Hiding in the Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale

Alejandro Gómez-Boix, Pierre Laperdrix, Benoit Baudry

Hiding in the Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale

Alejandro Gómez-Boix
Univ Rennes, Inria, CNRS, IRISA
Rennes, France
alejandro.gomez-boix@inria.fr

Pierre Laperdrix
Univ Rennes, Inria, CNRS, IRISA
Rennes, France
pierre.laperdrix@inria.fr

Benoit Baudry
KTH Royal Institute of Technology
Stockholm, Sweden
baudry@kth.se

ABSTRACT

Browser fingerprinting is a stateless technique, which consists in collecting a wide range of data about a device through browser APIs. Past studies have demonstrated that modern devices present so much diversity that fingerprints can be exploited to identify and track users online. With this work, we want to evaluate if browser fingerprinting is still effective at uniquely identifying a large group of users when analyzing millions of fingerprints over a few months.

We analyze 2,067,942 browser fingerprints collected from one of the top 15 French websites. The observations made on this novel dataset shed a new light on the ever-growing browser fingerprinting domain. The key insight is that the percentage of unique fingerprints in this dataset is much lower than what was reported in the past: only 33.6% of fingerprints are unique by opposition to over 80% in previous studies. We show that non-unique fingerprints tend to be fragile. If some features of the fingerprint change, it is very probable that the fingerprint will become unique. We also confirm that the current evolution of web technologies is benefiting users' privacy significantly as the removal of plugins brings down substantively the rate of unique desktop machines.

KEYWORDS

browser fingerprinting; privacy; software diversity

ACM Reference Format:
Alejandro Gómez-Boix, Pierre Laperdrix, and Benoit Baudry. 2018. Hiding in the Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale. In *WWW 2018: The 2018 Web Conference, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3178876.3186097

1 INTRODUCTION

Web browsers share device-specific information with servers to improve online user experience. When a web browser requests a webpage from a server, by knowing the platform or the screen resolution, the server can adapt its response to take full advantage of the capabilities of each device. In 2010, through the data collected by the Panopticlick website, Eckersley showed that this information is so diverse and stable that it can be used to build what is called *a browser fingerprint* to track users online [15]. By collecting information from HTTP headers, JavaScript and installed plugins, he was able to uniquely identify most of the browsers. With the gathered data, Eckersley not only showed that there exists an incredible diversity of devices around the world but he highlighted that this very same diversity could be used as an identification mechanism on the web. Since this study, researchers have looked at new ways to collect even more information [13, 14, 18, 24–26, 32, 34, 36], measure the adoption of these techniques on the Internet [10, 11, 16, 29], propose defense mechanisms [12, 17, 19–21, 28], and track devices over long periods of time [37]. In 2016, a study conducted by Laperdrix et al. [22] with the AmIUnique website confirmed Eckersley's findings. The authors noted a shift in the most discriminating attributes with the addition of new APIs like Canvas and the progressive removal of browser plugins. They also demonstrated that fingerprinting mobile devices is possible, but with a lower degree of success.

Tracking users with fingerprinting is a reality. If a device presents the slightest difference compared to other ones, it can be identified and followed on different websites. While Panopticlick and AmIUnique proved that tracking is possible, one problem arises when looking at both datasets: their bias. First, both websites are dedicated to fingerprinting, people who visited them are interested in the topic of online tracking. It limits the scope of their studies. Then, looking at the general statistics page of the AmIUnique website from July 2017, we can clearly see a bias as 57% of visitors are on Windows, 15% on Linux, 13% on Mac, 5% on Android and 4% on iOS. The latest statistics from StatCounter for the month of July 2017 reveal that the OS market share is dominated by Android with a percentage around 40%, followed by Windows at 36%, iOS at 13%, Mac at 5% and Linux under 1% [6]. One can then ponder about the impact of such a big difference on the effectiveness of browser fingerprinting.

In this paper, we investigate whether tracking can be extended to websites that target a broad audience. We analyze 2,067,942 fingerprints collected from one of the top 15 French websites, and we investigate whether browser fingerprinting techniques are still effective in identifying users by collecting the same attributes reported in the literature. Our first two research questions are related to this issue:

RQ 1. How uniquely identifiable are the fingerprints in our data?

RQ 2. Can non-unique fingerprints become unique if some value changes?

The other questions are related to the characteristics of the dataset and the possible impact of the evolution of web technologies:

RQ 3. Can the circumstances under which fingerprints are collected affect the obtained results?

WWW 2018, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5639-8/18/04.
https://doi.org/10.1145/3178876.3186097

RQ 4. Does the evolution of web technologies limit the effectiveness of browser fingerprinting?

Where previous studies reported having above 80% of unique fingerprints, we obtained a surprising number: 33.6% of unique fingerprints. This gap can be explained by the targeted audience as our study looks at fingerprints collected from the global population and not necessarily biased towards users interested in online privacy. The difference is even more noticeable when looking at the 251,166 fingerprints coming from mobile devices. 18.5% of them are unique which is in direct contradiction with the 81% that has been observed by Laperdrix et al. [22]. These results show another aspect of browser fingerprinting and its tracking capabilities with the current evolution of web technologies. Here, we extend the analyses carried out by Eckersley [15] and Laperdrix et al. [22] by putting the browser fingerprinting domain under a different light.

Our key contributions are:

- We explore the current state of browser fingerprinting with the analysis of 2,067,942 fingerprints composed of 17 different attributes. We also provide the first large-scale study of JavaScript font probing and we measure its real-life effectiveness.
- We show that by collecting these attributes and targeting a much broader audience, browser fingerprinting is not as effective as it was reported in the literature. While previous studies reported having above 80% of unique fingerprints, we obtained 33.6%.
- We compare our dataset with the ones from Panopticlick and AmIUnique and we explain in details the numerous differences that can be observed.
- We provide a discussion on the future of browser fingerprinting and what these results mean for the domain and for future applications of this technique.

The paper is organized as follows. Section 2 introduces our new dataset along with the ones from Panopticlick and AmIUnique. Section 3 analyzes the diversity of browser fingerprints in our data and compares the three datasets by providing detailed statistics to help explain the differences. Section 4 discusses the impact of our results on the domain and we simulate possible technical evolutions to have an insight on future applications of this technique. Finally, Section 5 concludes this paper.

2 DATASET

This section introduces the three different datasets that form the basis of the comparison in the next section. First, we give a short description of the two available sets of browser fingerprint statistics conducted on a large scale. Then, we describe the attributes collected to form the browser fingerprints analyzed here.

2.1 Previous studies

2.1.1 Panopticlick. In 2010, Peter Eckersley launched the Panopticlick website with the goal of collecting device-specific information via a script that runs in the browser [15]. The script collected values for 10 different web browser features and its execution platform. Features were collected from three different sources: HTTP protocol, JavaScript and Flash API. Eckersley collected 470,161 fingerprints from January 27th to February 15th, 2010. Data obtained by Panopticlick is "representative of the population of Internet users who pay enough attention to privacy" [15], so in this sense the data is quite biased. In the study performed by Eckersley, the list of fonts (collected through the Flash API) and the list of plugins (collected via JavaScript) were the most distinguishable attributes.

