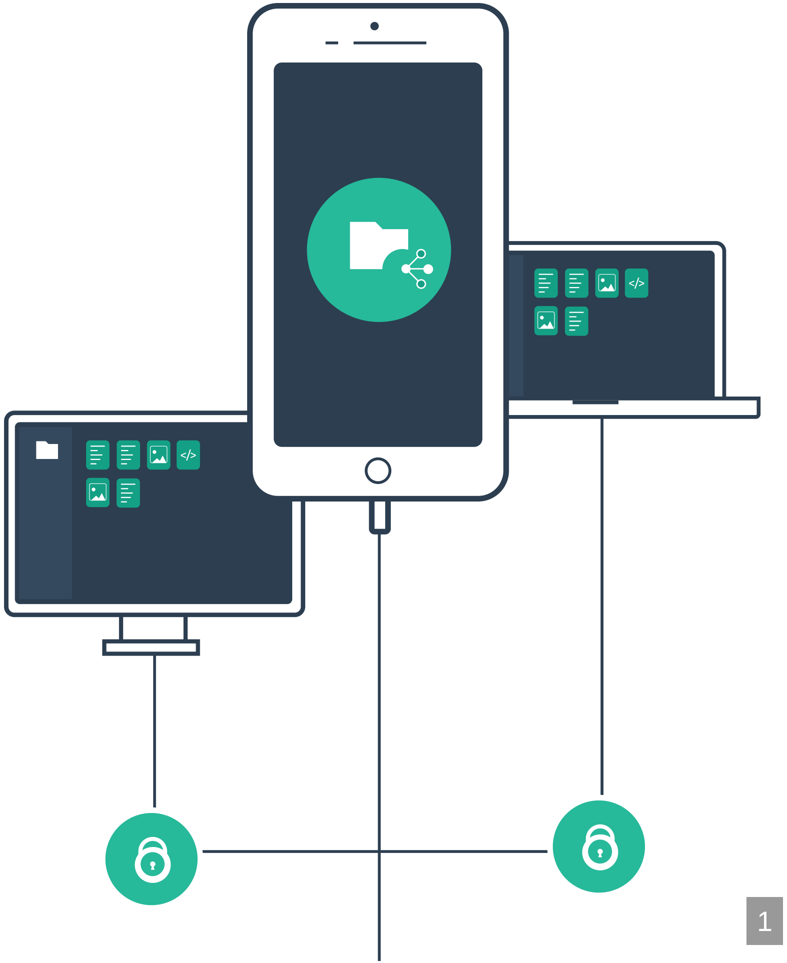
PEERGOS

Peergos: Capability-Based Access Control for an Encrypted Web

Dr Ian Preston
Co-founder, CEO



The web is under attack

- ~~empower people~~
- ~~improve lives~~
- ~~protect democracy~~
- surveillance
- control
- coercion

A better foundation

- E2EE
- host idependent
- self-sovereign identity
- protect metadata, including social graph

Peergos: a protocol for agency

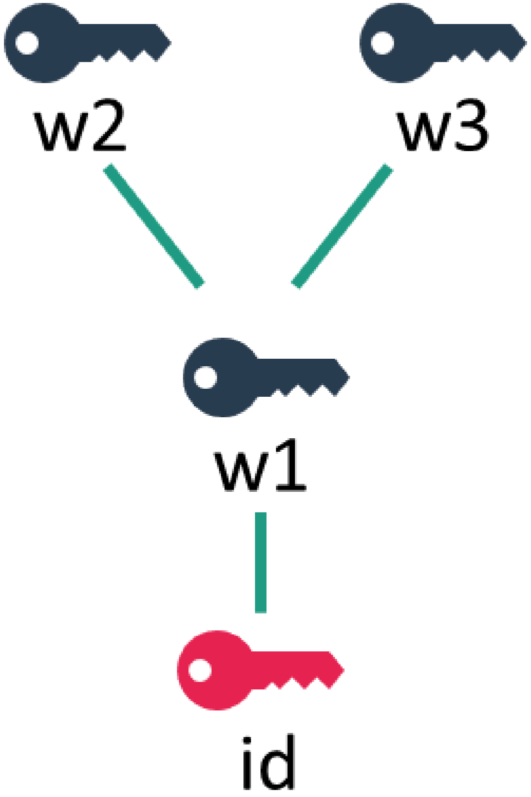
- Global, E2EE social filesystem
- Fine-grained access control
- Signed content-addressed (host independent)
- Automatic host migration (preserving links)
- Sandboxed apps/sites
- Portable, self-sovereign identity

The server is an adversary

1. Content-addressed data
2. Signed updates
3. End-to-end encryption

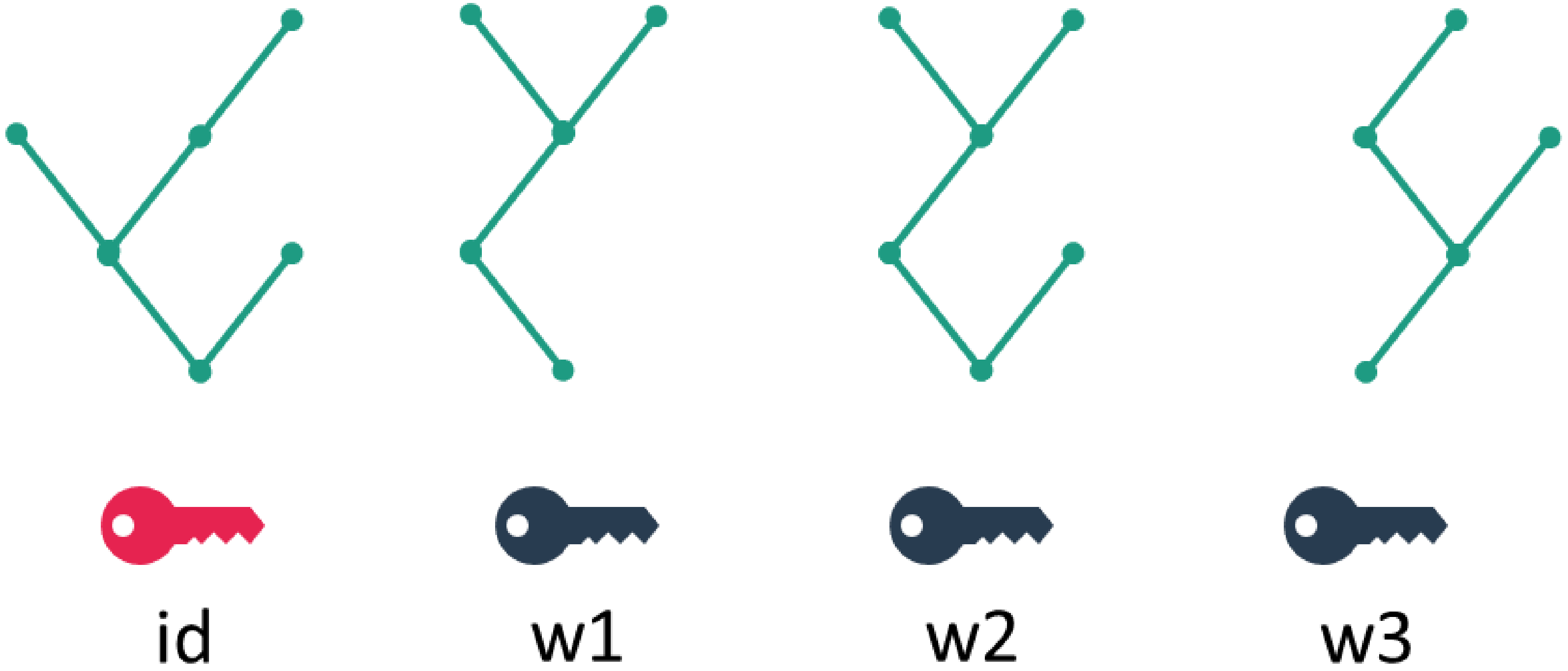
→ location independent addressing (host independence)

Signed merkle forest



Signed merkle forest

Signed merkle forest



CHAMP

- compressed hash-array mapped prefix-trie
- key value store
- independent of insertion order (unlike btree)
- balanced

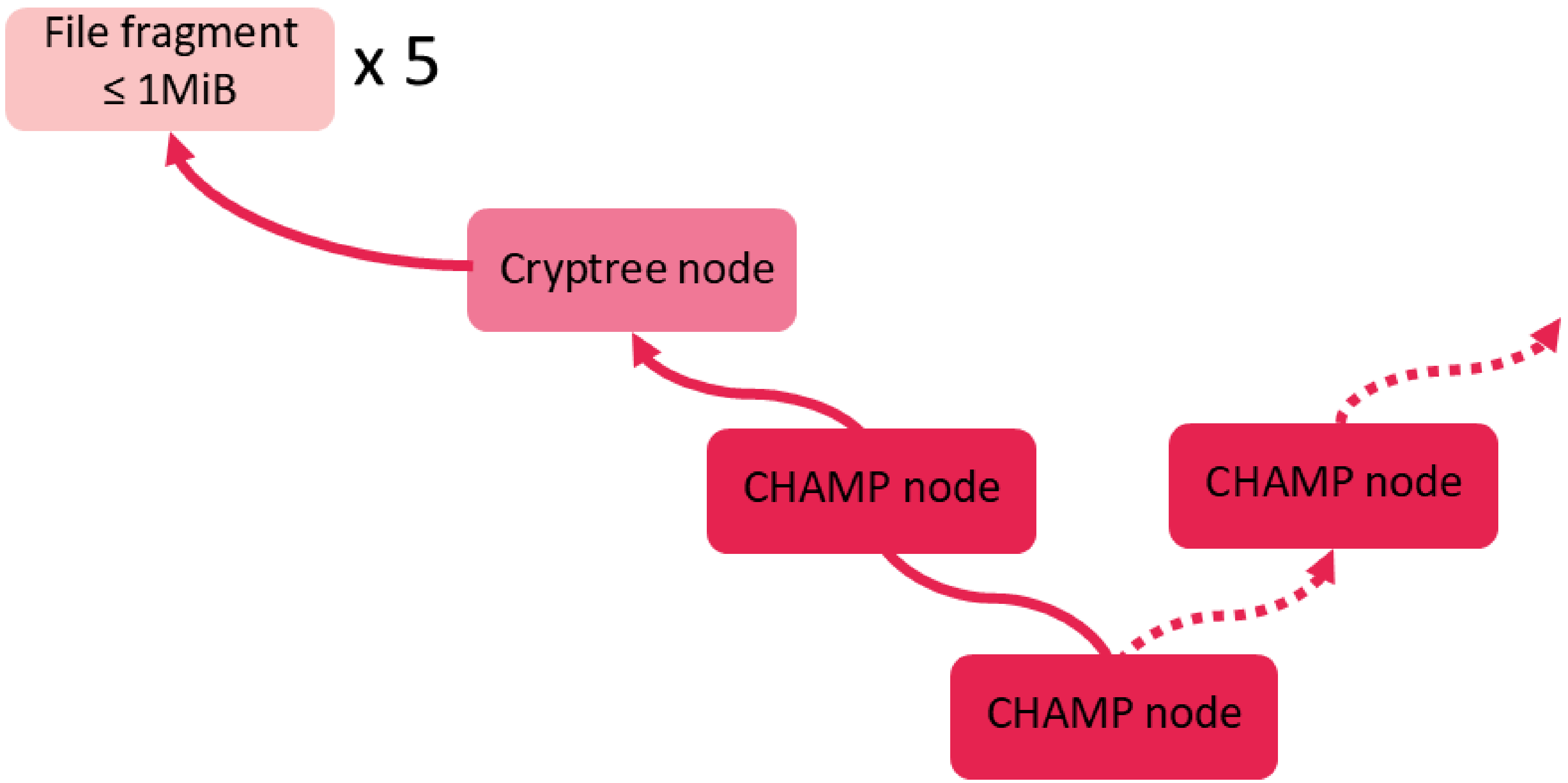
Keys are random 32 bytes

Values are cryptree nodes (encrypted metadata)

