Evaluating the effectiveness of defences to web tracking

Darrell Newman

Technical Report

RHUL–ISG–2018–7

5 April 2018

Information Security Group
Royal Holloway University of London
Egham, Surrey, TW20 0EX
United Kingdom

Darrell Newman
100851994

Evaluating the effectiveness of defences to web tracking

Supervised by Dr. Geraint Price

Submitted as part of the requirements for the award of the MSc in Information Security at Royal Holloway, University of London.

I declare that this assignment is all my own work and that I have acknowledged all quotations from published or unpublished work of other people. I also declare that I have read the statements on plagiarism in Section 1 of the Regulations Governing Examination and Assessment Offences, and in accordance with these regulations I submit this project report as my own work.

Signature:

Date:

Intentionally Blank

Summary

This thesis fulfils part of the requirement for the award of MSc in Information Security at Royal Holloway, University of London. It takes an extensive look at web tracking, a topic of particular interest to academia in recent years, and how practical it is to defend against. To do this, we will look at the different approaches and methods adopted by trackers, from traditional approaches using cookies to the more recent discoveries such as those for producing unique values known as fingerprints to help identify browsers and devices across the internet.

With this knowledge, a comprehensive list of defences is presented and a selection picked for investigation based on their ease of use and, more importantly, effectiveness. These are tested using a bespoke framework written in Python specifically for this study to automate Mozilla's Firefox browser and gather and analyse the results. The results show that for the defences tested it is practical to reduce being tracked by a respectable amount, although impractical to completely prevent tracking. Finally, a number of recommendations are presented to help preserve privacy while online.

Acronyms

CNAME Canonical Name.

DNT Do Not Track.

DOM Document Object Model.

ICR Internet Connection Record.

LSO Local Storage Object.

MAC Media Access Control.

PBM Private Browsing Mode.

SOP Same Origin Policy.

TOR The Onion Router.

TP Tracking Protection.

TPL Tracking Protection List.

VPN Virtual Private Network.

Glossary

Canonical Name A DNS record type allowing a domain to point to another domain. For example, a DNS CNAME record can be used to point the domain `http://documents.mysite.com` to the actual site located at `http://docs.mysite.com`.

Cookie A small text file added to the user's browser by a website to store state between browser sessions.

Data Aggregator Data aggregators compile behavioural profiles on users by obtaining data from multiple sources and aggregating the data into summary profiles which can then be used for targeted advertising or sold on.

Document Object Model The Document Object Model is a programming interface used to represent the structure of a webpage within a web browser.

Domain Context A Domain Context is the environment in which a webpage is loaded. As webpages can load data from multiple domains, each part is loaded into it's own domain context for safety and security, preventing access to the domain context of other parts.

Fingerprinting A technique used to generate a unique value based on the particular configuration of the user's device, such as their web browser, which can be used to identify the device on future visits to the website.

Google PREF cookie A type of cookie set by Google containing a unique identifier and other user preferences, such as location information.

Internet Connection Record An Internet Connection Record contains the details of a connection made between a user and a service, such as the user connecting to the BBC website from their home PC.

Same Origin Policy The Same Origin Policy prevents access to resources owned by another context or domain. An example would be preventing `http://sitea.com` reading the cookies set by `http://siteb.com`.

List of Tables

List of Figures

Intentionally Blank

Contents

II Web Tracking Mechanisms

III Defences

IV Testing Browser Based Defences

V Conclusions and Closing Remarks

Intentionally Blank

Introduction

Privacy, and the wish to not be tracked online, has become an area of personal interest. Awareness of web tracking has increased in recent years, possibly due to trade in personal information and the murkier topic of mass surveillance being brought to the attention of the general public. As technology improves and offers new tools and approaches, tracking techniques become harder to identify and even harder to prevent, making the ability to shake off trackers increasingly more difficult. The aim of this study is to answer a single question:

Is it feasible to defend against being tracked whilst online?