2.1.2 AmIUnique. With the aim of performing an in-depth analysis of web browser fingerprints, the AmIUnique website was launched in November 2014. Collected fingerprints are composed of 17 features (among them, those proposed by Eckersley [15]). These fingerprints include recent technologies, such as the HTML5 canvas element and the WebGL API. In the study conducted by Laperdrix et al. [22], 118,934 fingerprints collected between November 2014 and February 2015 were analyzed. The authors validated Eckersley's findings with Panopticlick and provided the first extensive analysis of fingerprints collected from mobile devices. Data collected on this website is biased towards users who care about privacy and their digital footprint.

2.2 The dataset

The fingerprints used in this study have been collected through a script deployed in collaboration with the b<>com Institute of Research and Technology (IRT) on one of the top 15 French websites (according to the Alexa traffic rank) on two specific web pages: a weather forecast page and a political news page. The script ran for a six month period, from December 7th, 2016 to June 7th, 2017. To be compliant with the European directives 2002/58/CE and 2009/136/CE, and with the French data protection authority (CNIL), only visitors who consented to the use of cookies, and thus the use of fingerprinting techniques, were fingerprinted. When users first connect to one of these two pages, we set up a 6-months long cookie in their browser. This supports the identification of returning visitors.

Compared to the other two detailed studies, the website used to collect this dataset covers a wide range of topics and it is not dedicated to browser fingerprinting. According to the Hawthorne effect [23], if individuals are aware that they are being studied, a type of reaction occurs, in which individuals modify an aspect of their behavior in response to their awareness of being observed. In our case, this means that the fingerprints in this dataset are more representative of those found in the wild, since users are not enticed to play with their browsers to change their configuration and produce different fingerprints.

2.2.1 Fingerprinted attributes. In order to compare ourselves with previous studies, we rely on the same attributes found in the study conducted by Laperdrix et al. in 2016 [22]. The complete list of attributes is given in the 'Attribute' column of Table 2. However, to reflect recent technological trends, we made the following modifications to our script:

List of fonts. Fonts are usually collected through the Flash plugin. With a few lines of code, one can get access to the entire list of fonts installed on the user's system. However, because of security and stability reasons, plugins are being deprecated in modern browsers in favor of a feature-rich HTML5 environment [33]. Flash is expected to disappear definitely as Adobe announced the end-of-life of its solution for 2020 [4]. All major web browsers like Chrome,

TimesNEWRoman
TimesNEWRoman

Figure 1: Difference between Tinos (top) and Times New Roman (bottom).

Firefox, Edge and Safari already block Flash content or have removed support for it. This means that fingerprinting scripts must turn to another mechanism to get access to the list of fonts.

Nikiforakis et al. revealed that it is possible to probe for the existence of fonts through JavaScript [29]. A script can ask to render a string with a specific font in a *div* element. If the font is present on the device, the browser will use it. If not, the browser will use what is called a fallback font. By measuring the dimensions of the *div* element, one can know if the demanded font is used or if the fallback font took its place. The biggest difference between these two gathering methods is that fonts through JavaScript must be checked individually whereas Flash gives all the installed fonts in a single instruction. This means that testing a large number of fonts is time consuming and can delay the loading of a web page. For this reason, we chose to test 66 different fonts, some among the most popular 'web-safe fonts' which are found in most operating systems and other less common ones. Appendix A reports on the complete list of fonts we tested in our script.

Before deploying our script in production, we identified a limitation in how JavaScript font probing operates. We found out that some fonts can have the exact same dimensions as the ones from the fallback font. Figure 1 illustrates this problem. In the example, the two tested fonts are metrically comparable and have the exact same width and height. However, they are not identical as it can be seen in the shapes of some of the letters (especially "e", "a" and "w"). This means that font probing here will report incorrect results if one were to ask Times New Roman on a system with the Tinos font installed (or vice versa). To fix this problem, we measured the dimensions of a *div* against three font style variants. There are different typefaces that can be used by a web browser with the most popular ones being *serif*, *sans-serif*, *monospace*, *cursive* and *fantasy*. We chose the first three and we tested each font against the three of them, resulting in $66 * 3 = 198$ different tests. This way, we avoid reporting false negatives as the three fallback fonts have different dimensions.

Canvas. The Canvas API allows for scriptable rendering of 2D shapes and texts in the browser. Discovered by Mowery et al. [25], investigated by Acar et al. [10], and then collected on a large scale by Laperdrix et al. [22], canvas fingerprinting can be used to differentiate devices with pixel precision by rendering a specific picture following a set of instructions. In order to see how far we can go with this technique, we took as a basis the canvas test performed by Laperdrix et al. [22] and we made a more complex canvas element by combining new elements of different natures. First, the script asks the browser to render the two following strings: "*Yxskaftbud, ge vår WC-zonmö IQ-hjälp*" and "*Gud hjälpe Zorns mö qvickt få byxa*". Both strings are *pangrams* (a string with all the letters of the alphabet) of the Swedish alphabet. For the first string, we force the browser to use one of its fallback fonts by asking for a font with a fake name. Depending on the OS and the fonts installed on the device, fallback fonts may differ from one user to another. For the second line, the browser is asked to use the **Arial** font that is common in many operating systems. Then, we ask for additional strings with symbols and emojis. All strings, with the addition of a rectangle are drawn with a specific rotation. A second set of elements is rendered with four mathematical functions: a sine, a cosine and two linear functions. These functions are plotted on a specific interval and using the **PI** value of the JavaScript Math library as a parameter. The third set of elements consists in drawing a set of ellipsis. These figures are drawn with different colors and with different levels of transparency. Since filters for opacity change among browsers, it creates differences between them. The last element is a centered shadow that overlaps the canvas element. Figure 2 displays an example of a canvas rendering following the instructions of our script.

Figure 2: Example of a rendered picture following the canvas fingerprinting test instructions.

Cookies. Since we only have fingerprints from users who accepted the use of cookies, all fingerprints have the exact same value for this attribute.

2.2.2 Descriptive statistics. We distinguish two different kinds of fingerprints: those belonging to mobile devices and those belonging to desktop and laptop machines (we will refer to desktop and laptop machines as personal computers). To prevent collecting multiple copies of the same fingerprint from the same user, we store a cookie on the user's device with a unique ID for six months. Among the 2,067,942 fingerprints, the distinction is as follows: 1,816,764 come from personal computers (87.9% of the data), and the rest, 251,190 fingerprints come from mobile devices (12.1% of the data).

Table 1: OS market share distribution.

