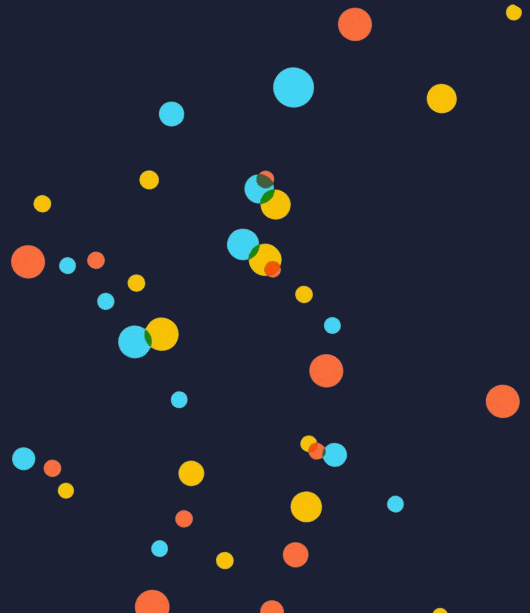


Trickle

Actual DAG structure may differ

Demo:

DAG LAYOUTS
dag.ipfs.io



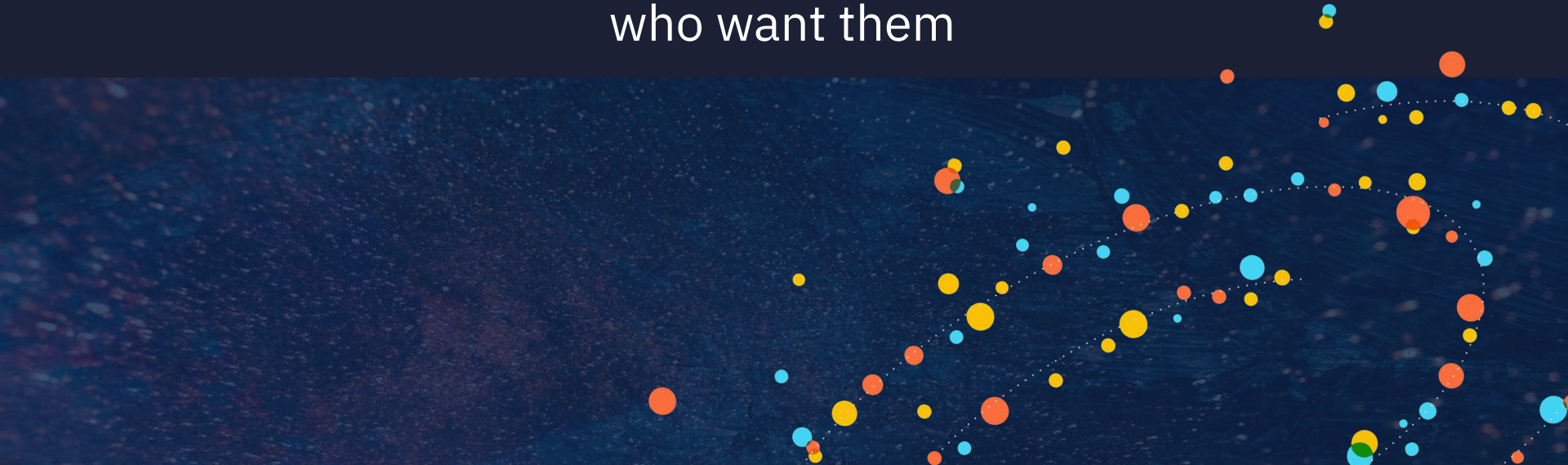
Bitswap

Trading blocks

Bitswap has two jobs:

To request blocks you need from your connected peers

To send blocks you have to peers who want them



Bitswap:

The Wantlist

Each peer stores the list of CIDs they want to get

A Wantlist message

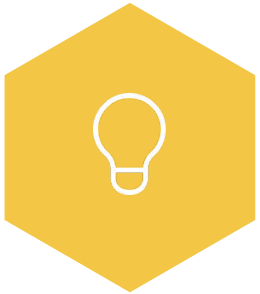
A list of CIDs you send as message to connected peers