Each cryptree node can link to 5 encrypted blocks

Immutable state - file system CHAMP

File system CHAMP



Cryptree metadata privacy

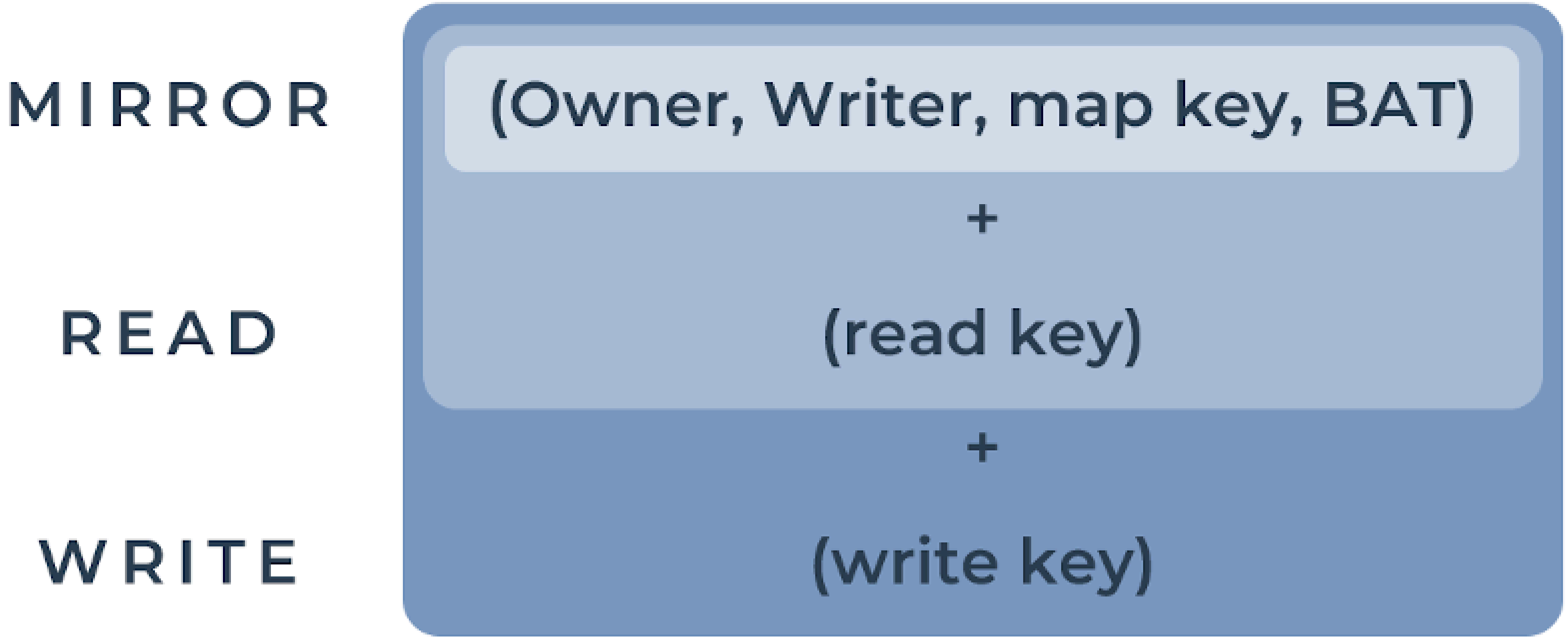
Host cannot see:

- file sizes
- file vs dir
- file/folder names
- folder topology
- social graph
- who or how many have access to a file/blob.

Capabilities and Access Control

Capabilities (Caps)

- Pure information, not identity-based
- Works in secret links etc.
- Can be revoked



MIRROR (Owner, Writer, map key, BAT)

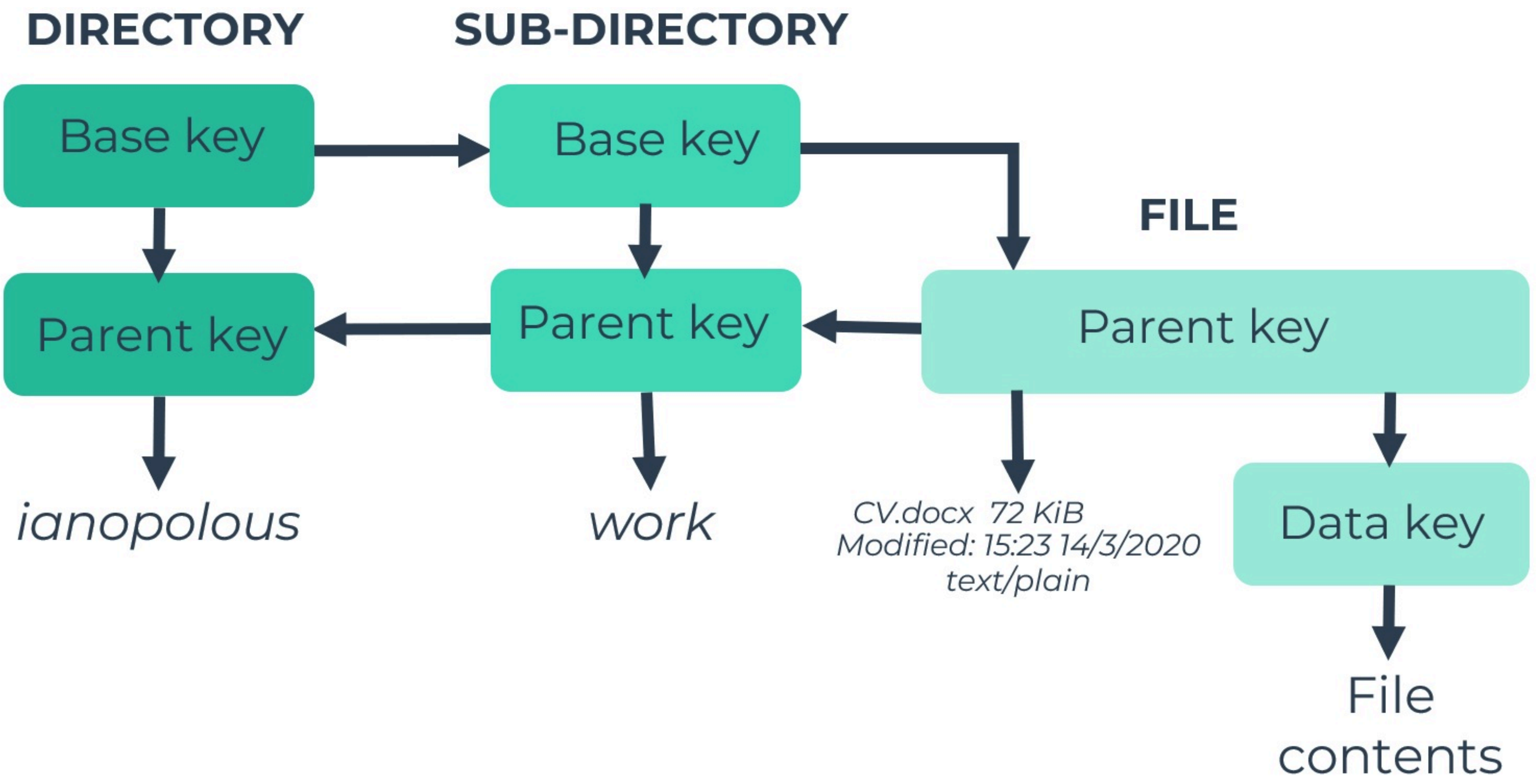
+

READ (read key)

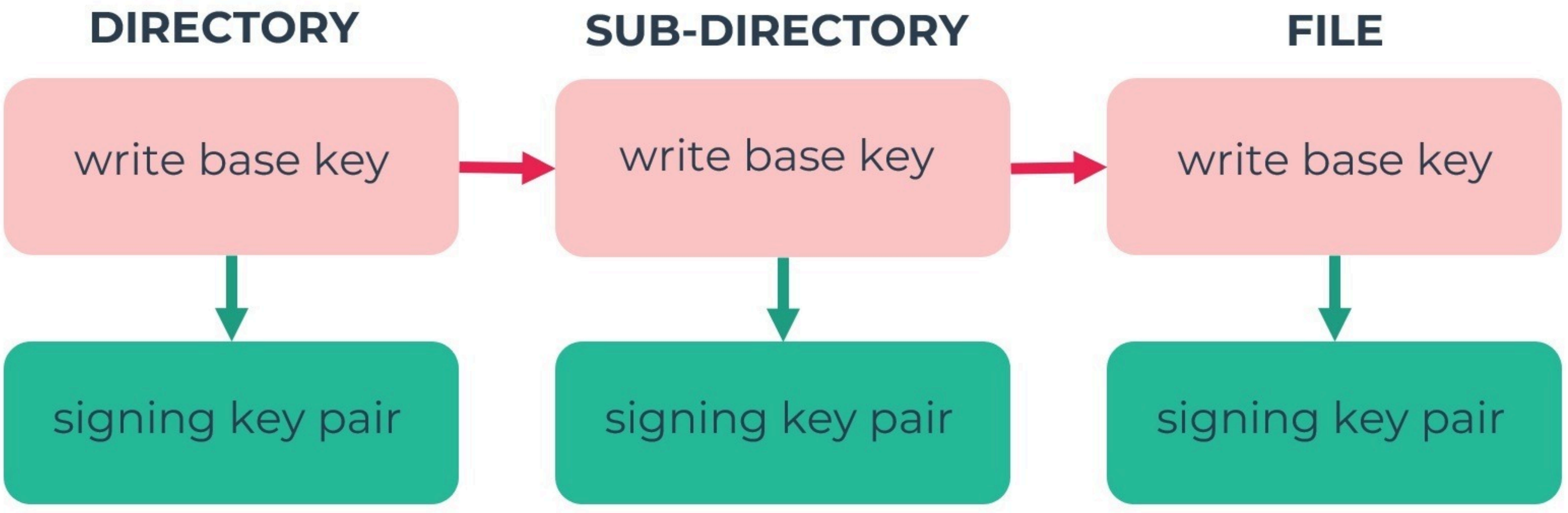
+

WRITE (write key)

Read Cryptree



Write Cryptree



Fast file seeking

How do we get cap to later chunks of a file?

mapkey => sha256(stream-secret + mapkey)

bat => sha256(stream-secret + bat)

1. local hashing
2. a single champ.get to retrieve the encrypted metadata
3. up to 5 block.get calls for fragments

