Searching for Indicators of Device Fingerprinting in the JavaScript Code of Popular Websites

Michael Rausch
Mathematics and Computer Science Department
Ripon College
Ripon, WI 54971
rauschm@ripon.edu

Nathan Good
Good Research
Berkeley, CA 94720
nathan.good@gmail.com

Chris Jay Hoofnagle
School of Law
Berkeley Center for Law & Technology
University of California, Berkeley
Berkeley, CA 94720
choofnagle@law.berkeley.edu

Abstract

Over the years a number of studies have investigated online tracking using cookies. Individuals and organizations are becoming aware of this form of tracking and are taking steps to protect their privacy by deleting and blocking cookies. In response, some companies have developed a form of tracking known as device fingerprinting that does not rely on cookies to identify a user. Fingerprinting could potentially override a user's attempts to prevent tracking.

In this paper we attempt to determine the prevalence of device fingerprinting by crawling the 1000 most popular sites on the internet in the United States as ranked by Quantcast searching for JavaScript code that could be used to fingerprint devices. We identified characteristics of code used to track devices from companies known to engage in device fingerprinting and counted how many sites used similar code. We found that less than 6% of websites surveyed contained such code.

1. INTRODUCTION

1.1 Overview and Background

For years, companies have sought to uniquely identify internet users for various reasons, including fraud prevention, analytics or advertising. Regardless of the intentions of the company, consumers have different privacy preferences, and some do not wish to be tracked at all, or wish not to be tracked when engaging in certain kinds of internet use.

Often companies will attempt to track individual internet users so personally tailored advertisements may be displayed that match the user's interests. According to Mookie Tenembaum, who founded Virtual Realities, "All advertisers, websites and networks use cookies for targeted advertising," [14]. By presenting advertisements related to a user's past behavior a company hopes to facilitate a higher conversion rate on the advertisement, leading to increased sales and profits. The identification process can be conducted through the use of third-party cookies. Many studies related to cookie-based tracking have been conducted, including analyses of methods and censuses [3][9]. The general public and media are also aware of cookies [15].

As the public became aware of this form of tracking, various techniques were developed to limit the effectiveness of third party cookies as a tracking technology. Internet users can now configure many popular browsers to reject third-party cookies, and many users periodically delete their cookies, frustrating a tracker's attempt to build useful profiles. Safari is configured to block third party cookies by default [16].

Recognizing how fragile this form of tracking has become, some companies implemented a new form of internet tracking technology: device fingerprinting [2]. Device fingerprinting does not rely on cookies to identify internet users. Instead a profile of the user may be created by querying the browser for information such as the user agent, screen dimensions of the device, fonts and plugins installed. Oracle's documentation states "Device fingerprinting data may be gathered from multiple sources including secure cookie, flash shared object, user agent string, custom agent, mobile application, browser header data. The intelligent identification does not rely on any single attribute type so it can function on user devices not following strict specifications" [5]. Kount, a tracking company, states that device fingerprinting "…thoroughly examines any device via numerous attributes…" and states "While device type may vary, the following characteristics are typically examined: Network, SSL, Javascript, Browser, Operating System, Flash, HTTP" [8].

These device fingerprinting companies won the attention of the popular media. Sensational claims were published. Mark Douglas was reported as saying that "it [device fingerprinting] can completely replace the use of cookies" in a Wall Street Journal article [2]. Forbes published an article on device fingerprinting with the headline "The Web Cookie is Dying" [13] suggesting that device fingerprinting would supplant cookies. Academics interested in privacy also began to study device fingerprinting [6][9].

Though others in the past have conducted research into the ways fingerprinting may be accomplished, a census investigating the prevalence of device fingerprinting on popular websites had not been published at the time this project was conducted. This census aims to give researchers and the public a deeper understanding of the potential impact device fingerprinting has on the internet.

1.2 Related Work

People have been interested in the concept of remotely identifying devices based on their system configuration for many years. Device fingerprinting captured the interest of the academic community at least as early as 2005, when Kohno, Broido and Claffy conducted an investigation into device fingerprinting using clock skews [7]. They found that it was possible to identify a device remotely regardless of the distance separating the identifier and the system. They realized that their method could be used to potentially track devices connected to the internet without the consent of the owners of the device.

