Browser fingerprinting: current research and the years ahead

Pierre Laperdrix
GT PVP-SSLR
June, 22th 2022
About me

• CNRS researcher in the Spirals team in the UMR CRIStAL in Lille

• Working on web security and privacy: browser fingerprinting, web tracking, history sniffing, application debloating, mobile application security...

• Open positions for internships and PhDs in the team! Don’t hesitate to contact us!
Outline

I. What is browser fingerprinting? How to protect against it?

II. What is currently being done in the fingerprinting domain?

III. What to expect in the future?
I. Internet and web browsers
I. Internet in 1995

Browsers send device-specific information to servers to improve user experience on the web.
What happens when we start collecting all the information available in a web browser?
I. Definition of browser fingerprinting

Definitions

• A browser fingerprint is a set of information related to a user’s device from the hardware to the operating system to the browser and its configuration.

• Browser fingerprinting refers to the process of collecting information through a web browser to build a fingerprint of a device.
I. See your own fingerprint

https://amiunique.org (Am I Unique)

- Website launched in November 2014
- Collected 5,000,000+ fingerprints so far
- Browser extension available to see the evolution of your own browser fingerprint
I. Example of a browser fingerprint

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>User agent</td>
<td>Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/102.0.0.0 Safari/537.36</td>
</tr>
<tr>
<td>HTTP headers</td>
<td>text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,<em>/</em>;q=0.8,application/signed-exchange;v=b3;q=0.9 gzip, deflate, br en-US,en;q=0.9</td>
</tr>
<tr>
<td>Fonts</td>
<td>Century Schoolbook, Source Sans Pro Light, DejaVu Sans Mono, Bitstream Vera Serif, URW Palladio L, Bitstream Vera Sans Mono, Bitstream Vera Sans, ...</td>
</tr>
<tr>
<td>Platform</td>
<td>Win32</td>
</tr>
<tr>
<td>Screen resolution</td>
<td>3840x2160x24</td>
</tr>
<tr>
<td>Timezone</td>
<td>-480 (UTC+8)</td>
</tr>
<tr>
<td>Hardware concurrency</td>
<td>4</td>
</tr>
<tr>
<td>Battery level</td>
<td>38%</td>
</tr>
<tr>
<td>WebGL vendor</td>
<td>NVIDIA Corporation</td>
</tr>
<tr>
<td>WebGL renderer</td>
<td>GeForce GTX 3070 Ti/PCIe/SSE2</td>
</tr>
<tr>
<td>Canvas</td>
<td>Cwm fjordbank glyphs vext quiz, 😊</td>
</tr>
<tr>
<td>Browser extensions</td>
<td>Cwm fjordbank glyphs vext quiz, 😊</td>
</tr>
</tbody>
</table>
I. Impact on privacy

What makes fingerprinting a threat to online privacy?

1. It is really easy to collect all this data. No need for extra permissions.
2. Several studies have investigated the diversity of browser fingerprints.

- 470,161 fingerprints, 94.2% were unique
- 118,934 fingerprints, 89.4% were unique
- 1,816,776 desktop fingerprints, 35.7% were unique

Tracking is possible
I. Protection against fingerprinting

• Goal: to protect users against browser fingerprinting, i.e. to prevent them from being tracked online
I. Protection against fingerprinting - Blocking scripts

• The fingerprinting script is simply not executed.