DEMO

Fast seeking within a movie

Websites and Applications

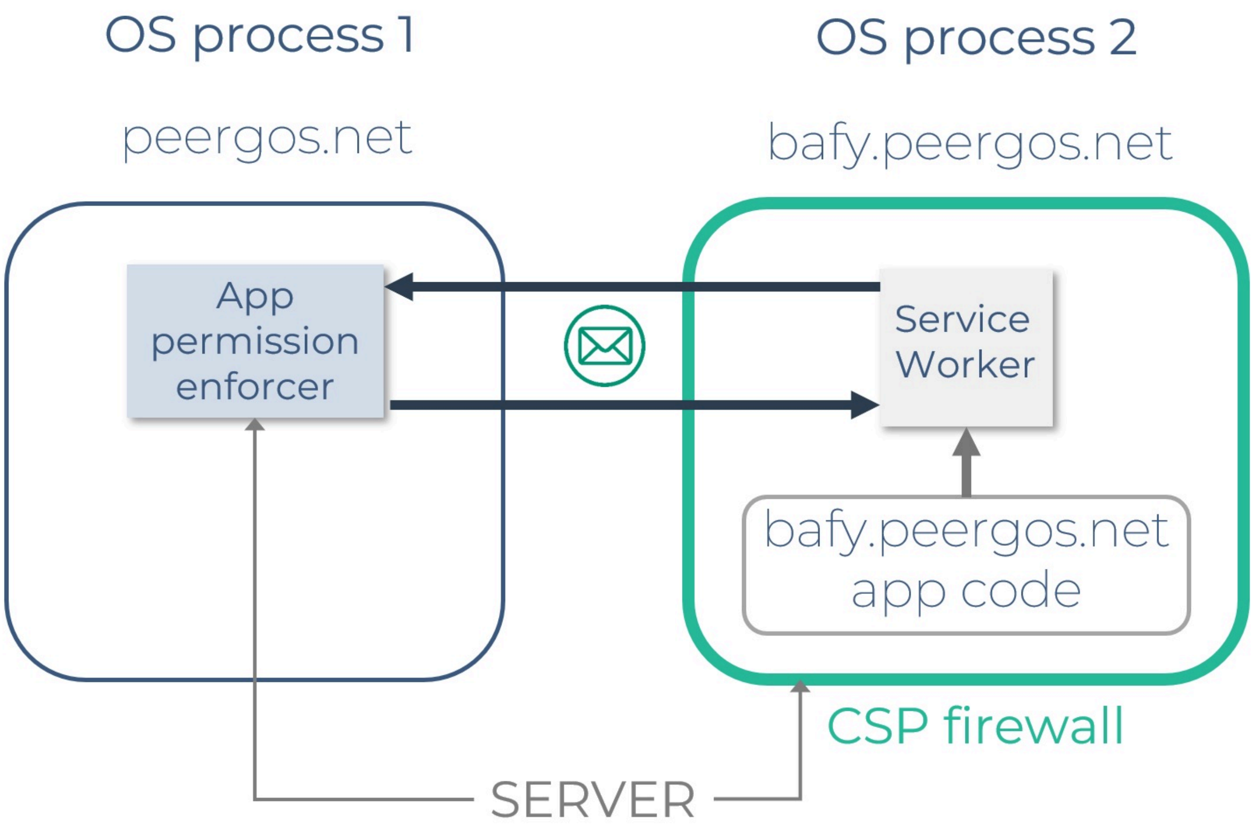
Exfiltration-proof apps

- run untrusted code over private data
- an app can't exfiltrate data
- an app can't read anything it's not granted acess to
- works in existing browsers, without add-ons
- simple REST API (without a server!)

Application authoring

- just a folder of HTML5
- author controls visibility
- basic permissions (e.g. register for certain file types)
- frictionless publishing - drag and drop
- 0 permissions = private website

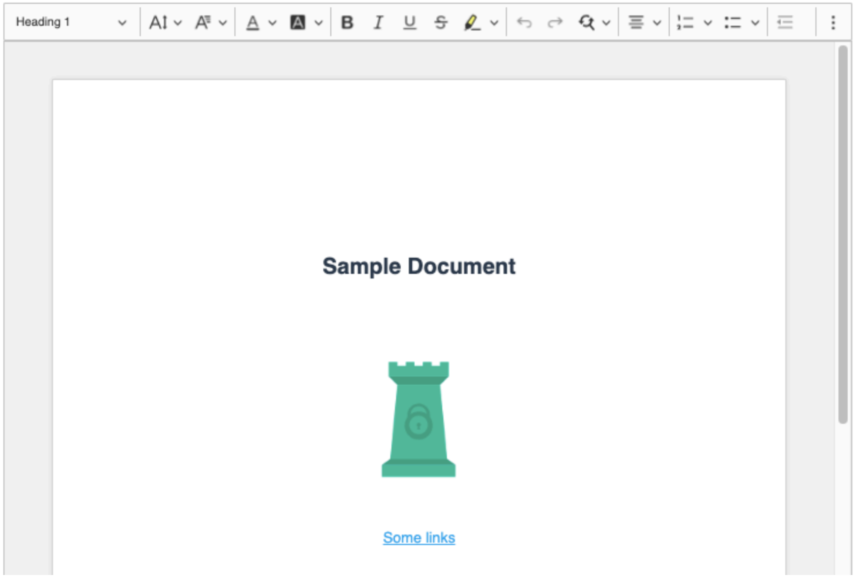
Application sandbox



What can an app do?

Edit files

- Word processor



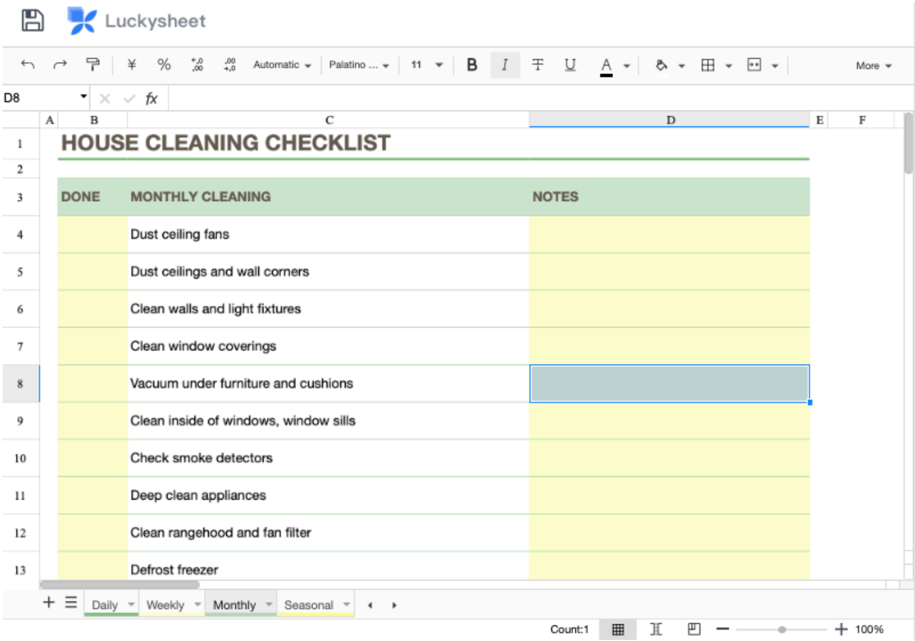
Sample Document

Some links

What can an app do?

Edit files

- Word processor
- Spreadsheets



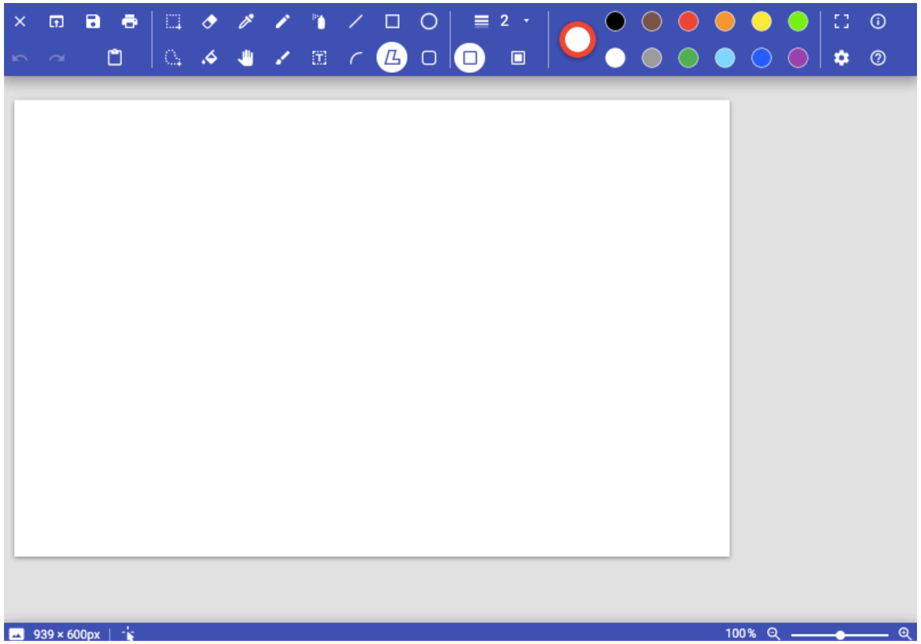
HOUSE CLEANING CHECKLIST

DONE	MONTHLY CLEANING	NOTES
	Dust ceiling fans	
	Dust ceilings and wall corners	
	Clean walls and light fixtures	
	Clean window coverings	
	Vacuum under furniture and cushions	
	Clean inside of windows, window sills	
	Check smoke detectors	
	Deep clean appliances	
	Clean rangehood and fan filter	
	Defrost freezer	

What can an app do?

Edit files

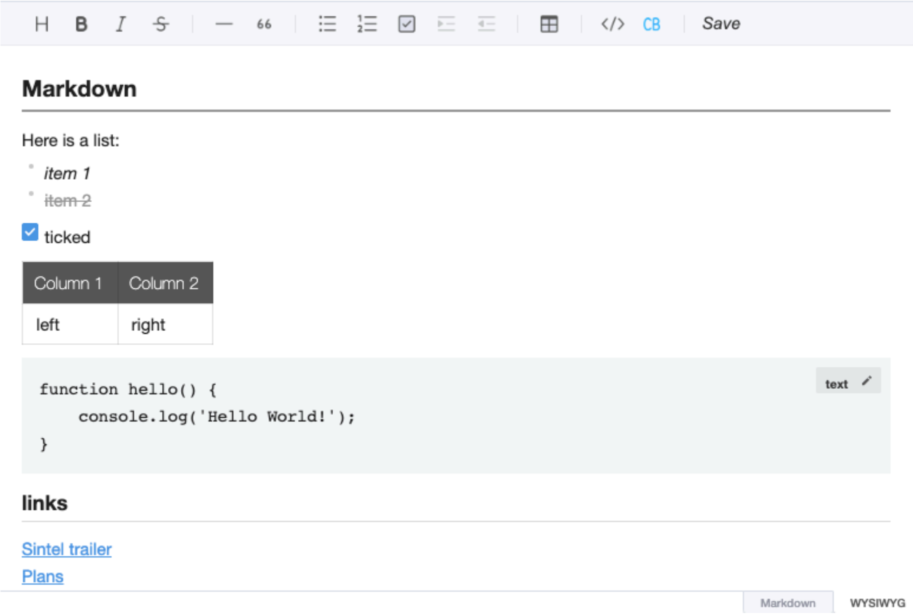
- Word processor
- Spreadsheets
- Image editor



What can an app do?