Eckersley wrote a paper in 2010 that analyzed the effectiveness of device fingerprinting by attempting to fingerprint over 400,000 browsers. This was accomplished through the Panopticlick project [6]. Panopticlick is a website created by the Electronic Freedom Foundation to promote awareness of device fingerprinting. On Panopticlick visitors are fingerprinted and information that was gathered about their system via JavaScript and Flash is displayed. At the end of the project Eckersley found that over 80% of browsers detected had a unique fingerprint. He thought that the sampling of browsers in his study was not a good representation of the general population using the internet since only those interested in privacy would visit Panopticlick.

In 2013 Nikiforakis et al. published a paper studying how device fingerprinting worked on the internet [9]. They identified specific fingerprinting scripts and analyzed them, searching for the techniques (including JavaScript calls) that could be used to fingerprint devices.

After we conducted our research in the summer of 2013 Acer et al. published a paper on a fingerprinting census that they had conducted. They conducted a crawl of one million websites looking for evidence of JavaScript and Flash based device fingerprinting. They found that 97 of the top 10,000 most popular websites employed Flash-based fingerprinting [1].

Device fingerprinting is (perhaps surprisingly) quite resilient. Even if the user's system changes (for example through software updates) an algorithm can detect the change and correctly match the user to his or her original fingerprint, with a high degree of success [6]. If a user attempts to avoid being fingerprinted by reporting false information to the fingerprinting script their unusual behavior can make them even easier to fingerprint [9].

2. METHODS

We conducted the research in three phases: a search for the JavaScript code used by device fingerprinters, a crawl to find JavaScript code from popular websites, and finally a search of the collected code for scripts that could be used for fingerprinting.

Search for calls used for fingerprinting

Before searching for the prevalence of device fingerprinting on the web we needed to have an understanding of how commercial device fingerprinting was carried out. We sought to understand the techniques used by sites engaged in fingerprinting. Without this information we would be unsure of what data to collect. To find fingerprinting techniques, we followed the same strategy employed by Nikiforakis et al [9]. We chose three companies that are known to engage in device fingerprinting or claim to use fingerprinting: BlueCava [4], ThreatMetrix [12], and ReputationManager [10]. We used Ghostery, a browser add-on that provides information related to privacy on the web, to find websites that used scripts from these companies. We visited these websites and viewed the page's source code. We analyzed all of the JavaScript included in the page and the JavaScript that was linked to the page to look for calls that could potentially be used to fingerprint a device. This was mostly done to verify the work conducted by Nikiforakis et al. [9] and Eckersley [6]. We wanted to ensure that the code and techniques used by entities engaged in fingerprinting was still accurately reflected by the academic papers and had not changed since those papers were published.

Crawl

We created a custom web crawler using the Java programming language and the JSoup library and used it to crawl the web in search of JavaScript code inclusions in websites. While conducting the crawl we passed a user-agent string identifying the crawler as a Firefox browser in the hopes that we would be served content similar to that seen by a web user navigating the internet with Firefox. A breadth-first search was used to visit five additional links in the domain and collect all text located between the <script> tags and everything in the JavaScript files that were linked to in the page's code. Four of the five links visited were random. The crawler made an effort to visit a link in the domain containing the words "sign" for sign-in or "login" for the fifth link. We explicitly attempted to visit login pages because we assumed that a site owner could potentially be more interested in fingerprinting users that were attempting to access more secure parts of the site that required a password. We seeded our crawl with Quantcast's top thousand websites. In addition to visiting the homepage and five links in the domain we also followed all links included in iframes with a height of zero or one. These iframes would be essentially invisible to the website's visitors since they are only one pixel tall. We followed any links included in those iframes and collected any JavaScript found there, so we would not miss JavaScript that could have an effect on the page through the iframe.

Analysis of JavaScript Code

Analysis of the collected code was perhaps the easiest step from a technical standpoint and the most difficult from a logical standpoint. We wanted to avoid false positives, without eliminating real fingerprinting code. We found that there were three calls in particular that were common to all commercial fingerprinting entities in our sample. All three companies used JavaScript to query the browser for information regarding the screen properties, the browser's user-agent, and Flash settings. We eliminated all code that did not at least mention these three calls together.