• Some existing solutions

  Browser extensions

  Browser with built-in protection

<table>
<thead>
<tr>
<th>Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS</td>
</tr>
<tr>
<td>Browser</td>
</tr>
<tr>
<td>JavaScript engine</td>
</tr>
<tr>
<td>Loaded webpage</td>
</tr>
<tr>
<td>Fingerprinting script</td>
</tr>
</tbody>
</table>
I. Protection against fingerprinting - Blocking browser APIs

- The fingerprinting script will collect less information.
- Some existing solutions

CanvasBlocker  Brave  Tor browser

Fingerprinting script
Loaded webpage
JavaScript engine
Browser
OS
Hardware
I. Protection against fingerprinting - Injecting JavaScript

- The injection of JavaScript overwrites the default methods of the JavaScript engine.
- Can change values
  - "navigator.platform"
    - Default: "Win64"
    - New value: "Linux x86_64"
- Can inject noise
I. Protection against fingerprinting - The problem of inconsistencies

My fingerprint

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>User agent</td>
<td>Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/43.2357.124 Safari/537.36</td>
</tr>
<tr>
<td>Accept</td>
<td>text/html,application/xhtml+xml,application/xml;q=0.9,image/webp;v=0.8</td>
</tr>
<tr>
<td>Content encoding</td>
<td>gzip, deflate, br</td>
</tr>
<tr>
<td>Content language</td>
<td>en-US,en;q=0.8</td>
</tr>
<tr>
<td>List of plugins</td>
<td>Plugin 0: Shockwave Flash; Shockwave Flash 21.8.0; NPSWF32_21_0_0_102.dll.</td>
</tr>
<tr>
<td>Platform</td>
<td>MacIntel</td>
</tr>
<tr>
<td>Cookies enabled</td>
<td>yes</td>
</tr>
<tr>
<td>Do Not Track</td>
<td>NC</td>
</tr>
<tr>
<td>Timezone</td>
<td>-60</td>
</tr>
<tr>
<td>Screen resolution</td>
<td>1920x1200x24</td>
</tr>
<tr>
<td>Use of local storage</td>
<td>yes</td>
</tr>
<tr>
<td>Use of session storage</td>
<td>yes</td>
</tr>
<tr>
<td>Canvas</td>
<td>Cwm fjordbank glyphs vext qu</td>
</tr>
<tr>
<td>WebGL Vendor</td>
<td>Not supported</td>
</tr>
<tr>
<td>WebGL Renderer</td>
<td>Not supported</td>
</tr>
<tr>
<td>List of fonts</td>
<td></td>
</tr>
<tr>
<td>Screen resolution</td>
<td>1920x1200</td>
</tr>
<tr>
<td>Language</td>
<td>fr</td>
</tr>
<tr>
<td>Platform</td>
<td>Windows 7</td>
</tr>
<tr>
<td>Use of AdBlock</td>
<td>yes</td>
</tr>
</tbody>
</table>
I. Protection against fingerprinting - Native spoofing

- Instead of injecting JavaScript, the source code of the browser is modified to send new values.
- Investigating JS objects is not enough to detect the modifications.
- Some existing solutions

![Diagram of the fingerprinting process]
I. Protection against fingerprinting - Tor browser and its fingerprint

- In theory, all fingerprints from the Tor Browser should be identical.
- In reality, differences can still be found (screen resolution, fonts, canvas...).
I. Protection against fingerprinting - Changing browsers

• One fingerprint for each browser
• The OS and Hardware layers are shared by both fingerprints.

• If you collect enough information on the OS and hardware, you are prone to cross-browser fingerprinting.
I. Recreating a complete environment

• Disposable environments with a unique fingerprint for each browsing session

• Database with different OS, fonts, plugins and browsers

• Use of virtualization to isolate the host OS from the new environment
I. Protection against fingerprinting

Many different approaches:
• Blocking scripts
• Blocking browser APIs
• Injecting JavaScript
• Native spoofing
• Changing browsers
• Recreating complete environments

Each technique has its strengths and weaknesses.
Outline

I. What is browser fingerprinting? How to protect against it?

II. What is currently being done in the fingerprinting domain?

III. What to expect in the future?
II. Current research – Going beyond browser APIs

To increase the number of attributes in fingerprints, researches are trying to go beyond what’s offered by browser APIs.