Edit files

- Word processor
- Spreadsheets
- Image editor
- Markdown editor



Markdown

Here is a list:

- *item 1*
- ~~item 2~~

☑ ticked

Column 1	Column 2
left	right

```
function hello() {
    console.log('Hello World!');
}
```

links

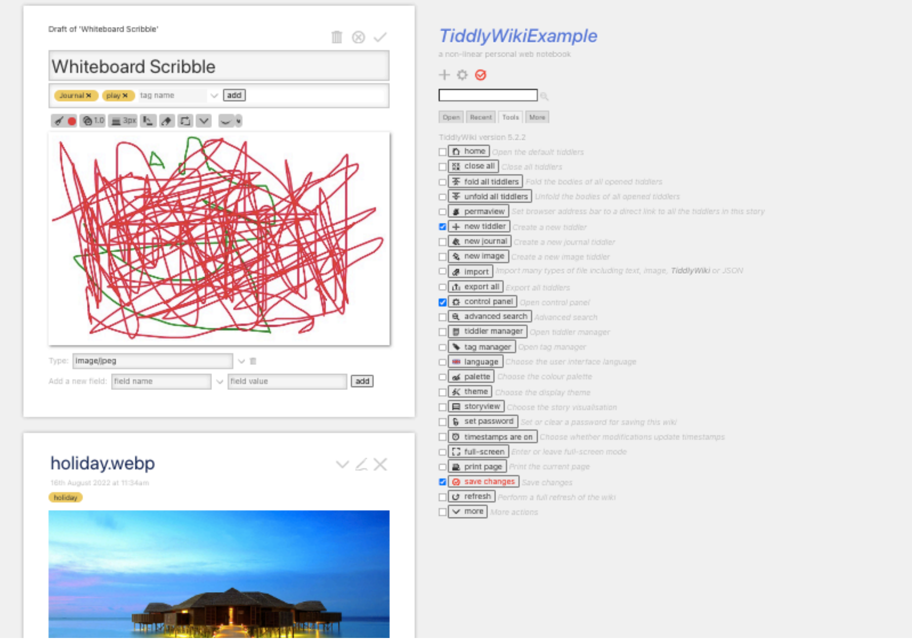
[Sintel trailer]

[Plans]

What can an app do?

Edit files

- Word processor
- Spreadsheets
- Image editor
- Markdown editor
- Tiddlywiki notebooks



Draft of 'Whiteboard Scribble'

Whiteboard Scribble

Journal ✕ play ✕

Type: image/jpeg

Add a new field: field name field value

holiday.webp

16th August 2022 at 11:34am

holiday

TiddlyWikiExample

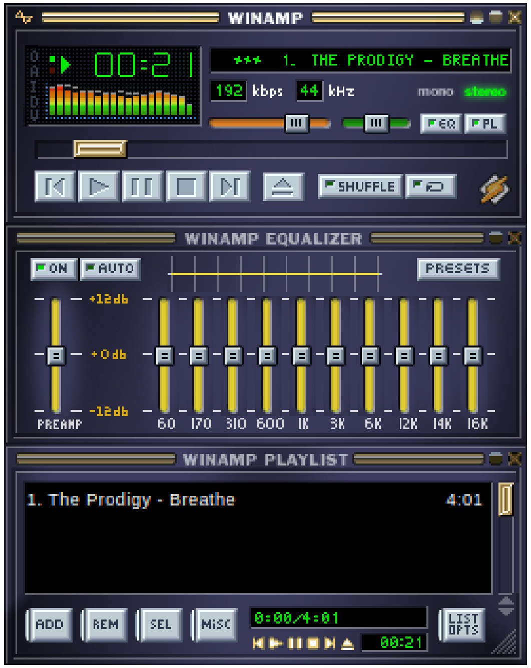
a non-linear personal web notebook

TiddlyWiki version 5.2.2

- □ home Open the default tiddlers
- □ close all Close all tiddlers
- □ fold all tiddlers Fold the bodies of all opened tiddlers
- □ unfold all tiddlers Unfold the bodies of all opened tiddlers
- □ permaview Set browser address bar to a direct link to all the tiddlers in this story
- ☑ new tiddler Create a new tiddler
- □ new journal Create a new journal tiddler
- □ new image Create a new image tiddler
- □ import Import many types of file including text, image, TiddlyWiki or JSON
- □ export all Export all tiddlers
- ☑ control panel Open control panel
- □ advanced search Advanced search
- □ tiddler manager Open tiddler manager
- □ tag manager Open tag manager
- □ language Choose the user interface language
- □ palette Choose the colour palette
- □ theme Choose the display theme
- □ storyview Choose the story visualisation
- □ set password Set or clear a password for saving this wiki
- □ timestamps are on Choose whether modifications update timestamps
- □ full-screen Enter or leave full-screen mode
- □ print page Print the current page
- ☑ save changes Save changes
- □ refresh Perform a full refresh of the wiki
- □ more More actions

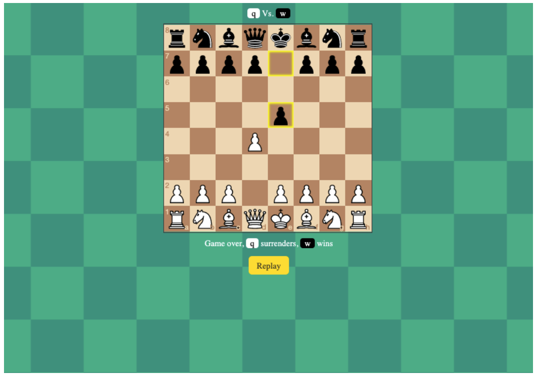
What else can an app do?

- Media player

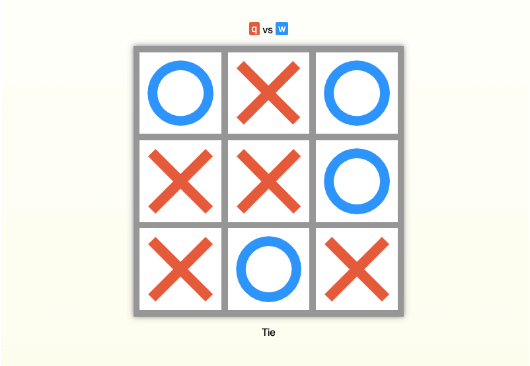


What else can an app do?

- Media player
- Multiplayer games



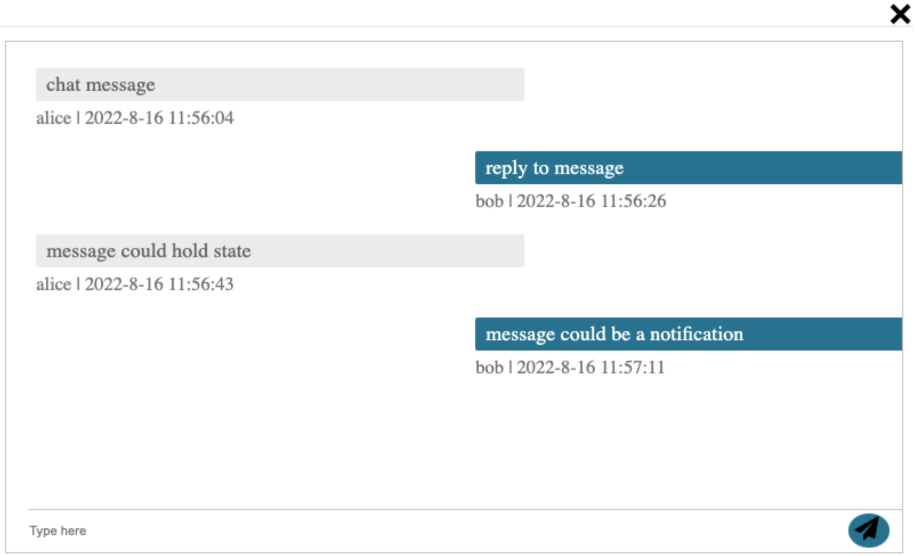
Game over, q surrenders, w wins



Tie

What else can an app do?

- Media player
- Multiplayer games
- Chat



chat message

alice | 2022-8-16 11:56:04

reply to message

bob | 2022-8-16 11:56:26

message could hold state

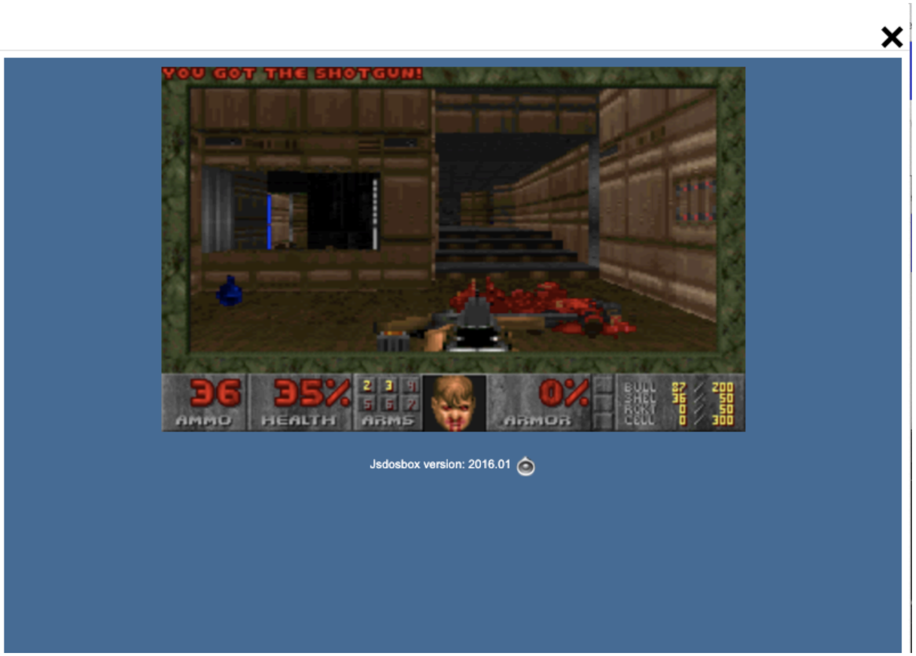
alice | 2022-8-16 11:56:43

message could be a notification

bob | 2022-8-16 11:57:11

What else can an app do?