The likely answer is no. However, more concrete evidence would be helpful in making future decisions, otherwise selecting defences to protect privacy online would become more luck than judgement, neither of which are ideal. If eliminating tracking proves not to be feasible, an acceptable mitigation strategy would be to reduce tracking to comfortable levels, or defend against specific methods depending on their impacts on privacy. This study aims to provide sufficient detail to make these decisions in an informed way.

Objectives and Scope

To answer the question posed in the *Introduction*, objectives will be set and scope defined before starting the study. The objectives are to:

1. *Research and explain the current techniques used to track users on the surface web*
2. *Analyse the privacy impacts resulting from web tracking*
3. *Appraise and evaluate the current defences to web tracking*
4. *Suggest and recommend effective defences to web tracking*

The scope for the study will be restricted due to the size of the subject area. To try and investigate all types of tracking on all platforms in the time available would be a recipe for disaster, so we will err on the side of caution and define the scope, constraining it to the following items:

1. *Tracking will be examined in the desktop environment, using the desktop version of Mozilla's Firefox web browser. Tracking on mobile environments will be out of scope.*
2. *We will be looking at tracking on the surface web, using mainstream websites for testing. The dark web is out of scope.*
3. *The tracking mechanisms in scope are those using stateful, stateless and fingerprinting techniques.*
4. *Existing defences as presented by academia are in scope, for example, web browser configuration and add-ons and extensions.*

Document Structure

The remainder of this document is structured as follows:

Part I - Web Tracking Primer provides an introduction to web tracking before reviewing the academic work related to our study, spanning from 2006 to 2017. We look at the reasons for tracking users, along with the implications and impacts of being tracked. It finds the impacts to be quite wide and varied, including behavioural targeted advertising, price discrimination, targeted malware delivery and affected decision outcomes.

Part II - Web Tracking Mechanisms delves into the approaches and technologies used to track users, split into three chapters: *Stateful Tracking Mechanisms*, *Fingerprinting* and *Other Tracking Mechanisms*. The first chapter examines the more traditional approaches to tracking based on storing data, the second chapter presents more recent approaches to stateless tracking such as browser fingerprinting, while the final chapter examines miscellaneous approaches such as Internet Connection Records held by ISPs.

Part III - Defences presents an extensive list of defences over three chapters: *Defences Based on the Web Browser*, *Defences Based on Web Browser Configuration* and *Technical Defences*. Each defence is reviewed and presented with advantages and disadvantages.

Part IV - Testing Browser Based Defences selects a number of defences for testing and evaluation. All the defences selected are easily implemented and based on the web browser. This part goes on to appraise and evaluate their effectiveness and presents the quantitative outcomes of these evaluations. Finally, recommendations are made based on the reviewed research and the findings from the tests conducted, split into two categories: *Easily Implemented Recommendations* and *Technical Recommendations*.

Part V - Conclusions and Closing Remarks concludes the study, finding reducing the effects of web tracking to be achievable but complete prevention to be inconclusive without conducting further research in this area, particularly in relation to browser fingerprinting. Finally, suggestions for future work are discussed before the author's closing remarks complete the study.

I Web Tracking Primer

1 Web Tracking 101

Before getting underway, we need to define what web tracking actually is in the context of our study. In this context, web tracking is the act of observing the online actions of a user so as to discover their interests and habits in order to either sell this information or show the user targeted advertising. This is quite a wordy explanation and not the only reason web tracking happens, as we'll see shortly in the *Why Track Users?* section, but it's sufficient for the moment.

The trackers aim is to consistently identify users and their devices on the websites the trackers are active on. We can use a social network as an example to help explain this further. After logging into the social network, a cookie is set on the user's device as a way of identifying this user. Once finished on the social network, the user moves on to browse a selection of other websites, each of which contains a widget owned by the social network (more on social widgets later). Each time a webpage containing a social widget loads, the cookie identifying the user is sent to the social network, along with the details of the website being visited, allowing the social network to discover the browsing activities of the user and create a profile with this data. This is only one example of tracking, using a common approach, so does not tell the whole story as we will see. Now, imagine this scenario on a truly magnified scale of tens or hundreds of trackers hoovering up your browsing activities on most of the sites you use on a daily basis, over years - quite a frightening prospect.