We searched for code that was similar to Bluecava's code, and placed it in one category. We decided that in order to fit in the BlueCava category a script would have to fulfill the minimum criteria set forth in Table 1. We searched through each script looking for the keywords listed in the table, and if enough were found we deemed that the code was sufficiently similar to Bluecava's code to be put in our Bluecava category. We also searched for files that contained the phrase "clients.bluecava" in the hopes of finding a script served by Bluecava itself.

All words in this column	At least one word from each set in this column	At least one word from this column
navigator.plugin window.ActiveXObject Ajax screen.width screen.height getTimezoneOffset	navigator.language systemLanguage userLanguage ---------- CLSID clsid ---------- Flash flash ---------- .getFonts font Font	Gears MSIE Math.LOG2E Math.LOG10E Dnt

Table 1: Minimum criteria to fit in BlueCava category

Similarly, we created a category of files that contained JavaScript similar to ThreatMetrix's code. Table 2 contains a list of the minimum amount of evidence necessary to place a script in the ThreatMetrix category. All files that fit this category were grouped together. We also searched our dataset for the address "online-metrix.net" since we found a ThreatMetrix script hosted at https://h.online-metrix.net/fp/check.js?org_id=u8fxw6sf&session_id=09a1cfd0d12a9be7aa16f61067f4d95c&_=1373306724562 and we wanted to find other possible fingerprinting scripts if they existed on that domain.

All words in this column	At least one word from the following column
Plugins mimeType ActiveXObject getTimezoneOffset userAgent screen progID	Flash flash

Table 2: Minimum criteria to fit in ThreatMetrix category

Finally, we created a category of files that contain JavaScript similar to that found in Iovation's code. By similar we mean any file that fit the criteria set forth in Table 3. We also went through our data looking for the strings "https://ci-mpsnare.iovation.com/snare.js" and "https://mpsnare.iesnare.com/snare.js" which are the addresses of Iovation scripts.

All words in this column	At least one word from this column	At least one word from this column
Navigator.plugin Navigator.language ActiveXObject Navigator.userAgent Date Time Screen.height Screen.width Clsid	Flash flash	Shockwave shockwave

Table 3: Minimum criteria to fit in Iovation category

We were only completely certain that a script was used for device fingerprinting when we found exact duplicates of code we had already analyzed in the first step of our methods.

Limitations of Data

There are several potential issues with the data collected over the course of the crawl. Since we were using a custom-made crawler we may have been served different content then a user utilizing a commercial browser. We attempted to mitigate this concern by sending the servers we were connecting to a user agent string identical to those used by a Firefox browser. We used the top thousand sites from Quantcast's Top Million U.S. Web Sites to seed our crawl. We discovered that 98 of the 1000 sites in the list had no domain associated with it, as if those profiles were hidden. Since there was no domain listed the site could not be visited in our crawl. Perhaps the sites with hidden profiles were more likely to contain evidence of device fingerprinting. We obviously cannot verify this hypothesis without the addresses of the sites. In addition, we are merely collecting the JavaScript code. We did not detect JavaScript as it was being executed. We do not know whether the code is ever used by the site, or when it is employed. We merely note its existence.

Limitations of Analysis

It is almost impossible to state with absolute certainty whether JavaScript code is being used to fingerprint devices or for some other purpose. Companies conducting fingerprinting do not normally have an incentive to indicate which scripts are used for fingerprinting. On the contrary, some fingerprinting scripts are deliberately difficult to find and analyze. Some of the companies consider their code to be confidential or a trade secret [4]. JavaScript code may go through a process known as minification, which removes whitespace unnecessary to the function of the program. Minification results in smaller file sizes and faster page load time. The removal of whitespace can make JavaScript much more difficult for a human to read and analyze. As a consequence the code is often difficult for human readers to interpret. Minification is generally not intended to hide the code's purpose. Code can be deliberately obfuscated, however, making it much more difficult to for a human to read or a script to analyze. Clearly it can be in a company's interest to hide the implementation of their code. If a person who wishes to avoid detection could find and understand the techniques used by fingerprinting scripts it would be easier to achieve their goal of anonymity. Our analysis of the data may not have discovered fingerprinting code that has been obfuscated.

