IPFS ÞING
Brussels, Belgium
2023

Content Based Addressing and the Web Security Model

Fabrice Desré
fabrice@capyloon.org

IPFS ÞING
Brussels, Belgium
2023

The Web Security Model

IPFS ÞING
Brussels, Belgium
2023

Same-Origin Policy (SOP) : A set of rules about how browsers behave in cross-origin situations.

"Only the site that stores some information in the browser may later read or modify that information."
(https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.215.6662)

Concretely:

- A site from origin-A can't read pixels of an image loaded from origin-B.
- Scripts run in the origin they are loaded *by*, not the origin they are loaded *from*.
- Fetching content from a different origin with XHR or fetch() is not possible.
- Access to cross-origin iframes requires postMessage().

Protecting your site from compromised dependencies:

- Sub resource integrity (SRI)

```
<script src="https://example.com/example-framework.js"
integrity="sha384-Li9vy3DqF8tnTXuiaAJuML3ky+er10rcgNR/VqsVpcw+ThHmYcwiB1pbOxEbzJr7">
</script>
```

Cross-Origin Resource Sharing (CORS)

In some situation the SOP prevents legitimate uses for cross-origin data.

A typical use case is using an API from a 3rd party site.

CORS allows the site from which the resource is being loaded to make exceptions to the SOP.

For XHR/fetch() and HTTP request methods that can modify data (usually HTTP methods other than GET, or for POST usage with certain MIME types), the browser sends a *preflight* requests to the server in order to learn if the cross-origin load is permitted.

Side Channel Attacks

Even without an explicit API to access it, some information can be deduced by observing secret leaks. Targets are commonly:

- User navigation history.
- Data from other sites, bypassing the SOP.

Some attacks:

- Cache timing (mitigated by double keying of cache entries).
- Computed CSS values of visited links (mitigated, by limiting observable CSS side effects. Details in https://dbaron.org/mozilla/visited-privacy).
- SVG Filters timing! (mitigated by making the algorithm run in constant time).
- … others

IPFS ÞING
Brussels, Belgium
2023

Content Security Policy (CSP)

CSP provides an additional layer of security by allowing a site owner to enforce rules such as:

- Scripts, stylesheet, media, fonts, workers url patterns.
- Navigation patterns.
- Form actions
- Mixed content blocking
- Sandboxing similar to iframes.

When used properly, CSP is a very powerful tool to prevent injection attacks.

Configuration is done by setting the `Content-Security-Policy` HTTP header or the equivalent meta tag.

You can also enable violation reporting, a very useful monitoring tool.

IPFS ÞING
Brussels, Belgium
2023

Overall…

The Web has a fairly robust set of security mechanisms.

Browsers vendors add implementation specific measures (eg. site isolation, OS level sandboxing, ASLR) to mitigate code execution attacks.

The main goal is to protect users, but also to ensure sites are sandboxed from each other.

IPFS ÞING
Brussels, Belgium
2023

Privacy On the Web

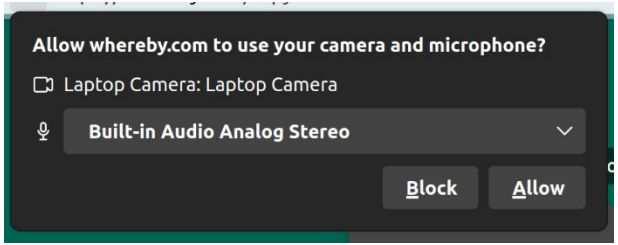
Privacy != Security

You can't have privacy without security, but great security doesn't imply privacy.

Privacy is closely tied to users individual autonomy: ensuring that users can make informed decision on their own consent.

Examples of informed decisions:

- Some browser features (tracking & fingerprinting protection)
- Permissions prompts



IPFS ÞING
Brussels, Belgium
2023

Permission Prompts Limitations

Some powerful/dangerous APIs we need to make the web competitive with "native" can't neither be granted by default, nor by user consent.

"*Let me use TCP sockets*", "*Let me grab a CPU wake lock*", "*Let me read system preferences*"

Different solutions to that problem over the years:

- Firefox OS: grant powerful apis access to signed, packaged code.
- Google Project Fugu: let's hope that user prompts and pickers will be enough.
- Google Isolated Web Apps: oops, actually we need something similar to Firefox OS (but with Web Bundles).

IPFS ÞING
Brussels, Belgium
2023

The root of the problem: establishing trust.