OS	Our data	AmIUnique Nov'14-Jul'17 [22]	StatCounter Jul'17 [6]
Windows	93.5%	63.7%	84%
MacOS	5.5%	14.9%	11%
Linux	0.9%	16.9%	1.8%
Android	72%	55.6%	70%
iOS	18.8%	42.3%	22%
Windows Phone	7.6%	<1%	1%

Table 1 reports on the distribution of operating systems in both our dataset and the one from the AmIUnique website. Statistics gathered from StatCounter for the month of July 2017 have also been added to give an idea how close they are from the global population. First, by looking at the differences between our newly collected data and AmIUnique, we can see that there is a significant

difference in terms of distribution. Notably, we can see a clear bias in the demographic that AmIUnique attracted since the percentage of Linux desktop machines is much higher than the reported by StatCounter. Then, if we compare our numbers with the ones from StatCounter, we can see that we provide a closer representation of the global population as the percentages for both distributions are close to each other.

Table 2 summarizes the essential descriptive statistics of our dataset. The 'Distinct values' column provides the number of different values that we observed for each attribute, while the 'Unique values' column provides the number of values that occurred a single time in our dataset. For example, the *Use of local/session storage* attribute has no unique values since it is limited to "yes" and "no". Moreover, in our data, all users accepted the use of cookies, so all the fingerprints have "yes" for this attribute. Other attributes can take a high number of values. For example, we observed 6,618 unique values for the list of fonts. In fact, we also know the higher bound for the number of distinct values for this attribute. We perform in total $66 * 3$ tests and each one can take the value 'true' or 'false'. These results in 2^{66*3} possible combinations even if, in practice, many of them will not be found.

3 ANALYSIS AND COMPARISON

In this section, we first analyze how diverse browser fingerprints are in our dataset. Then, we analyze the level of identifying information of each attribute that makes up the fingerprint. Finally, we compare our dataset with the two available sets of fingerprint statistics, provided by Eckersley in 2010 [15] and Laperdrix et al. in 2016 [22].

3.1 Browser fingerprint diversity

Our data was collected on a much larger scale than previous studies and targeting a much broader audience, which leads to the **RQ 1. How uniquely identifiable are fingerprints in our data?** This question aims at determining how diverse the browser fingerprints are in this novel dataset. Using attributes from Table 2, we succeeded in uniquely identifying 33.6% of fingerprints in our dataset. On personal computers, 35.7% of fingerprints are unique while this number is lower on mobile devices with 18.5%. On personal computers, the threat is less important than reported in other studies. On mobile devices, the number is much smaller but the threat comes from elsewhere: closed platforms with integrated tracking applications.

Figure 3 represents the distribution of the anonymity sets. A set represents a group of fingerprints with identical values for all the collected attributes. If a fingerprint is in a set of size 1, it means that this fingerprint is unique and it can be identified. On mobile devices, the percentages of fingerprints belonging to sets of size larger than 50 is around 59%, while on personal computers this percentage is around 8%. It means that the number of devices sharing equal fingerprints on mobile devices is larger than on personal computers. This can be explained by the fact that the software and hardware environments of these devices are much more constrained than on desktop and laptop machines. Users buy very specific models of smartphones that are shared by many. The largest set of mobile devices contains 13,241 fingerprints, while for personal computers it contains 1,394 fingerprints.

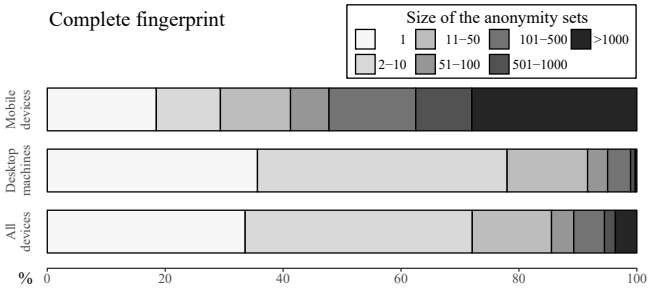

Figure 3: Comparison of anonymity set sizes between mobile devices and desktop/laptop machines.

Low rates of success at uniquely identifying browser fingerprints in our data reveal that, by collecting more than two million browser fingerprints on a commercial website, it is very unlikely that a fingerprint is unique and hence exploitable for tracking. Possibilities for a fingerprint to be unique are three times lower than in the previous datasets collected for research purposes (Panopticlick and AmIUnique).

3.1.1 Unique fingerprints. There are 46,459 unique fingerprints on mobile devices and 647,741 on personal computers. A fingerprint is unique due to one of the following reasons:

- It has an attribute whose value is only present once in the whole dataset.
- The combination of all its attributes is unique in the whole dataset.

On mobile devices, 73 % of fingerprints are unique because they contain a unique value, while this percentage is around 35% for personal computers.

While mobile fingerprints tend to be unique because of their unique values, laptop/desktop fingerprints tend to have combinations of values so diverse that they create unique fingerprints. The most distinctive attributes are *canvas* on mobile devices and *plugins* on personal computers. Fingerprints with unique canvas values represent 62% of unique fingerprints on mobile devices, while on personal computers, fingerprints with unique combinations of plugins represent 30% of unique fingerprints.

3.1.2 Investigating changes on browser fingerprints. Over the course of its lifetime, a device exhibits different fingerprints. This comes from the fact web technologies are constantly evolving and thus, web browser components are continually updated. From the operating system to the browser and its components, one single update can change the exhibited browser fingerprint. For instance, a new browser version is directly reflected by a change in the user-agent. A plugin update is noticeable by a change in the list of plugins. When web browsers evolve naturally, changes happen automatically without any user intervention and this affects all users.

Natural evolution of web technologies is not the only reason why fingerprints evolve. There are some parameters that usually are the choice of the users, such as the use of cookies, the presence of the "Do Not Track" header, or the activation of specific plugins. Users are allowed to change these values at anytime. Besides, some attributes such as timezone or fonts are indirectly impacted by a

Table 2: Browser measurements for the data.

Attribute	Dataset		Mobile devices		Personal computers	
	Distinct values	Unique values	Distinct values	Unique values	Distinct values	Unique values
User-agent	19,775	8,702	10,949	5,424	8,826	3,278
Header-accept	24	9	9	2	19	8
Content encoding	30	8	19	5	25	4
Content language	2,739	1,313	961	529	2,128	958
List of plugins	288,740	196,898	81	33	288,715	196,882
Cookies enabled	1	0	1	0	1	0
Use of local/session storage	2	0	2	0	2	0
Timezone	60	16	39	1	58	18
Screen resolution and color depth	2,971	1,015	434	159	2675	897
Available fonts	17,372	6,618	94	36	17,326	6,603
List of HTTP headers	610	229	158	78	491	164
Platform	32	5	21	2	26	3
Do Not Track	3	0	3	0	3	0
Canvas	78,037	65,787	30,884	28,768	47,492	37,194
WebGL Vendor	27	1	20	2	26	3
WebGL Renderer	3,691	657	95	10	3,656	661
Use of an ad blocker	2	0	2	0	2	0

change in the environment, such as traveling to a different timezone or adding fonts (fonts can be added intentionally or come as a side effect of installing new software on a device). Which gives rise to the **RQ 2. Can non-unique fingerprints become unique if some value changes?** and specifically, do non-unique fingerprints become unique if only one value changes?