- Media player
- Multiplayer games
- Chat
- Doom



More example apps:
github.com/peergos/example-apps

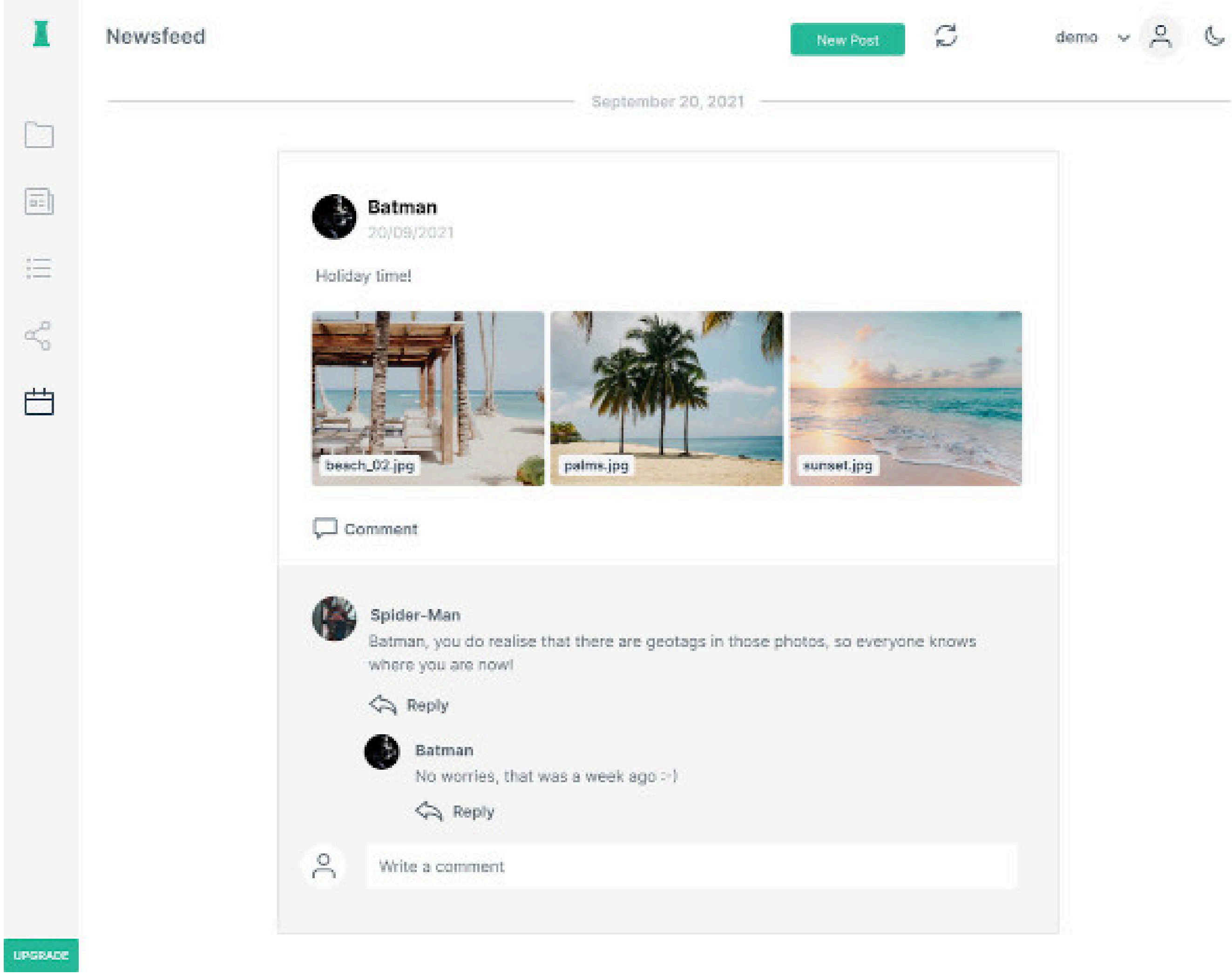
DEMO

Some apps

Social media

- most social media is bad for society
  - Allows micro targeted political profiling
  - Designed to be addictive
- normal in-person conversations are good for society
- Peergos social media is modelled on in-person conversations

Social feed



September 20, 2021

Batman
20/09/2021

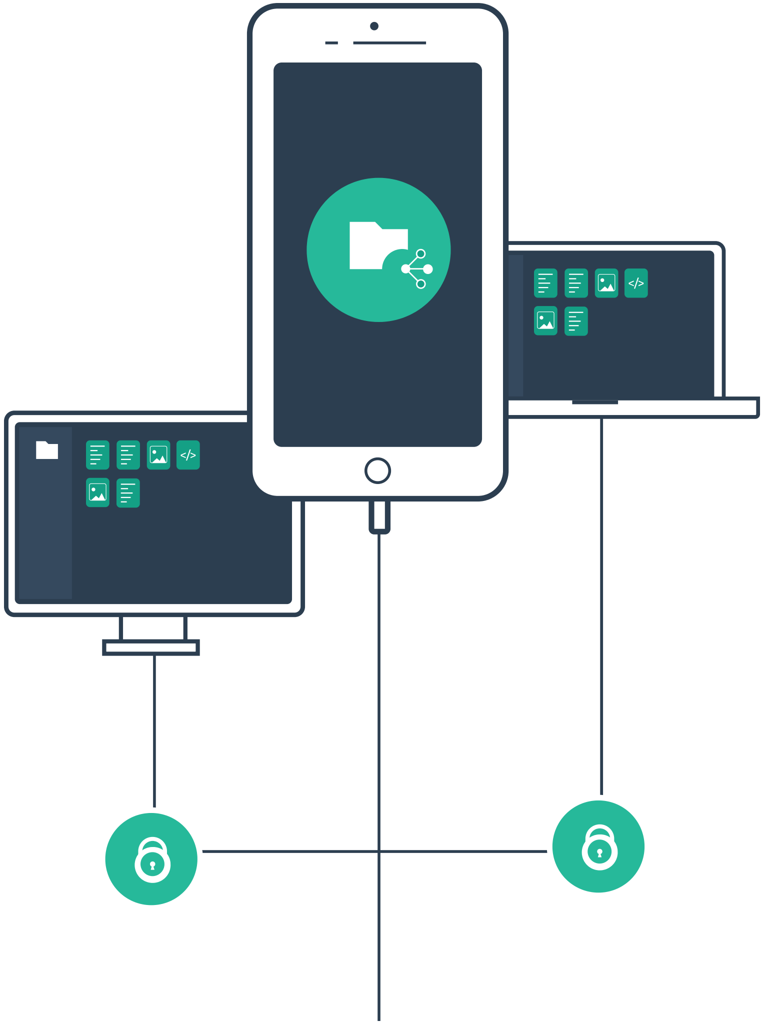
Holiday time!

Spider-Man
Batman, you do realise that there are geotags in those photos, so everyone knows where you are now!

Batman
No worries, that was a week ago :-)

Without privacy, there can be no democracy.

PEERGOS



Have fun!

1. Self host, or sign up: peergos.net

2. Write your own apps

3. Protect yourself and democracy

Questions?

Dr Ian Preston

ian@peergos.org

peergos.org

@peergos

"The defence of privacy will be the saviour of the future."

- Gus Hosein,

Executive Director Privacy International

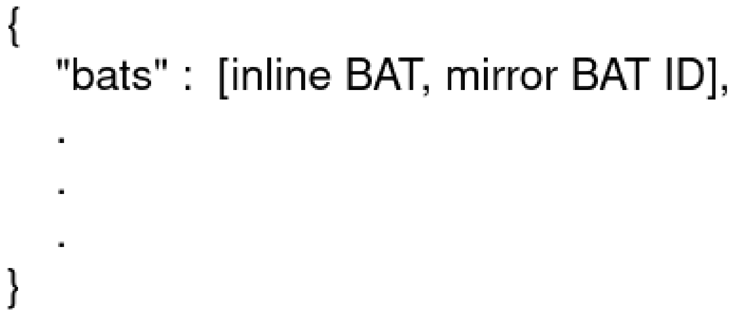
Block Access Tokens (BATs)

- Don't put encryped data in public!
- Post-quantum ciphertext access control
- 2 BATs per block
- Send S3 V4 sig (89 bytes) with hash
- Tied to requesting peer-id and time
- Recipient verifies signature against requesting peer-id and the BAT

Inline BATs

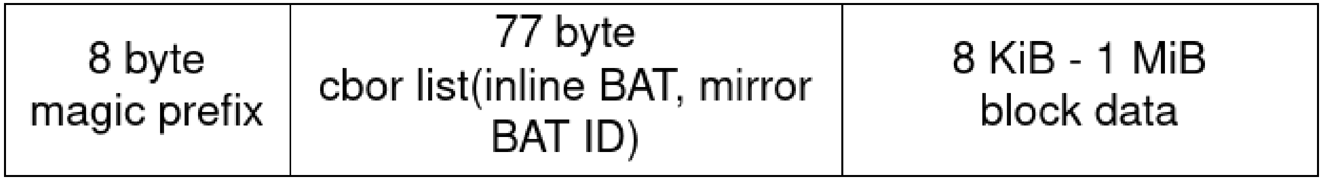
- BAT = 32 random bytes

CBOR



```
{
    "bats" :  [inline BAT, mirror BAT ID],
    .
    .
    .
}
```

RAW



8 byte magic prefix	77 byte cbor list(inline BAT, mirror BAT ID)	8 KiB - 1 MiB block data

Why is privacy so important?

- Fundamental human right

  "No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence."
  The Universal Declaration of Human Rights

Necessary for democracy

Without privacy:

- Mass surveillance
- Monitoring and crushing dissent
- Voter manipulation from profiling
- Swinging elections

What happens when you visit a website?

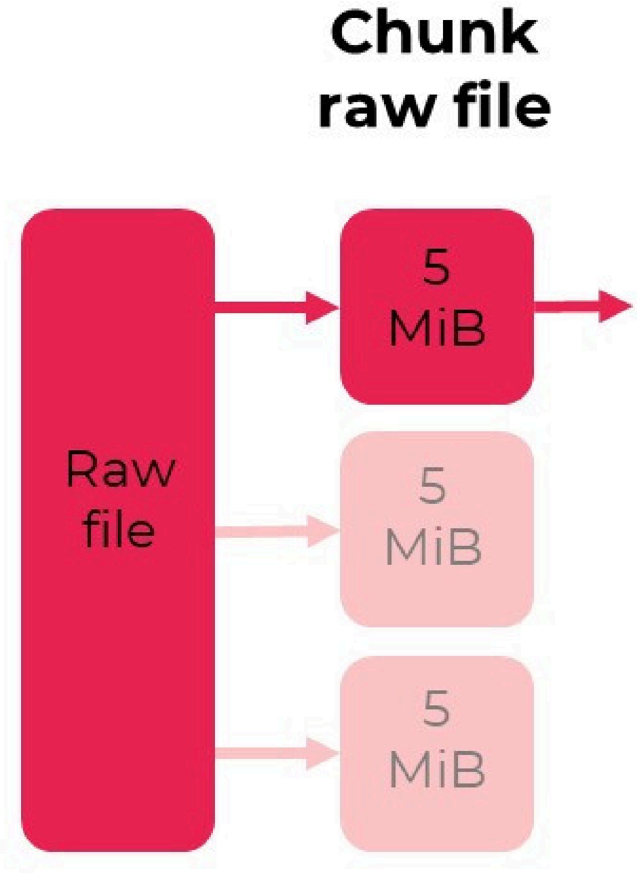
- DNS lookup
- TLS connection (trust 140 CAs)
- Arbitrary code from server
- Load 3rd party code
- Login to site?
- Send personal data to server?
- Manipulation by AI curated feed
- Subverted democracy, compromised consensus

Can a better design fix these problems?