While the full trust chain can be very long (including software toolchains, hardware and supply chain), let's consider we can trust the device, OS and web runtime to work as designed.

The generic threat model still wants to avoid:

1. Malicious code from being served and executed.
2. Private data from being leaked.

Code signing and proper reviews provides 1. but is cumbersome. Using signatures conflates data verification with trust anchoring.

Data leakage is very difficult to guarantee in practice without being "air gapped" by default or playing whac-a-mole with issues.

IPFS ÞING
Brussels, Belgium
2023

A Proposal: Web Tiles

IPFS ÞING
Brussels, Belgium
2023

Web Tiles to the Rescue

A different approach to the permissions and packaging problem. Tiles are:

- **Safe by default**: On top of the regular web safety properties, tiles have additional privacy preserving characteristics enforced by default.
- **Powerful by default**: Since they can't leak data even to the app author, we can grant them more powerful api access. User consent still required in some situations (eg. write access to storage).
- **Content-addressable**: That provides the integrity guarantees we need.
- **Bundled and linkable**: A tile is made of set of resources, all content addressed and linkable through a URL.

IPFS ÞING
Brussels, Belgium
2023

Web Tiles Basics

Technical design overview, reusing many pieces of the dWeb stack:

- A Tile content is exposed through tile://CID/path urls.
- Data leaks are prevented by enforcing by default a very strict CSP to all tile:// documents.
- Tiles can be registered/installed in the user agent. Registration can leverage extensions to the PWA manifest.

IPFS ÞING
Brussels, Belgium
2023

Web Tiles Composability

Tiles are not just standalone apps. Their real super-power comes when they can be combined together or with regular web content.

- Tiles should be activable as capability providers, similarly to intents on Android.
- The set of capabilities need to be defined by the tiles themselves, not by the runtime.
- Tiles should be able to run with or without user visible UI.
- Tiles should be easy to discover, install and share.

IPFS ÞING
Brussels, Belgium
2023

Tiles Prototyping in Capyloon

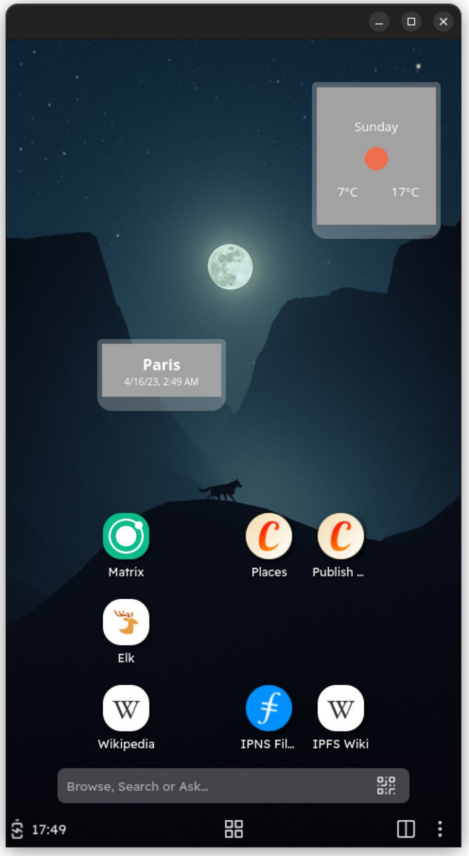
IPFS ÞING
Brussels, Belgium
2023

Capyloon: an experimental Web Based OS

A Firefox OS descendant with dWeb super powers:

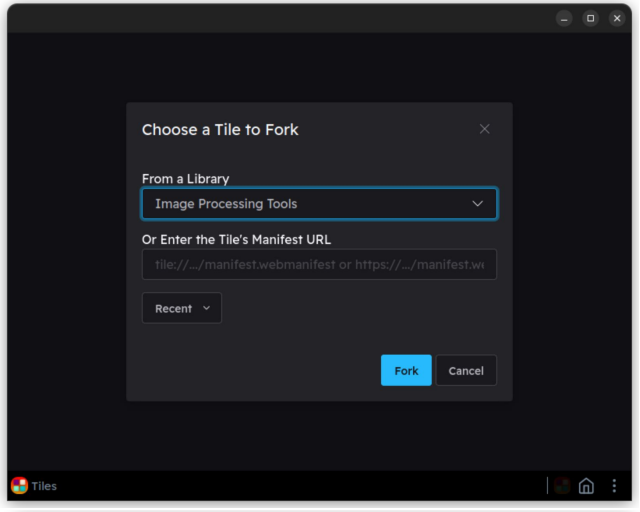
- ipfs:// and ipns:// native protocol handlers backed by a local node (n0's ~~iroh~~ beetle).
- DID and UCANs exploration for permissions.
- Pinning content to Estuary
- QR Code based local-first data exchange.
- Tiles!