Let us take as an example of a user with a non-unique fingerprint. Running Chrome 55 on Windows 10, the browser displays the following value for the *Content language* header:

```
fr-FR,fr;q=0.8,en-US;q=0.6,en;q=0.4
```

For some reason, the user decides to add the Spanish language. The browser then displays the following value:

```
fr-FR,fr;q=0.8,en-US;q=0.6,en;q=0.4,es;q=0.2
```

By changing the language settings, does the fingerprint become unique?

In order to answer this type of question and to study how resilient non-unique fingerprints are in the face of evolution, we conducted an experiment. We looked at analyzing the impact made by the user's choice on the uniqueness of their fingerprints. There is a set of attributes whose values cannot be changed such as attributes related to the hardware and software environment on which the browser is running. The *Platform* attribute is linked to the operating system, while *WebGLVendor* and *WebGLRenderer* reveal information about the GPU. Attributes such as *User-agent*, *List of HTTP headers* or *Content encoding* are beyond the control of the user because they are related to the HTTP protocol. However, attributes such as *Cookies enabled*, *Do Not Track*, *Content language* and *List of plugins* are a direct reflection of the user's choice. Nevertheless, *Cookies enabled*, *Do Not Track*, *Use of local/session storage* are limited to "yes" and "no", so they do not offer a very discriminant information. This leaves the *Content language*, *List of plugins*, *Available fonts* and *Timezone* under the scope of our analysis.

For the experiment, we chose fingerprints belonging to sets larger than 50 fingerprints. New values were chosen randomly from non-unique fingerprints that had the same operating system and web browser (including versions). This was made to ensure that the new values are consistent with fingerprints that can be found in the wild. This way, we avoid choosing values that are not characteristic

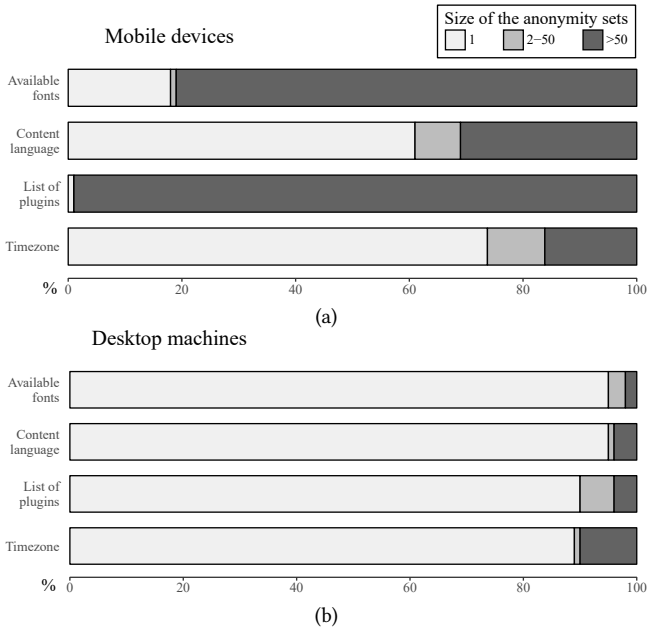

Figure 4: Anonymity sets resulting of changing values randomly in sets larger than 50 fingerprints on mobile devices (a) and Personal computers (b).

of the fingerprint environment. For example, two browsers can have the same language configuration, but the encoding is different depending on the web browser. Example:

```
Windows 8.1, Chrome,
        fr-FR,fr;q=0.8,en-US;q=0.6,en;q=0.4
Windows 8.1, Firefox,
        fr-FR,fr;q=0.8,en-US;q=0.5,en;q=0.3
```

Both browsers are running on the same operating system and have the same language configuration: French/France[fr-FR], French[fr], English/United States[en-US] and English[en]. But, depending on the web browser, the final language headers are different.

Results. The experiment was repeated ten times and results were averaged. Figure 4 represents the distribution of the anonymity sets resulting of randomly changed values for the *Content language*, *List of plugins*, *Available fonts* and *Timezone* on mobile devices and desktop/laptop machines. First, we can clearly notice an important difference between devices. For desktop/laptop machines, more than 85% of fingerprints turned into unique fingerprints. This is due to the fact that combinations of values tend to be so diverse on personal computers that they make up unique fingerprints. On mobile devices, when changing the *Available fonts* and the *List of plugins*, over 80% of fingerprints remained in large sets. These results are explained by the absence of diversity in these attributes on mobile devices. Results are very different for *Content language* and *Timezone*: over 60% of fingerprints turned into unique fingerprints. This can be explained by the lack of diversity in these attributes. As most users share the same timezone and languages, a single change on one of these two attributes dramatically increases the likelihood of the fingerprint to become unique.

By looking at the results of the experiment, we can conclude that if one single feature of a fingerprint changes, it is very probable that this fingerprint becomes unique. In the end, desktop/laptop fingerprints tend to be much more fragile than their mobile counterparts.

3.2 Comparison of attributes

Mathematical treatment. We used entropy to quantify the level of identifying information in a fingerprint. The higher the entropy is, the more unique and identifiable a fingerprint will be. Let H be the entropy, X a discrete random variable with possible values $x_1; ...; x_n$ and $P(X)$ a probability mass function. The entropy follows this equation:

$$H(X) = -\sum_{i=0}^{n} P(x_i) log_b P(x_i) \quad (1)$$

We use the entropy of Shannon where $b = 2$ and the result is expressed in bits. One bit of entropy reduces by half the probability of an event occurring. In order to compare all three datasets which are of different sizes, we applied the *Normalized Shannon's entropy*:

$$\frac{H(X)}{H_M} \quad (2)$$

H_M represents the worst case scenario where the entropy is maximum and all values of an attribute are unique ($H_M = log_2(N)$ with N being the number of fingerprints in our dataset).

The advantage of this measure is that it does not depend on the size of the anonymity set but on the distribution of probabilities. We are quantifying the quality of our dataset with respect to an attribute uniqueness independently from the number of fingerprints in our database. This way, we can qualitatively compare the datasets despite their different sizes.

Table 3 lists the Shannon's entropy for all attributes from both the Panopticlick and AmIUnique studies, and our dataset. Column 'Entropy' shows the bits of entropy and column 'Norm.' shows the normalized Shannon's entropy. The last two rows of Table 3 show the worst case scenario where the entropy is maximum (i.e. all the values are unique) and the total number of fingerprints. In the novel dataset analyzed here, the most distinctive attributes are the *List of plugins*, the *Canvas*, the *User-agent* and the *Available fonts*.

Due to differences in software and hardware architecture between mobile devices and personal computers, we computed entropy values separately. By comparing entropy values between mobile devices and personal computers, we observed three attributes where the difference is significant.