3. RESULTS AND DISCUSSION

After independently finding and analyzing the JavaScript code identified by Ghostery as tracking scripts used by BlueCava, Iovation and ThreatMetrix, we created a list of fingerprinting techniques used by these commercial tracking companies. We found several discrepancies between our list of techniques used by fingerprinters and the list created by Nikiforakis et al in their 2013 study [9]. The companies may have changed

their code since the study or Nikiforakis et al may have analyzed a different script. We found that Iovation did not attempt to collect information on time zone while Nikiforakis et al. reported that Iovation did use the time zone as part of the fingerprinting process. Iovation also sought to enumerate plugins, queried for the ActiveXObject, and appeared to mention clsids, which Nikiforakis did not report.

We discovered that none of the code gathered from the links in the iframes contained evidence of device fingerprinting.

We found three websites that fulfilled the criteria to be included in the BlueCava category. None of these sites contained scripts from BlueCava. There was one domain which contained no JavaScript that would fulfill the criteria to be put in the BlueCava category but did contain a link to "//clients.bluecava.com/data/?p=280C1688-8CC3-4510-B2E6-F5F74AB76AEF" which, according to Ghostery, is a tracker. This site contains code from Bluecava that fulfills the requirements to be placed in the BlueCava category of fingerprinting scripts.

Our crawler located two websites that contained JavaScript code that met all the requirements necessary to be included in the ThreatMetrix category. We saw no evidence of any scripts from ThreatMetrix itself.

Finally, we found 54 websites in the Iovation category. All three of the websites in the BlueCava category were also found in the Iovation category. We found five websites that used fingerprinting scripts from Iovation.

There is a striking disparity between the number of websites that fit in the Iovation category compared to the BlueCava and ThreatMetrix categories: 54 to 5. There are almost certainly more false positives in the Iovation category. There were significantly more domains using Iovation scripts though, so perhaps the methods used by Iovation are among the most popular, implemented or copied device fingerprinting techniques on the internet today.

We found evidence of device fingerprinting in a very small fraction of the most popular websites. Only 5.6% of the top 1000 websites contained sufficient evidence to be deemed similar to commercial fingerprinting code in our analysis. If we limit ourselves to only considering scripts identified as originating from BlueCava, Iovation or ThreatMetrix and confirmed with Ghostery as tracking scripts, only six of the thousand sites in our sample (just 0.6%) contain device fingerprinting code. Cookies, in contrast, are used on almost all the most popular sites [3].

4. CONCLUSION

Our crawl of the top thousand most popular websites as ranked by Quantcast found very little evidence of device fingerprinting. It appears that the reports of the death of the cookie were greatly exaggerated by the media. The results of this study indicate that

tracking via device fingerprinting is still only employed by a very small fraction of popular websites.

We can only speculate as to why this would be so. It is possible that cookies are still deemed stable enough to function as a tracking technology. If this is the case there would be little motivation to implement a different tracking system like device fingerprinting. There is also a possibility that many more websites are using device fingerprinting and are either obfuscating their code or using techniques that are not similar to the three commercial tracking companies we studied.

One other possible explanation could be proposed. It is possible that the number of visitors that popular websites receive is so incredibly large that a unique fingerprint cannot be established based on the system configuration. It is possible that fingerprinting as a technology is less viable on very popular websites. Say for example that an internet user navigates the internet on a computer and browser that can be distinguished by fingerprinting entities in a set of ten million other users, but not one hundred million other users. This user could then be uniquely identified on a website with one thousand visitors, but not on a website with one billion visitors. The increased difficulty of collecting enough information to uniquely identify a user in a vast pool of other users may discourage popular sites from engaging in device fingerprinting.

Future Work

It would be useful to have a crawl that visited more domains and visited more links in each domain. A deeper, broader search would tell us more about the prevalence of device fingerprinting. The crawl should also be conducted using real browsers instead of the Java crawler. Sites using device fingerprinting may respond differently depending on the web browser used.

It would also be helpful to collect more than JavaScript code. Those who wish to fingerprint can also collect useful information from http headers and by querying Adobe Flash [9]. The crawl should be enhanced to collect this information as well so we may more confidently determine whether a site is using fingerprinting techniques.

Once we develop a system that can accurately identify a site that uses fingerprinting it could be helpful to develop a browser extension that informs users as they are browsing whether or not they are being fingerprinted by the website they are visiting. It would also be useful if this browser extension could be configured to block fingerprinting once it is detected.