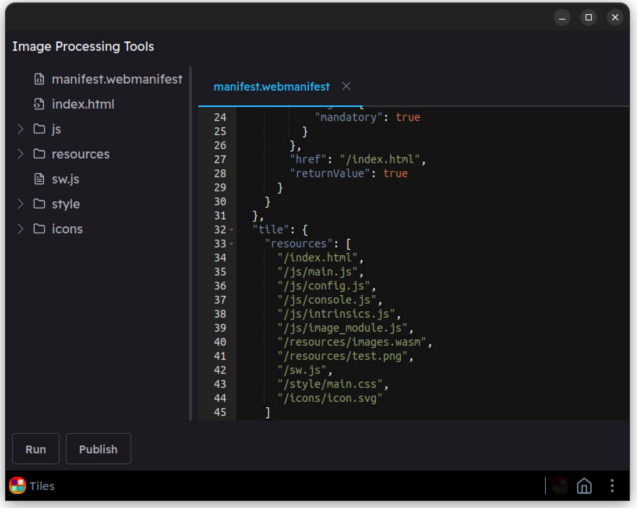
Builds available for some Android based devices and Linux phones, as well as for desktop (Linux/Mac).



IPFS ÞING
Brussels, Belgium
2023

Tiles in Capyloon





Gecko implementation of the tile:// protocol handler, and associated default CSP.

Installation is done like for PWA, with an additional step to download all resources from a Tile at install time and pin them to the IPFS local block store.

The WebActivity support from Capyloon is used as-is for composability: the caller payload is dispatched to a service worker, allowing headless processing or UI based flows.

A Tiles app provides ways to create, fork and publish tiles. (sample tiles: https://github.com/capyloon/tiles)

IPFS ÞING
Brussels, Belgium
2023

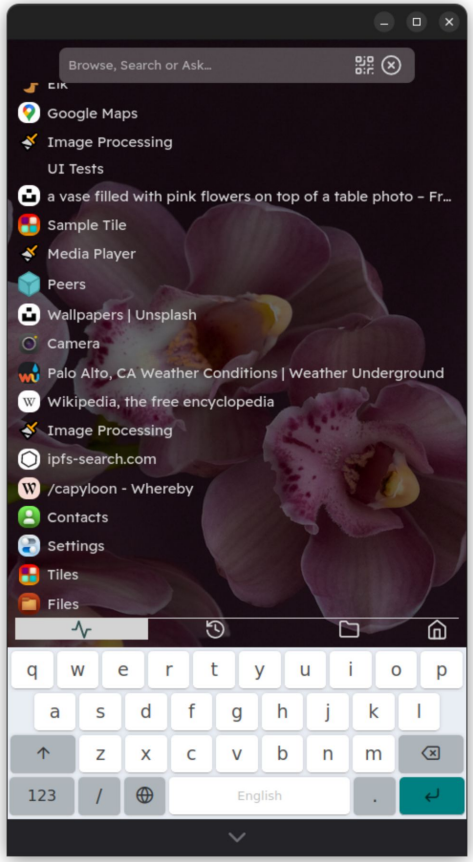
Tiles in Capyloon

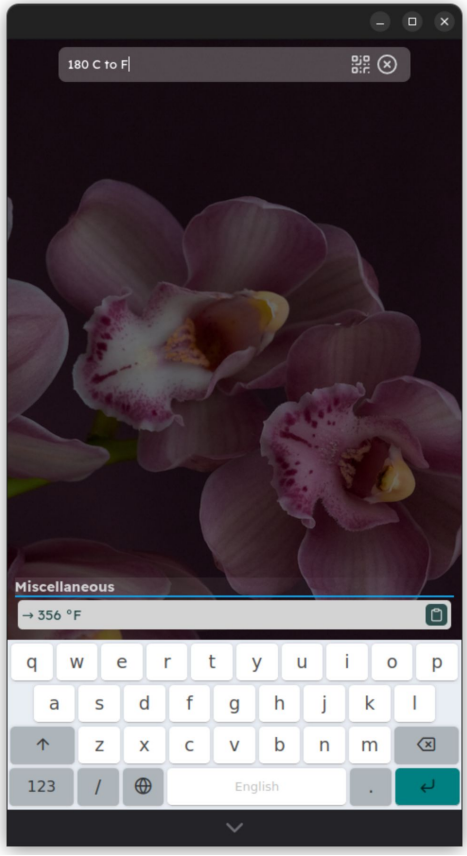
Headless use case: data source for the homescreen.