A Blocks message

1 or more blocks prefixed with how to calculate their CIDs

Bitswap:

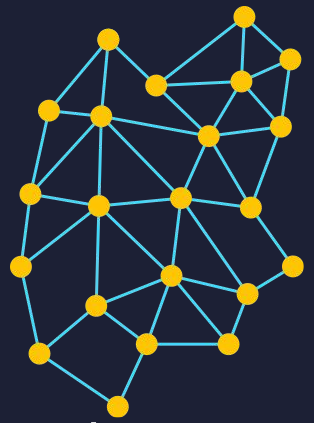
Sessions



At a protocol level, bitswap is simple. It’s message based and all messages contain wantlists, or blocks

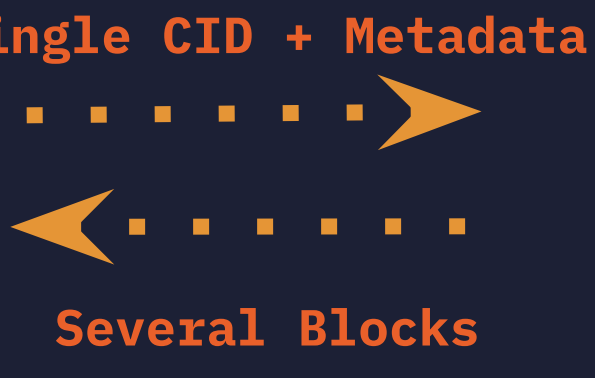
Bitswap Sessions add optimizations to only send wantlists to peers most likely to have the blocks you want.

GraphSync

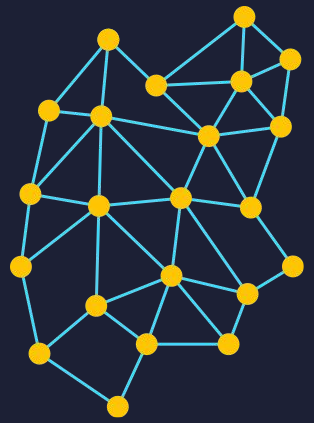


A protocol for synchronizing IPLD Graphs among peers. It allows to make a single request to a remote peer for all results of traversing an IPLD Selector on the remote peer's local IPFS graph.

GraphSync

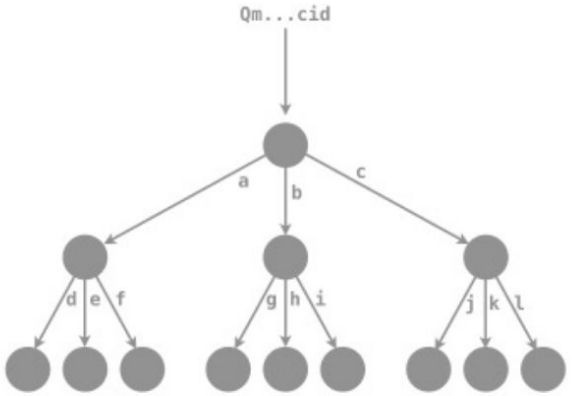


IPLD Selectors

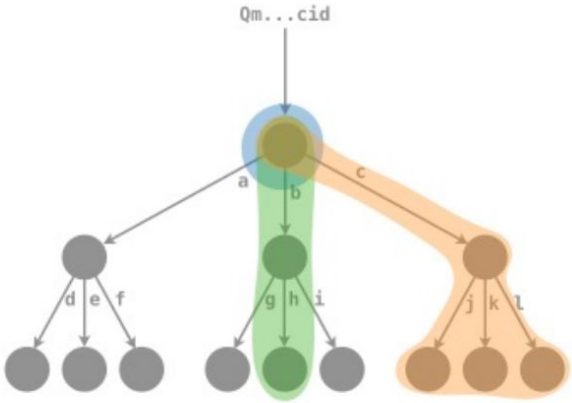


Expressions that identify ("select") a subset of nodes in an IPLD dag.