The largest difference is for the *List of plugins* with a difference of 0.485 for the normalized entropy. It can be explained by the lack of plugins on mobile devices as web browsers on mobile devices take full advantage of functionalities offered by HTML5 and JavaScript. On personal computers, the *List of plugins* is the most discriminant attribute while it is almost insignificant for mobile devices. We can observe in Table 2 that among 251,166 fingerprints coming from mobile devices, there are only 81 distinct values for plugins. The second significant difference is 0.214 for the *Available fonts*. Installing fonts on mobile devices is much more restrained than on personal computers. Even if we test a very limited set of fonts through JavaScript compared to what could be collected through Flash, we can see that there is clearly more diversity on personal computers. The last significant difference is for the *User-agent* attribute, with a difference of 0.182. On mobile devices, the user-agent has the highest entropy value. This is because phone manufacturers include the model of their phone and even sometimes the version of firmware directly in the user-agent as revealed by Laperdrix et al. [22].

Attributes *Use of an ad blocker* and *Use of local/session storage* have very low entropy values because their values are either "yes" or "no".

We also tested the impact of compressing a canvas rendering to the JPEG format. It should be noted that the JPEG compression comes directly from the Canvas API and is not applied after collection. Due to the lossy compression, it should come as no surprise that the entropy from JPEG images is lower than the PNG one usually used by canvas fingerprinting tests (from 0.407 to 0.391).

In the study realized by Eckersley [15], the analysis of browser fingerprints was performed without differentiating between mobile and desktop fingerprints. Later, some researchers conducted studies about browser tracking mechanisms either on desktop machines [10, 13] or on mobile devices [35, 38] but not on both. In 2016, Laperdrix et al. [22] provided the first extensive study about browser fingerprinting on mobile devices, they proved that both kind of devices presented different discriminating attributes. If the analysis of browser fingerprinting is not carried out by differentiating mobile devices from personal computers, results obtained will not be representative of both kinds of devices. In our data, 12.1% of fingerprints belong to mobile devices so mobile fingerprints represent a small part of the entire data. If we take a look at Table 3, entropy values for attributes like *List of plugins* or *Available fonts* are largely influenced by the group that contains the majority of fingerprints which, in our case, is the one with personal computers. For future work, it is strongly recommended to differentiate mobile devices from personal computers (laptops and desktop machines) to obtain more accurate results.

Table 3: Shannon's entropy for all attributes from Panopticlick, AmIUnique and our data.

	Panopticlick		AmIUnique		Dataset		Mobile devices		Desktop/laptop machines	
Attribute	**Entropy**	**Norm.**	**Entropy**	**Norm.**	**Entropy**	**Norm.**	**Entropy**	**Norm.**	**Entropy**	**Norm.**
Platform	-	-	2.310	0.137	1.200	0.057	2.274	0.127	0.489	0.024
Do Not Track	-	-	0.944	0.056	1.919	0.091	1.102	0.061	1.922	0.092
Timezone	3.040	0.161	3.338	0.198	0.164	0.008	0.551	0.031	0.096	0.005
List of plugins	15.400	0.817	11.060	0.656	9.485	0.452	0.206	0.011	10.281	0.494
Use of local/session storage	-	-	0.405	0.024	0.043	0.002	0.056	0.003	0.042	0.002
Use of an ad blocker	-	-	0.995	0.059	0.045	0.002	0.067	0.004	0.042	0.002
WebGL Vendor	-	-	2.141	0.127	2.282	0.109	2.423	0.135	1.820	0.088
WebGL Renderer	-	-	3.406	0.202	5.541	0.264	4.172	0.233	5.278	0.254
Available fonts	13.900	0.738	8.379	0.497	6.904	0.329	2.192	0.122	6.967	0.335
Canvas	-	-	8.278	0.491	8.546	0.407	7.930	0.442	8.043	0.387
Header Accept	-	-	1.383	0.082	0.729	0.035	0.111	0.006	0.776	0.037
Content encoding	-	-	1.534	0.091	0.382	0.018	1.168	0.065	0.153	0.007
Content language	-	-	5.918	0.351	2.716	0.129	2.291	0.128	2.559	0.123
User-agent	10.000	0.531	9.779	0.580	7.150	0.341	8.740	0.487	6.323	0.304
Screen resolution	4.830	0.256	4.889	0.290	4.847	0.231	3.603	0.201	4.437	0.213
List of HTTP headers	-	-	4.198	0.249	1.783	0.085	1.941	0.108	1.521	0.073
Cookies enabled	0.353	0.019	0.253	0.015	0.000	0.000	0.000	0.000	0.000	0.000
H_M (worst scenario)	18.843		16.860		20.980		17.938		20.793	
Number of FPs	470,161		118,934		2,067,942		251,166		1,816,776	

3.3 Comparison with Panopticlick and AmIunique

In the data collected by Panopticlick, Eckersley observed that 83% of visitors had instantaneously recognizable fingerprints. This number reached 94% for devices with Flash or Java installed. With the AmIUnique website, Laperdrix and colleagues observed that 89.4% of fingerprints from their dataset were unique. Thanks to the high percentages of unique browser fingerprints, browser fingerprinting established itself as an effective stateless tracking technique on the web.

However, with our study, we provide an additional layer of understanding in the fingerprinting domain. By having 33.6% of unique fingerprints compared to the 80+% of the other two studies, we show that browser fingerprinting may not be effective at a very large scale and that the targeted audience plays an important role in its effectiveness.

3.3.1 Comparing data size. When analyzing the percentages of unique fingerprints, the amount of fingerprints is an important element that influences the results. As discussed by Eckersley in [15], the probability of any fingerprint to be unique in a sample of size N is $1/N$. It is clear that probabilities of being unique in our dataset are much lower than the probabilities of being unique in the AmIUnique one.

With the aim of establishing a more equitable comparison, we took some samples with the same number of fingerprints as the AmIUnique data and we then calculated the percentage of unique fingerprints. We perform a comparison with the AmIUnique data because the amount of fingerprints is four times smaller than the one collected by Panopticlick. Because our dataset spans a six month period, we divided the data into six parts, each part containing data for one month. We kept the same proportion between mobile devices and desktop machines as the AmIUnique data, so we randomly took 105,829 desktop/laptop fingerprints and 13,105 mobile fingerprints from each month. Results were averaged.

On average, 56% of personal computers are unique, while 29% of mobile devices are unique. These percentages show that low ratios of unique fingerprints are influenced by the number of fingerprints. Even so, results obtained on the sample are significantly distant from those obtained by Laperdrix et al. [22]. These results show that performing tracking with fingerprinting is possible, yet difficult.

3.3.2 Comparing entropy values. Comparison with Panopticlick can be established by taking into account only six attributes. We observe that entropy values for our dataset and Panopticlick differ significantly for all attributes, except for the *Screen resolution*. Entropy values for the *Screen resolution* attribute hardly change for the three datasets.

Regarding *Timezone* and *Cookies enabled*, drops occur in entropy values due to the characteristics of our dataset. As we explained in Section 2, we analyze fingerprints from users who accepted cookies, and most of them live in the same geographic region. The difference in the entropy value for *Content language* is due to the fact that most users are located in the same geographic region, which implies that most of them share the same language. In fact, 98% of users present the same value for timezone, which corresponds to Central European Time Zone UTC+01:00 and as a direct consequence of this, 97.7% of fingerprints present French as their first language.