There are many interesting questions regarding the interplay between cookies and device fingerprinting. A question that immediately presents itself is whether sites that use device fingerprinting use fewer cookies than sites that do not use fingerprinting. If a site were confident that its fingerprinting system was effective it is reasonable to assume that they would feel less need to use cookies for tracking. Perhaps these sites use cookies and rely on fingerprinting as a sort of backup system in case the user deletes their cookies. Years ago, Eckersley suggested that websites could use fingerprinting as a way to

respawn cookies that a user had previously deleted [6]. It would be interesting if someone could find evidence of this practice.

5. ACKNOWLEDGMENTS

Our thanks to ACM SIGCHI for allowing us to modify templates they had developed, and to Tyler Stocksdale for technical assistance provided. We would also like to thank Aimee Tabor, Director of Education at TRUST Science & Technology Center.

This work was supported in part by TRUST (Team for Research in Ubiquitous Secure Technology), which receives support from the National Science Foundation (NSF award number CCF-0424422).

6. REFERENCES

[1] ACAR, G. JUAREZ, M. NIKIFORAKIS, N. DIAZ, C. GURSES, S. PIESSENS, F. PRENEEL, B. 2013. FPDetective: Dusting the Web for Fingerprinters. *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security* (CCS '13). ACM, New York, NY, USA, 1129-1140. DOI=10.1145/2508859.2516674 http://doi.acm.org/10.1145/2508859.2516674

[2] ANGWIN, J. AND VALENTINO-DEVRIES, J. 2010. Race Is On To 'Fingerprint' Phones, PCs. The Wall Street Journal. Published Dec 1 2010.

[3] AYENSON, M. WAMBACH, D. SOLTANI, A. GOOD, N. HOOFNAGLE, C. 2011. Flash Cookies and Privacy II: Now with HTML5 and ETag Respawning. http://ssrn.com/abstract=1898390

[4] BlueCava Code. http://ds.bluecava.com/v50/AC/BCAC5.js

[5] Device Fingerprinting. Oracle. 2013. http://docs.oracle.com/cd/E27559_01/admin.1112/e27207/finger.htm

[6] ECKERSLEY, P. 2010. How Unique Is Your Web Browser? https://panopticlick.eff.org/browser-uniqueness.pdf

[7] KOHNO, T., BROIDO, A. AND CLAFFY, K. 2005 Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 93--108, May 2005. http://www.caida.org/publications/papers/2005/fingerprinting/

[8] Multi-Layer Device Fingerprinting. Kount. 2013. http://www.kount.com/products/complete/multi-layer-device-fingerprinting

[9] NIKIFORAKIS, N. KAPRAVELOS, A. JOOSEN, W. KRUEGAL, C. PIESSENS, F. VIGNA, G. 2013. Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting. *IEEE Symposium on Security and Privacy*, 2013, pp. 541-555. https://lirias.kuleuven.be/bitstream/123456789/393661/1/cookieless_sp2013.pdf

[10] OLSON, S. 2009 Device Fingerprinting Techniques – Several Choices. June 4, 2009. https://www.iovation.com/blog/device-fingerprinting-techniques-several-choices

[11] SOLTANI, A. CANTY, S. MAYO, Q. THOMAS, L. HOOFNAGLE, C. 2009. Flash Cookies and Privacy. http://ssrn.com/abstract=1446862

[12] How Device Fingerprinting Works. *ThreatMetrix*. 2012. http://www.threatmetrix.com/how-device-fingerprinting-works/

[13] TANNER, A. 2013. The Web Cookie Is Dying. Here's The Creepier Technology That Comes Next. *Forbes.* http://www.forbes.com/sites/adamtanner/2013/06/17/the-web-cookie-is-dying-heres-the-creepier-technology-that-comes-next/

[14] VIRTUAL REALITIES. "United Virtualities Develops ID Backup to Cookies." United Virtualities.31 Mar. 2005. 07 July 2009. http://www.unitedvirtualities.com/UV-Pressrelease03-31-05.htm

[15] WALL STREET JOURNAL. What They Know: The Business of Tracking You on the Internet.

[16] What is Safari. Apple. 2013. http://www.apple.com/safari/what-is.html