IPLD Selectors



IPLD Dag



IPLD Selectors

selector string	nodes
/sel/cid/Qm…cid	1
/sel/path/Qm…cid/b/h	3
/sel/glob/Qm…cid/c/*	5

Naming Data:

Mutable Data

Mutable Data:

IMMUTABILITY NOT ENOUGH

24h

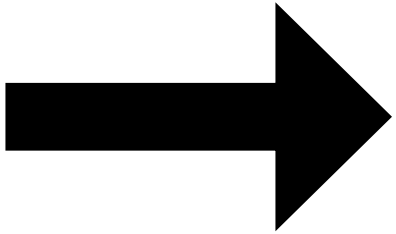
later

Naming Data:

IPNS

Name

Signing Key



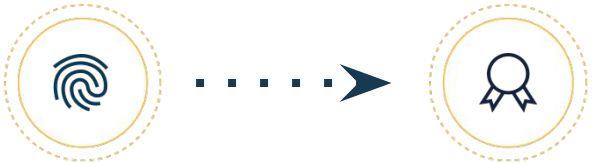
Kademlia
PubSub
DNS
MDNS
Workers

Data

Path

Mutable Data:

SIGNATURES



Mutabile Data:

IPNS

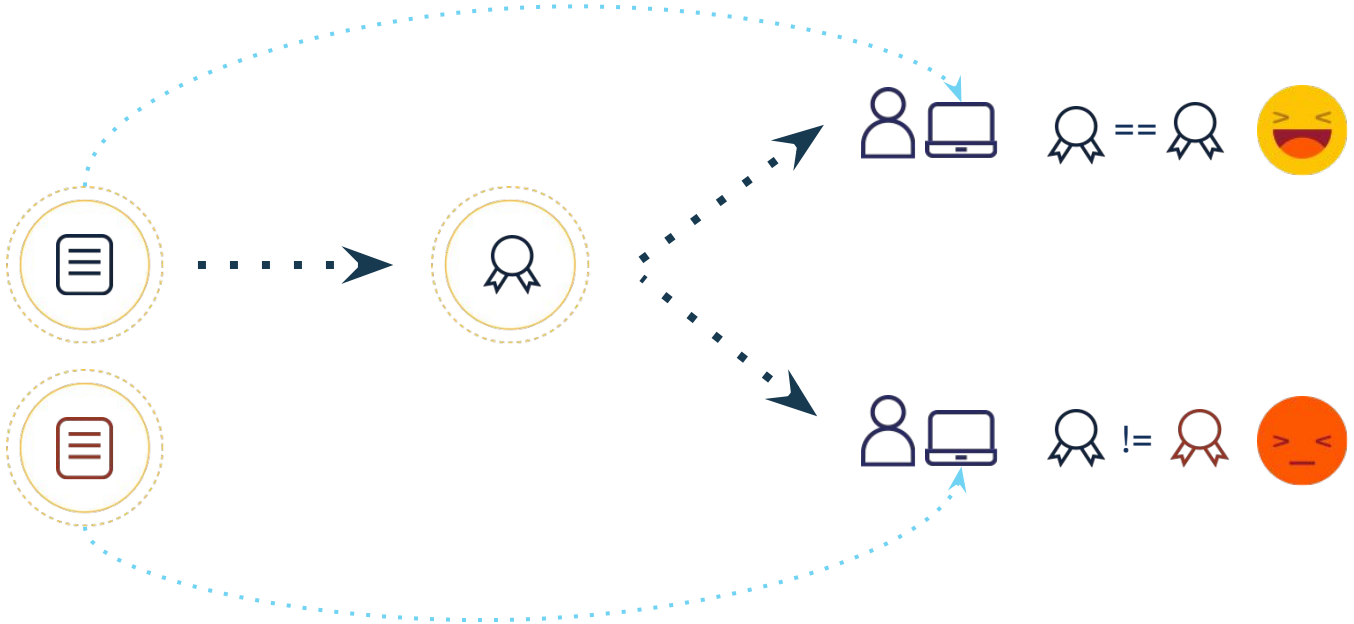
Path (e.g. /ipfs/bafy…)
Sequence Number
Signature
Public Key