The noticeable drop in the entropy values for the *Timezone* and *Content language* affects the fingerprint diversity. To a great extent, the lack of diversity in any attribute has a direct impact on the fingerprint diversity. By decreasing the amount of values that an attributes can take, the identifying value of the attribute is reduced and therefore the identifying value of the browser fingerprint decreases. It means that the diversity surface is reduced, which reduces the diversity among the browser fingerprints giving as result less identifiable browser fingerprints.

For the *List of plugins*, it is still the most discriminating attribute but a gradual decrease can be observed. From Panopticlick to AmIUnique, a difference of 0.24 is present. From AmIUnique to our dataset, the difference is 0.126 resulting in a decrease of 0.365 from Panopticlick to our data. This gradual decrease in the entropy value for the *List of plugins* is explained by the absence of plugins on mobile devices and by the removal of plugins from modern browsers. Over time, features have been added in HTML5 to replace plugins as they were considered a source of many security problems. Chrome

stopped supporting the old NPAPI plugin architecture on Chrome in 2015 (topic discussed in [22]). Mozilla dropped support in version 52 of the Firefox browser released in March 2017. Safari has never supported plugins, Flash is long discontinued for Android, and MS Edge for Windows 10 does not support most plugins. Anything else reliant on the Netscape Plugin API (NPAPI) is now dropped which means Silverlight, Java and Acrobat are gone [27].

The difference in the entropy value for *Available fonts* between Panopticlick and AmIUnique is explained by Laperdrix et al. [22]. Half of the fingerprints in the AmIUnique dataset were collected on browsers that do not have the Flash plugin installed or activated. Between AmIUnique and our data, the difference for the entropy value of fonts is 0.117. Even if collecting fonts through JavaScript is not as effective as with Flash, we observe that the entropy of fonts is still high, keeping its place as one of the top distinctive attributes. For the other attributes, we observe that the entropy values for both our dataset and AmIUnique are similar.

4 DISCUSSION

In this section, we discuss our results along with the potential implications on the browser fingerprinting domain.

4.1 The impact of different demographics

In Section 3, we compared our dataset with the two available sets of fingerprint statistics. There are two key elements that can influence the results of this analysis: the targeted audience and the evolution of web technologies. Previous datasets were collected through websites dedicated to browser fingerprint collection. Both websites `amiunique.org` and `panopticlick.eff.org` inform users about online fingerprint tracking, so users who visit these websites are aware of online privacy, interested in the topic or might be more cautious than the average web user. Our dataset is much different as it was collected by targeting a general audience through a commercial website. We believe that this difference in the fingerprint collection process is key to explain the differences between datasets, giving rise to **RQ 3. Can the circumstances under which fingerprints are collected affect the obtained results?**

Web technologies affect browser fingerprinting. The fact that some technologies are no longer used leads to the evolution of fingerprinting techniques. In some cases, it leads to a decrease in the identifying value of certain features, as we noticed in the attributes *List of plugins* and *Available fonts*. As a result of the progressive disappearance of plugins, the *List of plugins* is rapidly losing its identifying value.

In addition to the effects produced by the evolution of technologies, there are some issues resulting from the collection process. Some of them are caused by targeting a specific demographic group. For instance, the market share distribution across the planet is not uniform. According to StatCounter [6] in 2017, the European mobile market was led by Apple and Samsung, with similar participation percentages above 30%. Although the mobile market in North America is also led by Apple and Samsung, Apple represents about 50% of the market, while Samsung about 24%. If we collect a sample of mobile fingerprints from North America, there is a good chance that the sample will have a greater presence of Apple devices. So, the distribution of some features like the *Platform*, *WebGL Vendor* or *User-agent* will be more representative of Apple devices.

In the end, depending on the website, the use case or the targeted demographic, the results can greatly vary and the effectiveness of browser fingerprinting can change. Moreover, if we were to perform a similar study targeted at different countries, can we expect the same diversity of fingerprints? Do more developed countries have access to a wider range of devices and, as a consequence, present a larger set of fingerprints? Do more educated users have a tendency to specialize and configure more their devices which, as a result, would make their fingerprints more unique? From data that we gathered, it is impossible to answer these questions as we do not collect information beyond what is presented by the user's device. Yet, considering these different facets may be the key to understand the extent to which browser fingerprinting can work for tracking and identification. Its actual effectiveness is much more nuanced that what was reported in the past and it is far from being an answer to a simple yes or no question.

4.2 Towards a potential privacy-aware fingerprinting

An arms race is currently developing between users and third-parties. As people are getting educated on the questions of tracking and privacy on the web, more and more users are installing browser extensions to protect their daily browsing activities. At the end of 2016, 11% of the global Internet population is blocking ads on the web [8]. This represents 615 million devices with an observed 30% growth in a single year. With regards to browser fingerprinting, several browsers already include protection to defend against it. Pale Moon [1], Brave [3] and the Tor Browser [7] were the very first ones to add barriers against techniques like Canvas or WebGL fingerprinting. Mozilla is also currently adding its own fingerprinting protection in Firefox [2] as part of the Tor Uplift program [9]. With our study, we show that we do not know yet the full extent of what is possible with browser fingerprinting and as so, modern browsers are getting equipped with mitigation techniques that require a lot of development to integrate and maintain. But, **Does the evolution of web technologies limit the effectiveness of browser fingerprinting?**

In order to answer the **RQ 4**, we follow the idea proposed by Laperdrix et al. [22]. The authors simulated the effectiveness of browser fingerprinting against possible technical evolutions. We recreated some of their scenarios on our dataset.

Scenario n°1 - The end of browser plugins. Web browsers are evolving to an architecture not based on plugins. Despite the progressive disappearance of plugins, the list of plugins is still the most distinctive attribute for personal computers in our data. A glimpse of the impact of this scenario is observed on mobile devices, although some plugins still remain. To estimate the impact of the disappearance of plugins, we simulate the fact that they are all the same in our dataset, but only on personal computers and thus taking mobile devices as reference. The improvement is significant with a decrease of exactly 19.2% from 35.7% to 16.5%, taking slightly lower value than on mobile devices, which is 18.5%. Disappearance

of plugins for personal computers reduces significantly the effectiveness of browser fingerprinting at uniquely identifying users, as we observed first on mobile web browsers.

Scenario n°2 - Adherence to the standard HTTP headers. Laperdrix et al. [22] simulated this scenario assuming that the HTTP header fields had the same value for all fingerprints. We followed this idea, and in addition to that, we reduced the identifying information of the user-agent, just by keeping the name and version of the operating system and the web browser. On personal computers, the improvement is moderate with a decrease of 4.7% from 35.7% to 31% in overall uniqueness. However, on mobile fingerprints, we can observe a drop less significant of 2.3% from 18.5% to 16.2%. In the simulation of this scenario, Laperdrix et al. [22] obtained significant results compared with our results. The small drop is due to the low entropy value of the HTTP headers in our data of 0.085 compared with the value obtained by [22] of 0.249. Another element to consider is that we included a piece of information contained in the user-agent, illustrating that the combination of operating system and web browser still includes some diversity.