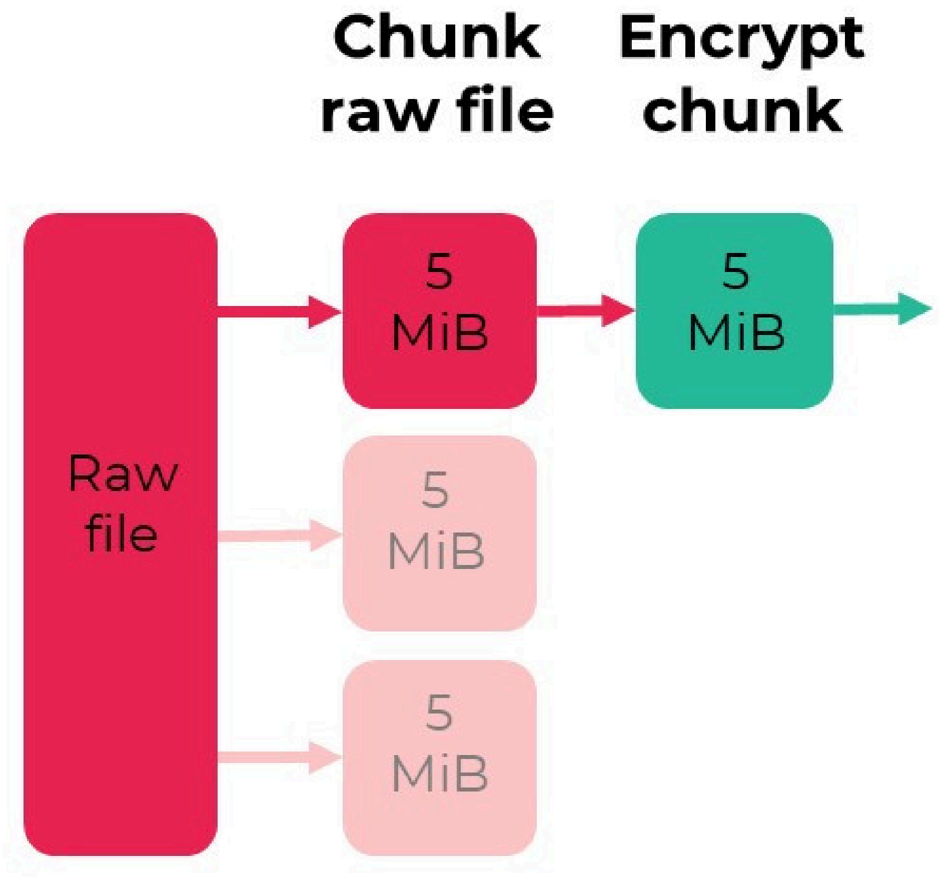
Requirements

- Protect against 3rd and 1st party surveillance
- Remove incentives that resulted in surveillance capitalism
- Return data and identity ownership to users
- Take your data between hosts and apps
- Work in existing browsers
- Work offline