When entering a request in the homescreen search bar, data is pulled from various sources (history, app installed, contacts, etc.).

A tile can also provide results in a privacy preserving way since it can only work with local data.

Here we use the "fend" universal calculator/convertor, compiled as a WASM module.
(https://github.com/printfn/fend)





IPFS ÞING
Brussels, Belgium
2023

Tiles in Capyloon

Document processing.

Tiles implementing the 'process-image' activity can be selected to edit images from the file manager.

Here also this relies on WASM code for the image processing.

IPFS ÞING
Brussels, Belgium
2023

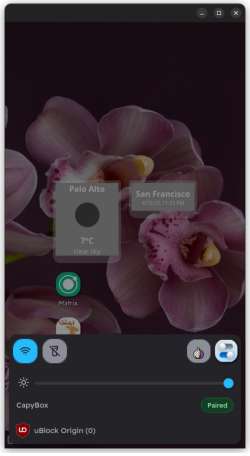
Tiles in Capyloon

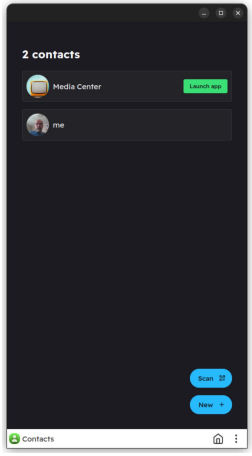
Remote Video Player.

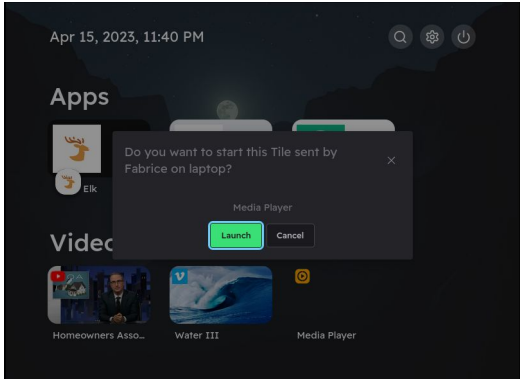
A more complex use case, showing device pairing and contact integration.

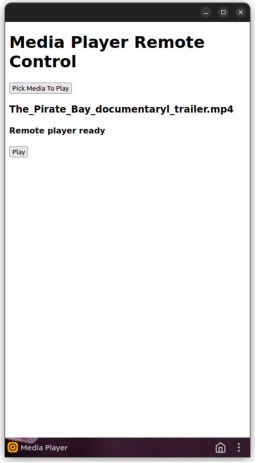
Once the devices are paired:

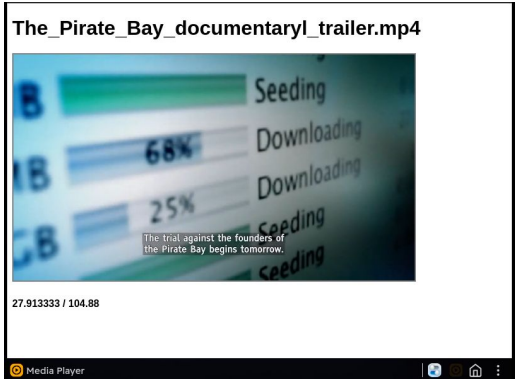
- The initiating device sends the Tile url to the remote device.
- The peers negotiate a webrtc connection with a built-in SDK and can then communicate over a simple data channel.
- In this demo, the initiating device uses Iroh to provide a video file and control the playback.











IPFS ÞING
Brussels, Belgium
2023

Tiles in Capyloon : next steps

More potential use cases…

- Multi player games.
- Content archives (wikipedia, recipes, etc.) with embedded searchable indexes.
- Privacy respecting mashups (eg. mixing gps traces with other on-device activities).
- Social Feed filtering algorithms.
- …

Open questions

- Discovery: centralized / decentralized indexes?
- Tile updates? (stable naming and addressing)
- Relation with CoD efforts in general?
- …

IPFS ÞING
Brussels, Belgium
2023

Thank you!

https://capyloon.org