Scenario n°3 - The end of JavaScript. By using only features collected through JavaScript (equivalent to remove HTTP features), it is possible to uniquely identify 28.3% of personal computers and 14.3% of mobile devices. By removing all features collected through JavaScript, fingerprint uniqueness drastically drops. On mobile devices, the percentage drops by 14.2% from 18.5% to 4.3%. On personal computers, the drop is abrupt from 35.7% to 0.7%. The improvement in privacy by removing JavaScript is highly visible, but the cost to the ease and comfort of using web services could be overly high.

These findings show that the evolution of web technologies can benefit privacy with a limited impact. While some of them are becoming a reality, others are more improbable.

Yet, it is possible to envision a future where a "privacy-aware" form of fingerprinting is possible, i.e. one that does not enable identification but that can still provide the security benefits touched upon in the literature. First, the W3C has put privacy at the forefront of discussions when designing new APIs. In 2015, Olejnik et al. performed a privacy analysis of the Battery Status API [30]. They found out that the level of charge of the battery could be used as a short-term identifier across websites. Because of this study, this API has been removed from browsers several years after its inclusion [31] and it changed the way new APIs are making their way inside our browsers. A W3C draft has even been written on how to mitigate browser fingerprinting directly in web specifications [5]. This shows how important privacy is going forward and we can expect in the future that new APIs will not reveal any identifying information on the user's device. Then, looking at our own dataset, a privacy-aware fingerprinting seems achievable thanks to the low percentages of unique fingerprints we present in this study.

5 CONCLUSION

In this work, we analyzed 2,067,942 browser fingerprints collected through a script that was launched on one of the top 15 French websites. Our work focuses on determining if fingerprinting is still possible at a large scale. Our findings show that current fingerprinting techniques do not provide effective mechanisms to uniquely identify users belonging to a specific demographic region as 33.6% of collected fingerprints were unique in our dataset. Compared to other large scale studies on browser fingerprinting, this number is two to three times lower. This difference is even larger when only considering mobile devices as 18.5% of mobile fingerprints are unique compared to the 81% from [22].

The other key elements from our study are as follows. Personal computers and mobile devices have unique fingerprints that are composed differently. While desktop/laptop fingerprints are unique mostly because of their unique combinations of attributes, mobile devices present attributes that have unique values across our whole dataset. We show that by changing some features of the fingerprint, such as *Content language* or *Timezone*, it is very probable that the fingerprint will become unique. We also show that *User-agent* and HTML5 canvas fingerprinting play an essential role in identifying browsers on mobile devices, meanwhile the *List of plugins* is the most distinctive elements on personal computers, followed by the HTML5 canvas element. Furthermore, in the absence of the Flash plugin to provide the list of fonts, we used an alternative for collecting fonts through JavaScript. Even if the list of tested fonts is much smaller compared to what could be captured through Flash, collecting fonts through JavaScript still presents some good results to distinguish two devices from each other.

We also discussed some of the elements that can change the effectiveness of browser fingerprinting, such as the targeted demographic and the existing web technologies. Finally, we analyze the impact of current trends in web technologies. We show that the latest changes in fingerprinting techniques have benefited users' privacy significantly, i.e. the end of browser plugins is bringing down substantively the rate of uniqueness among desktop/laptop fingerprints.

ACKNOWLEDGMENT

We thank the b<>com Institute of Research and Technology (IRT) for their support and we are particularly grateful to Alexandre Garel for his collaboration in setting up the script on the commercial website and collecting the data. This work is partially supported by the CominLabs-PROFILE project and by the Wallenberg Autonomous Systems Program (WASP).

APPENDIX A. LIST OF TESTED FONTS

Andale Mono, AppleGothic, Arial, Arial Black, Arial Hebrew, Arial MT,Arial Narrow, Arial Rounded MT Bold, Arial Unicode MS, Bitstream Vera Sans Mono, Book Antiqua, Bookman Old Style, Calibri,Cambria, Cambria Math, Century, Century Gothic, Century Schoolbook, Comic Sans, Comic Sans MS, Consolas, Courier, Courier New, Garamond, Geneva, Georgia, Helvetica, Helvetica Neue, Impact, Lucida Bright, Lucida Calligraphy, Lucida Console, Lucida Fax, LUCIDA GRANDE, Lucida Handwriting, Lucida Sans, Lucida Sans Typewriter, Lucida Sans Unicode, Microsoft Sans Serif, Monaco, Monotype Corsiva, MS Gothic, MS Outlook, MS PGothic, MS Reference Sans Serif, MS Sans Serif, MS Serif, MYRIAD, MYRIAD PRO, Palatino, Palatino Linotype, Segoe Print, Segoe Script, Segoe UI, Segoe UI Light, Segoe UI Semibold, Segoe UI Symbol, Tahoma, Times, Times New Roman, Times New Roman PS, Trebuchet MS, Verdana, Wingdings, Wingdings 2, Wingdings 3

REFERENCES

[1] 2015. Pale Moon browser - Version 25.6.0 adds a canvas poisoning feature. (2015). https://www.palemoon.org/releasenotes.shtml.

[2] 2017. Fingerprinting protection in Firefox as part of the Tor Uplift Project – Mozilla Wiki. (2017). https://wiki.mozilla.org/Security/Fingerprinting.

[3] 2017. Fingerprinting Protection Mode – Brave browser. (2017). https://github.com/brave/browser-laptop/wiki/Fingerprinting-Protection-Mode.

[4] 2017. Flash & The Future of Interactive Content – Adobe. (2017). https://blogs.adobe.com/conversations/2017/07/adobe-flash-update.html.

[5] 2017. Mitigating Browser Fingerprinting in Web Specifications – W3C Draft. (2017). https://w3c.github.io/fingerprinting-guidance/.

[6] 2017. Operating System Market Share Worldwide – StatCounter. (2017). http://gs.statcounter.com/os-market-share.

[7] 2017. The Design and Implementation of the Tor Browser [DRAFT] "Cross-Origin Fingerprinting Unlinkability" – Tor Project Official website. (2017). https://www.torproject.org/projects/torbrowser/design/#fingerprinting-linkability.

[8] 2017. The state of the blocked web - 2017 Global Adblock Report by PageFair. (2017). https://pagefair.com/downloads/2017/01/PageFair-2017-Adblock-Report.pdf.

[9] 2017. Tor Uplift Project – Mozilla Wiki. (2017). https://wiki.mozilla.org/Security/Tor$_{U}$plift.