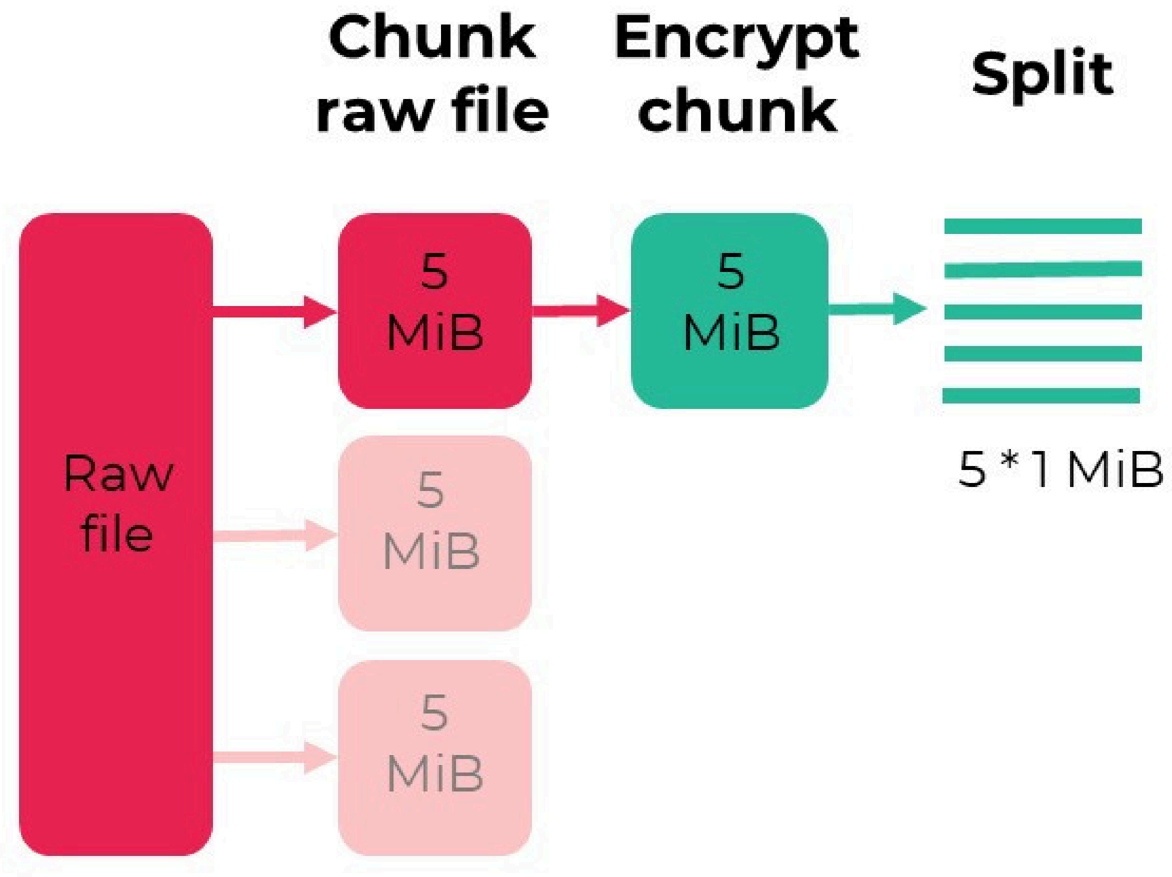
File upload



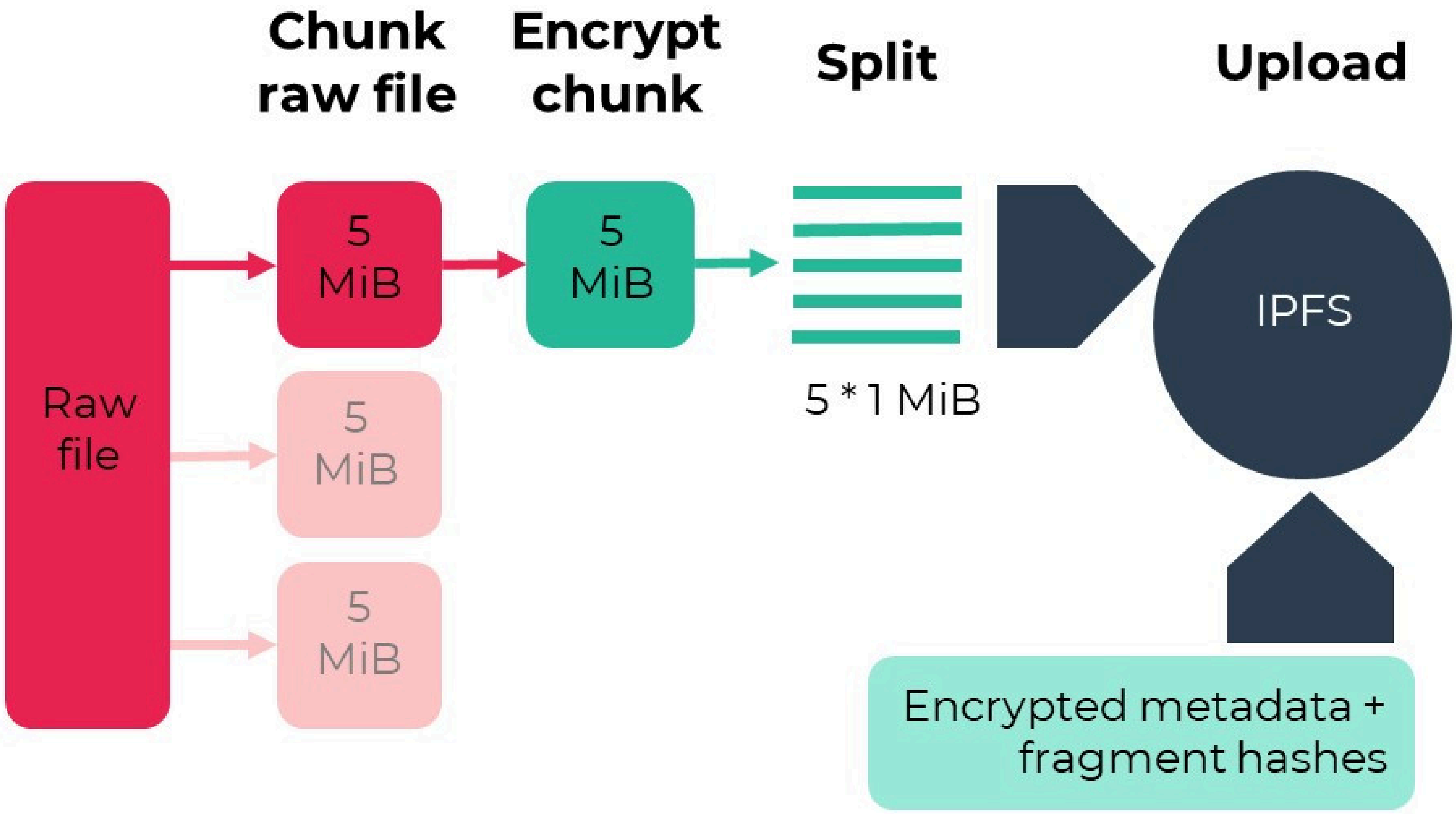
File upload



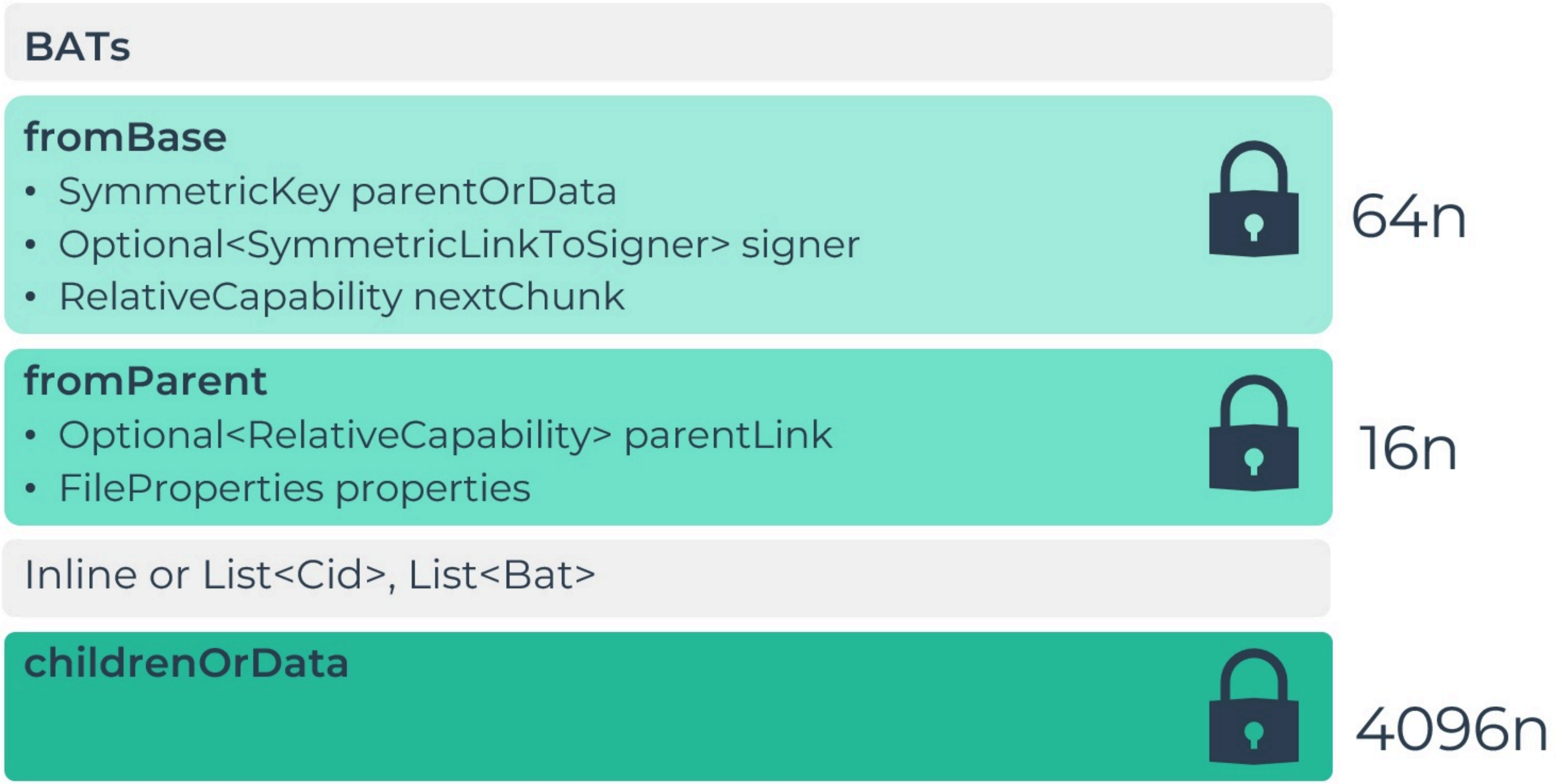
File upload


Chunk raw file
Encrypt chunk
Split
Raw file
5 MiB
5 MiB
5 MiB
5 MiB
5 * 1 MiB

File upload


Chunk raw file
Encrypt chunk
Split
Upload
Raw file
5 MiB
5 MiB
5 MiB
5 MiB
5 * 1 MiB
IPFS
Encrypted metadata + fragment hashes

Cryptree format



BATs

fromBase

- SymmetricKey parentOrData
- Optional<SymmetricLinkToSigner> signer
- RelativeCapability nextChunk

64n

fromParent

- Optional<RelativeCapability> parentLink
- FileProperties properties

16n

Inline or List<Cid>, List<Bat>

childrenOrData

4096n

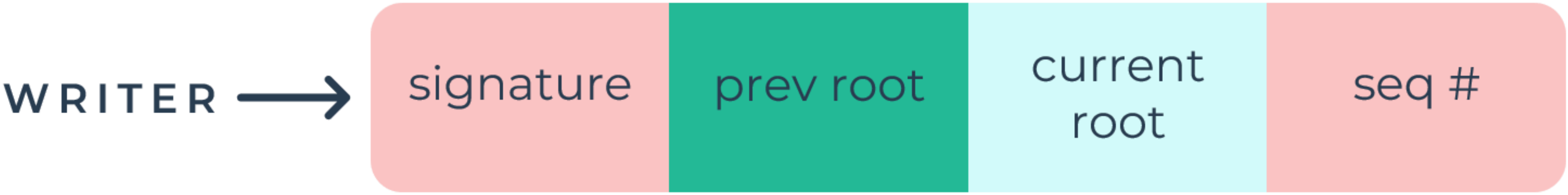
Retrieving a capability (PKI)

1. Use PKI (locally) to look up owner's host peer-id

PKI: username => signed(username, host, identity)

Retrieving a capability (mutable)

2. P2P HTTP request to host to get mutable pointer for writer
3. Verify signature and sequence for pointer
4. Get root hash from pointer



Retrieving a capability (immutable metadata)

5. champ.get(root, map key, BAT) => blocks (P2P HTTP request)
6. Repeat CHAMP lookup locally with returned blocks
7. Use read key to decrypt cryptree node, and get metadata

Retrieving a capability (immutable data)

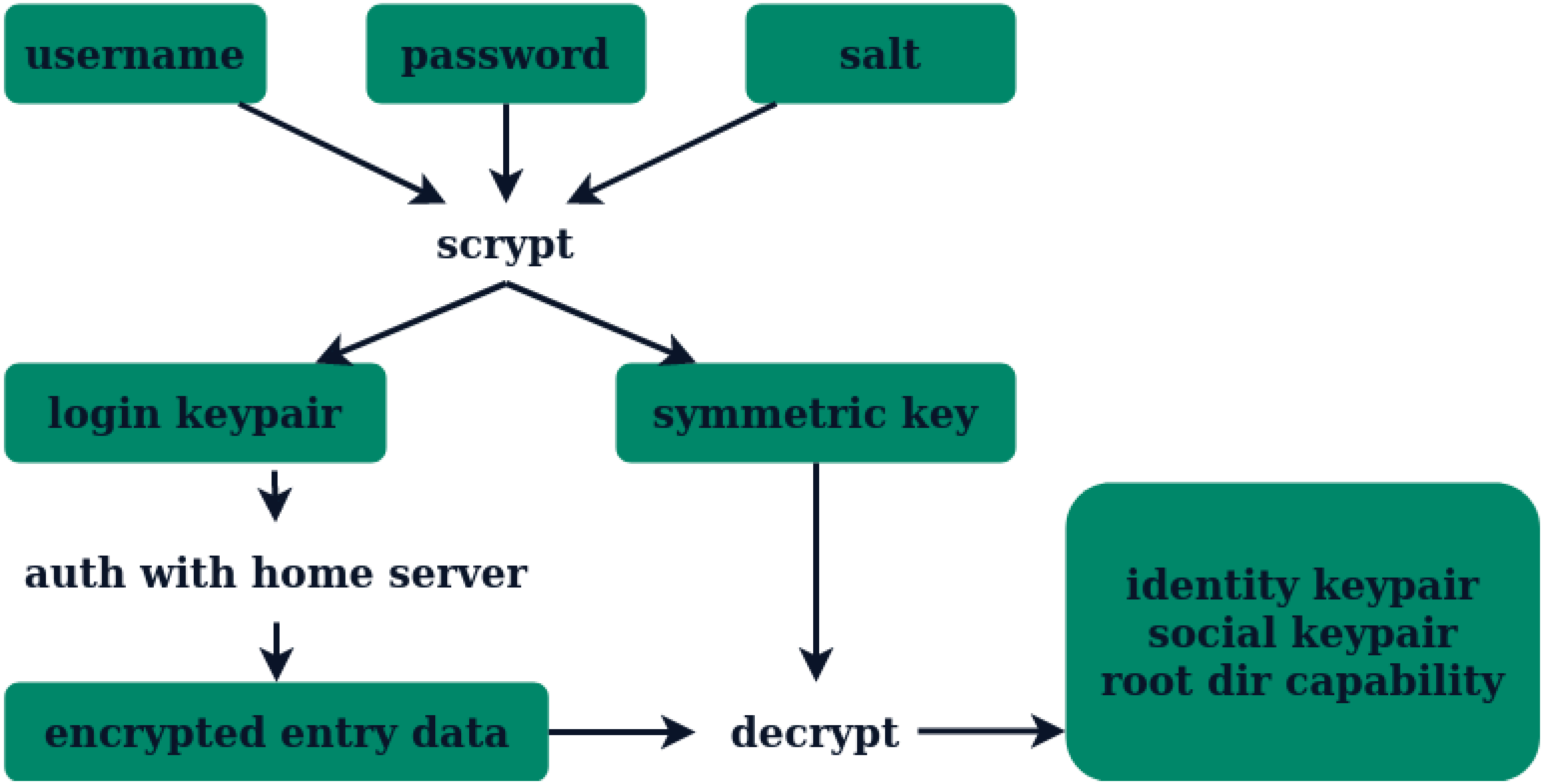
8. Get blocks for any chunk fragments using BATs and hashes in cryptree node
9. Concat blocks and decrypt (max 5 MiB)
10. Calculate next chunk's map key =

    sha256(stream-secret + current chunk map key)
11. Repeat steps 5-10

Browser app

- special app that renders folders of HTML
- isolates different folders
- internal links (relative)
- external links /peergos/username/path/to/file.html
- works in secret links!
- markdown pages work!

Login

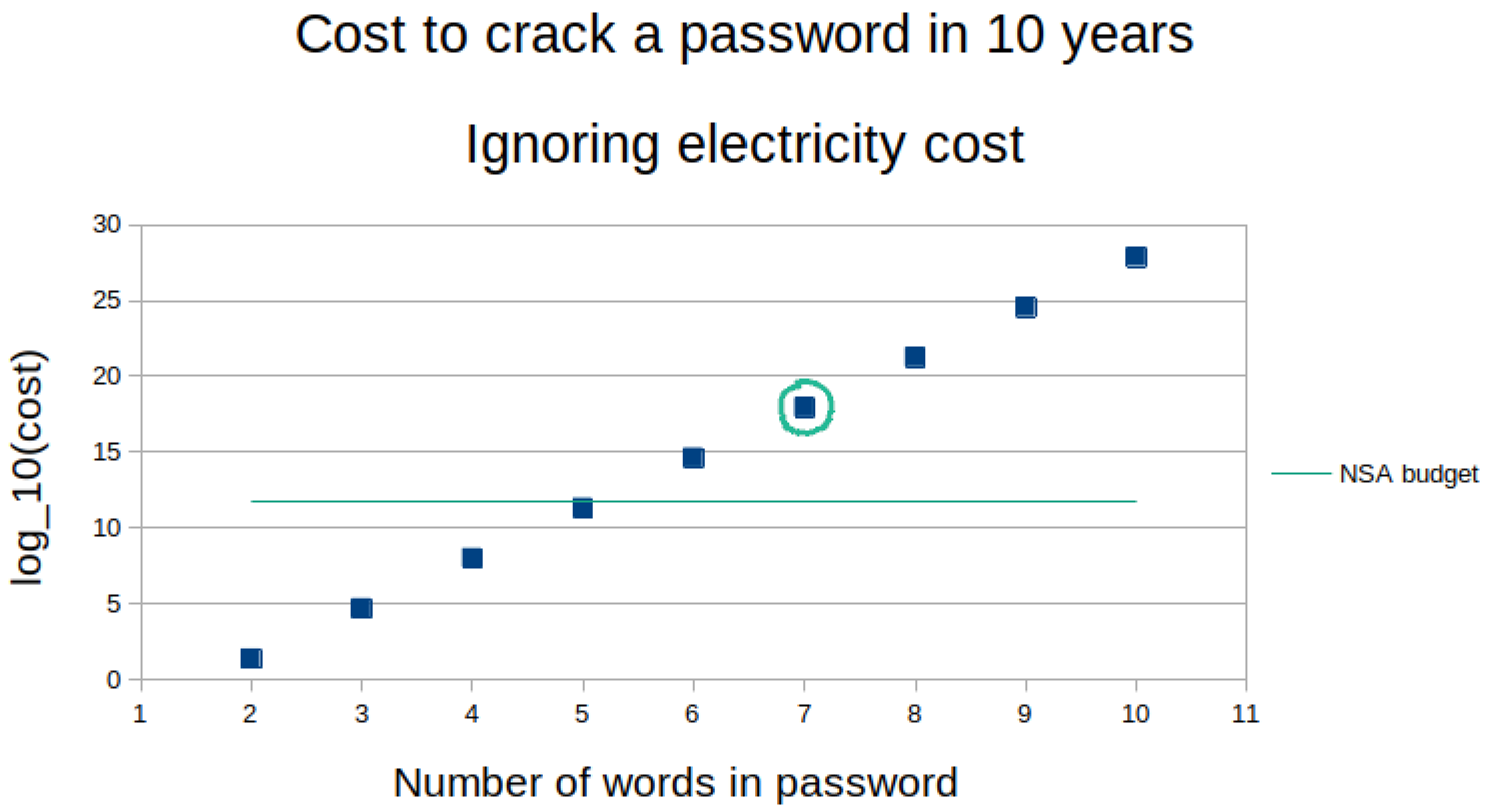


username
password
salt
scrypt
login keypair
symmetric key
auth with home server
encrypted entry data
decrypt
identity keypair
social keypair
root dir capability