• Browser extensions

• Web Audio fingerprinting

• Hardware fingerprinting
Detecting extensions poses a threat to online privacy because:

- The list of installed extensions can reinforce fingerprinting and make user unique on the web.
- It can reveal user’s preferences, browsing habits or demographic information.
II. Current research – Architecture of a browser extension

Structure of a browser extension
- Manifest.json is a mandatory file that provides metadata information on how the extension works.
- Background page implements long-term logic.
- Content scripts are scripts that are injected into visited webpages.
- Web accessible resources (WARs) are files like JS libraries or icons that can be accessed by the extension or any webpage.

Source: MDN Web Docs
II. Current research – Detecting browser extensions

1<sup>st</sup> method: WAR fingerprinting (2017)
- Probes specific WARs in the browser to identify an extension.
- Requires knowledge beforehand of extension IDs and paths of WAR files.
2\textsuperscript{nd} method: Intra/Inter communication fingerprinting (2020)

Extensions as part of their inner-workings exchange messages between components.

```html
<script>
var messages = []
window.addEventListener('message', (event) => {
  data = JSON.stringify(event['data']);
messages.push(data);
});
</script>
```
II. Current research – Detecting browser extensions

3rd method: Behavioral fingerprinting

A) Default behavior: Extensions might add/remove buttons, text or images on a webpage without any interaction (2017).
II. Current research – Detecting browser extensions

B) Style fingerprinting:
Extensions can modify the style of elements on the page (2021).

With the “Super Dark Mode” extension installed
II. Current research – Detecting browser extensions

C) Modifications after user interaction: Extensions modify the page after the user has interacted with it (2022).
Example: key presses, scrolling, mouse clicks

After clicking on the “Mercury Reader” extension button
II. Current research – Going beyond browser APIs

• Browser extensions
  • WAR fingerprinting
  • Intra/Inter communication fingerprinting
  • Behavioral fingerprinting

• Fingerprinting the hardware
Hypothesis: GPUs, even with the exact same model, show differences in their execution.

Finding: We can fingerprint the concurrent behavior of GPUs with a web browser.
II. Current research – Fingerprinting GPUs

- A GPU is composed of several dozens execution units.
- All execution units are not completely identical on a physical level.

Figure 1: Intel® core processor, SoC and its ring interconnect architecture.
II. Current research – Fingerprinting GPUs

How does DrawnApart work:
1) Points are rendered in a WebGL context in parallel by several different execution units.
2) All EUs return directly a single value except EU no. i which executes a stall function that takes time to compute.
3) We measure the time it takes to go through all EUs as each iteration is bounded by the slowest EU.
# II. Current research – Fingerprinting GPUs

<table>
<thead>
<tr>
<th>Accuracy (%)</th>
<th>Base Rate (%)</th>
<th>Device Count</th>
<th>GPU</th>
<th>Device Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>93.0±0.3</td>
<td>10.0</td>
<td>10</td>
<td>Intel HD Graphics 2500</td>
<td>Intel i5-3470</td>
</tr>
<tr>
<td>63.7±0.6</td>
<td>4.3</td>
<td>23</td>
<td>Intel HD Graphics 4600</td>
<td>Intel i5-4590</td>
</tr>
<tr>
<td>55.5±0.8</td>
<td>6.7</td>
<td>15</td>
<td>Intel UHD Graphics 630</td>
<td>Intel i5-8500</td>
</tr>
<tr>
<td>95.8±0.9</td>
<td>10.0</td>
<td>10</td>
<td>Nvidia GTX1650</td>
<td>Intel i5-10500</td>
</tr>
<tr>
<td>73.1±0.7</td>
<td>25.0</td>
<td>4</td>
<td>Apple M1</td>
<td>Apple Mac Mini M1</td>
</tr>
<tr>
<td>36.7±2.7</td>
<td>16.7</td>
<td>6</td>
<td>Mali-G71 MP20</td>
<td>+Samsung Galaxy S8/S8</td>
</tr>
<tr>
<td>54.3±5.5</td>
<td>16.7</td>
<td>6</td>
<td>Mali-G72 MP18</td>
<td>+Samsung Galaxy S9/S9</td>
</tr>
<tr>
<td>54.1±1.5</td>
<td>12.5</td>
<td>8</td>
<td>Mali-G76 MP12</td>
<td>Samsung Galaxy +S10e/S10/S10</td>
</tr>
<tr>
<td>92.7±1.8</td>
<td>16.7</td>
<td>6</td>
<td>Mali-G77 MP11</td>
<td>Samsung Galaxy S20/S20 Ultra</td>
</tr>
</tbody>
</table>