[10] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. 2014. The Web Never Forgets: Persistent Tracking Mechanisms in the Wild. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS '14)*. ACM, New York, NY, USA, 674–689. https://doi.org/10.1145/2660267.2660347

[11] Gunes Acar, Marc Juarez, Nick Nikiforakis, Claudia Diaz, Seda Gürses, Frank Piessens, and Bart Preneel. 2013. FPDetective: dusting the web for fingerprinters. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security (CCS '13)*. ACM, New York, NY, USA, 1129–1140. https://doi.org/10.1145/2508859.2516674

[12] Peter Baumann, Stefan Katzenbeisser, Martin Stopczynski, and Erik Tews. 2016. Disguised Chromium Browser: Robust Browser, Flash and Canvas Fingerprinting Protection. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society (WPES '16)*. ACM, New York, NY, USA, 37–46. https://doi.org/10.1145/2994620.2994621

[13] Károly Boda, Ádám Máté Földes, Gábor György Gulyás, and Sándor Imre. 2012. *User Tracking on the Web via Cross-Browser Fingerprinting*. Lecture Notes in Computer Science, Vol. 7161. Springer Berlin Heidelberg, Berlin, Heidelberg, 31–46. https://doi.org/10.1007/978-3-642-29615-4$_{4}$

[14] Yinzhi Cao, Song Li, and Erik Wijmans. 2017. (Cross-)Browser Fingerprinting via OS and Hardware Level Features. In *24nd Annual Network and Distributed System Security Symposium, NDSS*.

[15] Peter Eckersley. 2010. How Unique is Your Web Browser?. In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies (PETS'10)*. Springer-Verlag, Berlin, Heidelberg, 1–18. http://dl.acm.org/citation.cfm?id=1881151.1881152

[16] Steven Englehardt and Arvind Narayanan. 2016. Online Tracking: A 1-million-site Measurement and Analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*. ACM, New York, NY, USA, 1388–1401. https://doi.org/10.1145/2976749.2978313

[17] Amin FaizKhademi, Mohammad Zulkernine, and Komminist Weldemariam. 2015. FPGuard: Detection and Prevention of Browser Fingerprinting. In *Data and Applications Security and Privacy XXIX*. Lecture Notes in Computer Science, Vol. 9149. Springer International Publishing, 293–308. https://doi.org/10.1007/978-3-319-20810-7$_{2}$1

[18] David Fifield and Serge Egelman. 2015. Fingerprinting web users through font metrics. In *Proceedings of the 19th international conference on Financial Cryptography and Data Security*. Springer-Verlag, Berlin, Heidelberg.

[19] Ugo Fiore, Aniello Castiglione, Alfredo De Santis, and Francesco Palmieri. 2014. Countering Browser Fingerprinting Techniques: Constructing a Fake Profile with Google Chrome. In *Network-Based Information Systems (NBiS), 2014 17th International Conference on*. IEEE, 355–360.

[20] Pierre Laperdrix, Benoit Baudry, and Vikas Mishra. 2017. FPRandom: Randomizing core browser objects to break advanced device fingerprinting techniques. In *9th International Symposium on Engineering Secure Software and Systems (ESSoS 2017)*. Bonn, Germany. https://hal.inria.fr/hal-01527580

[21] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. 2015. Mitigating browser fingerprint tracking: multi-level reconfiguration and diversification. In *10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS 2015)*. Firenze, Italy. https://hal.inria.fr/hal-01121108

[22] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. 2016. Beauty and the Beast: Diverting modern web browsers to build unique browser fingerprints. In *37th IEEE Symposium on Security and Privacy (S&P 2016)*. San Jose, United States. https://hal.inria.fr/hal-01285470

[23] Rob McCarney, James Warner, Steve Iliffe, Robbert Van Haselen, Mark Griffin, and Peter Fisher. 2007. The Hawthorne Effect: a randomised, controlled trial. *BMC medical research methodology* 7, 1 (2007), 30.

[24] Keaton Mowery, Dillon Bogenreif, Scott Yilek, and Hovav Shacham. 2011. Fingerprinting Information in JavaScript Implementations. In *Proceedings of W2SP 2011*, Helen Wang (Ed.). IEEE Computer Society.

[25] Keaton Mowery and Hovav Shacham. 2012. Pixel Perfect: Fingerprinting Canvas in HTML5. In *Proceedings of W2SP 2012*, Matt Fredrikson (Ed.). IEEE Computer Society.

[26] Martin Mulazzani, Philipp Reschl, Markus Huber, Manuel Leithner, Sebastian Schrittwieser, Edgar Weippl, and FH Campus Wien. 2013. Fast and reliable browser identification with javascript engine fingerprinting. In *Web 2.0 Workshop on Security and Privacy (W2SP)*, Vol. 5.

[27] Mozilla Developer Network and individual contributors. 2017. Firefox 52 for developers. (2017). https://developer.mozilla.org/en-US/Firefox/Releases/52

[28] Nick Nikiforakis, Wouter Joosen, and Benjamin Livshits. 2015. PriVaricator: Deceiving Fingerprinters with Little White Lies. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 820–830. https://doi.org/10.1145/2736277.2741090

[29] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2013. Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy (SP '13)*. IEEE Computer Society, Washington, DC, USA, 541–555. https://doi.org/10.1109/SP.2013.43

[30] Łukasz Olejnik, Gunes Acar, Claude Castelluccia, and Claudia Diaz. 2016. *The Leaking Battery*. Springer International Publishing, Cham, 254–263. https://doi.org/10.1007/978-3-319-29883-2$_{1}$8

[31] Lukasz Olejnik, Steven Englehardt, and Arvind Narayanan. 2017. Battery Status Not Included: Assessing Privacy in Web Standards. In *3rd International Workshop on Privacy Engineering (IWPE'17)*. San Jose, United States.

[32] Iskander Sanchez-Rola, Igor Santos, and Davide Balzarotti. 2017. Extension Breakdown: Security Analysis of Browsers Extension Resources Control Policies. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/sanchez-rola

[33] J. Schuh. 2013. Saying Goodbye to Our Old Friend NPAPI. (September 2013). https://blog.chromium.org/2013/09/saying-goodbye-to-our-old-friend-npapi.html.

[34] Alexander Sjösten, Steven Van Acker, and Andrei Sabelfeld. 2017. Discovering Browser Extensions via Web Accessible Resources. In *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy (CODASPY '17)*. ACM, New York, NY, USA, 329–336. https://doi.org/10.1145/3029806.3029820

[35] Jan Spooren, Davy Preuveneers, and Wouter Joosen. 2015. Mobile Device Fingerprinting Considered Harmful for Risk-based Authentication. In *Proceedings of the Eighth European Workshop on System Security (EuroSec '15)*. ACM, New York, NY, USA, Article 6, 6 pages. https://doi.org/10.1145/2751323.2751329

[36] Oleksii Starov and Nick Nikiforakis. 2017. XHOUND: Quantifying the Fingerprintability of Browser Extensions. In *38th IEEE Symposium on Security and Privacy (S&P 2017)*. San Jose, United States.

[37] Antoine Vastel, Pierre Laperdrix, Walter Rudametkin, and Romain Rouvoy. 2018. FP-STALKER: Tracking Browser Fingerprint Evolutions. In *39th IEEE Symposium on Security and Privacy (S&P 2018)*. San Fransisco, United States.

[38] W. Wu, J. Wu, Y. Wang, Z. Ling, and M. Yang. 2016. Efficient Fingerprinting-Based Android Device Identification With Zero-Permission Identifiers. *IEEE Access* 4 (2016), 8073–8083. https://doi.org/10.1109/ACCESS.2016.2626395