Login security

- humans shouldn't create passwords
- generate password - 7 words from 2048 word list
- generated passwords have 7*11 = 77 bits of entropy
- GPU can calculate ~1M scrypt hashes per second
- 1 GPU would take 5 billion years
- 1 million GPUs would take 5000 years

Brute forcing a login (after hacking server)



Social

- follow request to open encrypted channel
- encrypted inbox for follow requests

App structure

- assets/index.html
- peergos-app.json

With STORE_APP_DATA permission

- data/

App manifest

```
{
    "displayName": "Painter",
    "description": "MS Paint clone,
    "version": "1.0.7",
    "author": "alice",
    "launchable": true,
    "folderAction": false,
    "appIcon": "icon.png",
    "fileExtensions": ["jpg","png"],
    "fileTypes":["image"],
    "permissions": ["EDIT_CHOSEN_FILE"]
}
```

Open a file of a certain type

- fileExtensions, fileTypes and mimeTypes
- wildcard supported

Permissions

- STORE_APP_DATA
- EDIT_CHOSEN_FILE
- READ_CHOSEN_FOLDER
- EXCHANGE_MESSAGES_WITH_FRIENDS
- ACCESS_PROFILE_PHOTO

Run parameters

Passed via query parameters

- path
- isPathWritable
- theme

REST API

Write and get app-private data:
/peergos-api/v0/data/path.to.file

POST a HTML form and store the results:
/peergos-api/v0/form/path.to.file

Send async messages to friends:
/peergos-api/v0/chat/

Files REST HTTP API

- GET - get file or dir
- GET(?preview=true) - get thumbnail
- POST - create file
- PUT - update file
- DELETE - delete file
- PATCH - append to a file

Chat REST API

```
# list all chats created by this app (GET)
* /peergos-api/v0/chat/

# create chat (POST => chatId)
* /peergos-api/v0/chat/

# get messages by local index (GET)
* /peergos-api/v0/chat/:chatId?from=0&to=100

# send message (PUT - text: message)
* /peergos-api/v0/chat/:chatId
```

Concurrent GC

- Kubo GC with 1 TiB S3 blockstore takes ~24 hours whilst holding a global lock
- V1 of Peergos GC takes 2 hours on same size blockstore, fully concurrent (no locks)
- V2 takes 7 mins!

How does writing blocks work?

1. startTransaction => tid
2. write blocks tagged with tid
3. commit new root to mutable pointers
4. closeTransaction(tid)

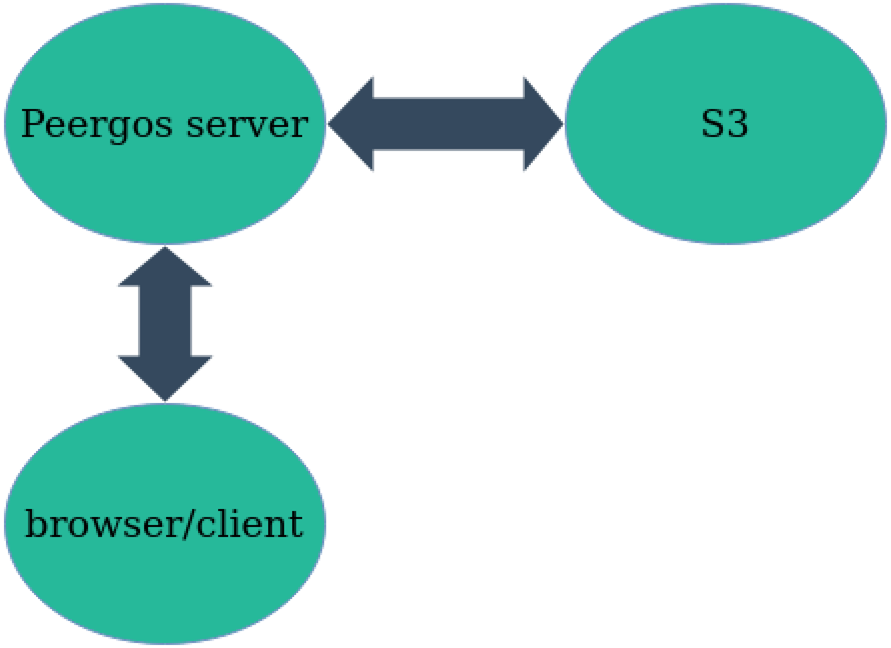
GC algorithm

1. List blockstore
2. List GC roots
3. List uncomitted writes (block writes are tagged with a tid)
4. Mark reachable (parallel) (skip raw blocks)
5. Delete unreachable (parallel)

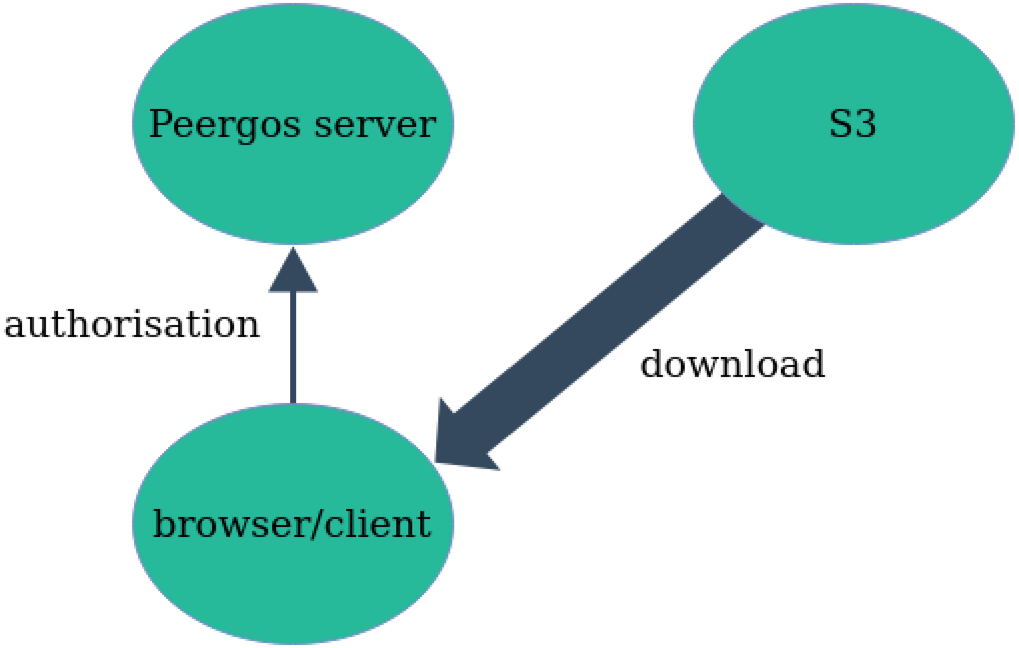
V2 - block metadata

- Store cid => links: list[cid], size in database
- total size ~0.05% of blockstore size
- avoid retrieving and parsing blocks during GC

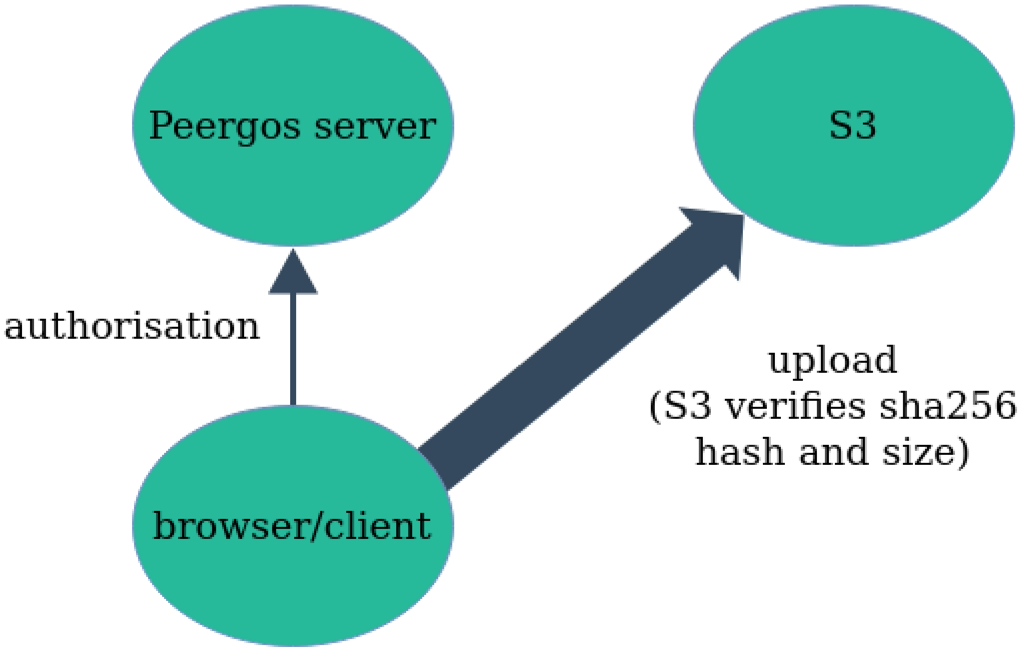
Bandwidth with S3 blockstore



Direct S3 blockstore reads



Direct S3 blockstore writes



What's in a directory?

Semantically a dir is list[children]

- list[cap]
- list[relative cap] ==> fast revocation
- list[named relative cap] ==> fast get-by-path

list[(name, [writer], map key, BAT, read key)]

Build security

- reproducible builds, both server and front end
- Don't use npm! Only 12 JS dependencies, all vendored
- Have our own simple, deterministic replacement for webpack
- self host all assets
- Most of the client code is written in a type-safe language (Java) and cross-compiled to JS