**Results:**
- Some GPUs are easier to identify than others with a varying accuracy.
- We tested swapping CPUs from two identical computers and DrawnApart was able to identify the swap.

[https://github.com/drawnapart/drawnapart](https://github.com/drawnapart/drawnapart)
II. Current research – Detecting fingerprinting

Detecting fingerprinting scripts on the Internet is more complicated than it seems.

If a script accesses the user agent or the timezone, is it to optimize the browsing experience? Or is it the first step towards building a browser fingerprint?

Several approaches have been tried over the years from static to dynamic analysis. Depending on the definition of fingerprinting used in a paper, the results can greatly vary: from 2% of websites using fingerprinting on the web to more than 60% for the least conservative.
II. Current research – Using fingerprinting positively

Browser fingerprinting can be used positively to improve security:

• To reinforce authentication

  ![Diagram of login and fingerprint](image)

  Login/Password + Browser fingerprint = User authenticated

• To combat bots

  ![Diagram of fingerprinting and device classes](image)

Google uses canvas fingerprinting to detect classes of device and identify emulation.
II. Current research

To sum up:

• Going beyond browser APIs to fingerprint the hardware

• Detecting usage of browser fingerprinting

• Using fingerprinting positively to improve security
Outline

I. What is browser fingerprinting? How to protect against it?

II. What is currently being done in the fingerprinting domain?

III. What to expect in the future?
On the user’s side, different solutions are being actively developed to protect against fingerprinting:

- **Tor browser** (since 2007): the goal is to remove as much as possible the differences between users. All users in theory should have the same fingerprint.

- **Brave browser** (since 2016): several APIs have been modified to protect against fingerprinting and Brave is the only one randomizing some attributes (“farbling”).

- **Firefox** (since 2017): block fingerprinting scripts present in specific filter lists.
Chrome browser (in 2024): Google is developing the Privacy budget which will limit the quantity of collected information.

1) As long as the script has some budget: APIs can be accessed without restriction.

2) When the budget expires: specific APIs will be blocked or will provide very limited information.
II. Future – Evolution of the EU legislation

• Right now, it is mandatory to ask the user before collecting a fingerprint but....no one is doing it?

• GDPR: General Data Protection Regulation
  • New set of rules that governs how data from EU citizens are collected and handled around the world.
  • It requires companies to be transparent on how they handle data.
  • Went into effect on May 25th 2018

• ePrivacy regulation
  • Successor of the cookie law
  • Requires consent to perform fingerprinting (exception for analytics from first-party servers)

One major problem: there is no built-in mechanism dedicated to fingerprinting
III. The years ahead – Evolution of the ad landscape

- There is a strong push for privacy preserving solutions for online tracking.
- Google is removing support for 3rd party cookies in late 2023 and it is already having a great impact on the ad industry.
- Two different directions are being adopted:

  - Use of « people IDs » in place of cookie IDs
  - Use of a mechanism to hide one user among many
    1) “FLoC” by Google
    2) “Privacy Preserving Ad Click Attribution For the Web” from the WebKit team
    3) “PARAKEET” from Microsoft
    4) “TURTLEDOVE” by Google

Where does browser fingerprinting fit into all this?
Thank you!
Stay safe online!
Any questions?