Mutable Data:

IPNS

Mutabile Data

SIGNATURE CHECKING



Mutabile Data:

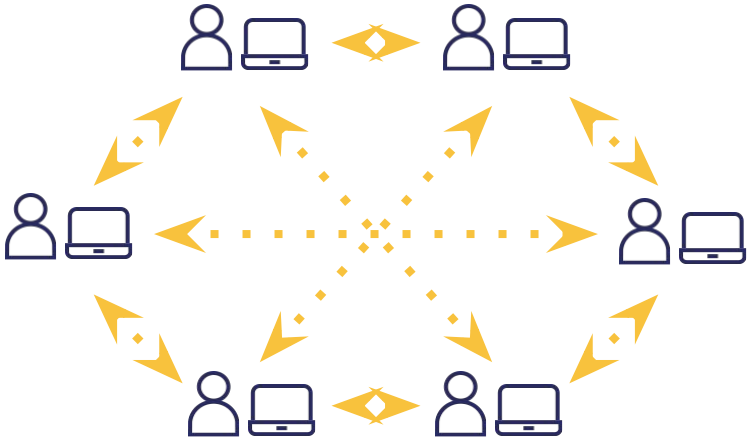
IPNS Routers

Router	Architecture	Update Propagation	Connections	Hops	Speed	Other notes
DHT	Distributed	Pull	100s	Log(Network Size)	Slow	High Bandwidth and Setup Cost
DNS	Centralized	Pull	1~3	1~3	Fast	
Workers	Centralized	Pull	1	1	Fast	
MDNS	Local Network	Pull	One per node in LAN	1	Fast	Only interesting for specific types of applications
PubSub FloodSub	Distributed*	Push	One per participating node	Network Diameter	Fast	Huge Bandwidth Cost.
PubSub GossipSub	Distributed*	Push	6~12	Log(Network Size)	Fast	Bandwidth cost under control as the network scales

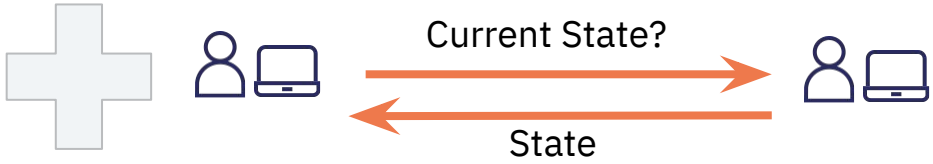
IPNS Pubsub Router

PubSub + Persistence Layer (Go-IPFS 0.5.0)

PubSub Protocol



Fetch Protocol



IPNS Pubsub Router

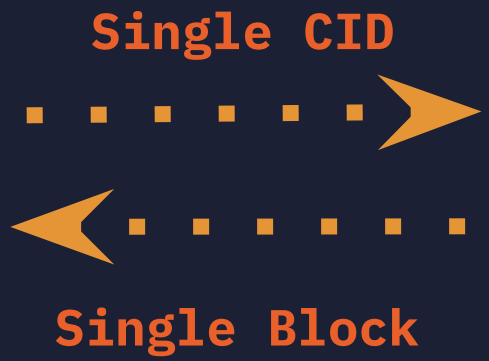
Multi-Writer Naming (Under development)

- IPNS: Version Number + Path
- Multi-Writer: Set of Previous Hashes
- PubSub is an opt-in system where each node has interest in the data
  - Willing to store full trees of hashes

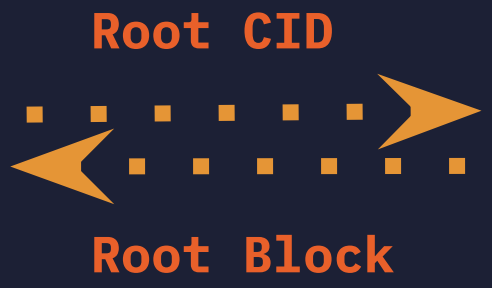
Phew, we did it!

Questions?

BitSwap



BitSwap



Parse Block, Get links

Several roundtrips