Many free services, such as news sites and email providers, use tracking to fund their services with online advertising. Sadly, tracking is not always restricted to free services, so the old adage *if you're not paying for it, you're the product* doesn't necessarily apply any more. Perhaps a better version is simply: *you're the product.*

2 Related Work

Early work in this area tends to focus on the leakage of PII and the resulting impacts to users' privacy. Studies such as *Generating a Privacy Footprint on the Internet* by Krishnamurthy and Wills looks to quantify the leakage of information between unrelated websites in the form of a 'privacy footprint' to provide a sense of the scale of information sharing taking place between entities [1]. Later studies by the same authors in 2009 and 2011 continue with privacy as a focus, this time concentrating on the leakage of information from both social networks [2] and a broader range of sites [3] respectively, with the 2011 study finding 75% of 120 websites tested leaked sensitive user information directly to third parties.

Other studies also look closely at social networks, such as Korolova's 2010 paper *Privacy Violations Using Microtargeted Ads: A Case Study* which hypothesises how extremely targeted advertising could be used to reconstruct a person's private information just by examining the displayed adverts, using Facebook as a platform for the study [4]. In the same year Toubiana et al. proposed an architecture for 'privacy-preserving advertising' [5], with the same approach being examined further in 2011 by Bilenko et al. in their work *Targeted, not tracked: Client-side solutions for privacy-friendly behavioral advertising* [6]. Javier Parra-Arnau's *Pay-per-tracking: A collaborative masking model for web browsing* study, published in early 2017, also examines this area of tracking [7].

Recently, Starov et al. studied the leakage of PII from contact forms of websites [8]. Sadly, the findings were considerable sharing of personal information with third parties without the user's knowledge, in some cases using quite underhanded approaches to data capture such as JavaScript libraries capable of sending each keystroke as it is typed without the need for the user to even submit the contact form. In the same paper, Starov et al. presented a countermeasure by developing a browser extension, 'Formlock', to warn users when they encounter potentially privacy-compromising web forms [8].

The use of cookies in surveillance by the security services has become of interest over the last few years, and a number of papers have studied the surveillance impacts of third party cookies [9, 10, 11].

A variety of studies look at tracking from new perspectives, such as Lerner's et al. analysis of web tracking evolution over the past 20 years [12], and Schelter's and Kunegis' 3.5 billion

site study to see if website content type and country of origin have an impact on the number of trackers present on a site [13].

A relatively large number of studies examine both tracking mechanisms and defences together, such as [3, 14, 15, 16, 11, 10, 17, 13, 18], with a selection focussing on particular tracking methods such as device fingerprinting [19, 20]. Few of the studies examine the effectiveness of defences or the ease with which they can be applied in everyday use. A popular defence examined is the browser add-on, with studies developing their own add-ons being a common approach [21, 8]. Later works (2015 onwards) tended to specialise on defences to particular threats, such as fingerprinting [22, 23, 24] or browser-based defences [25, 26, 27, 28, 29, 30], some of which produced good results such as a reported 67.5% reduction in cookies set when using Firefox's Tracking Protection (TP) [26].

3 Why Track Users?

Before diving into how users are tracked and how to defend against it, it would be helpful to examine the reasons why tracking is so prevalent on the internet today. One of the most active groups tracking users is online advertisers. Spending on online advertising surpassed £10 billion in 2016 [31], accounting for roughly half of the entire ad spend for that year [32]. To advertisers, the details of a single site are not as helpful as knowing all the sites the user has visited over an extended period of time. With this information, advertisers can combine these data to form behavioural profiles specific to individuals [14], often containing staggeringly accurate, personal data [33], and use it to deliver highly-targeted advertising tailored specifically to the user based on their past online activity.

Alongside advertisers, there are other actors with an interest in your data [9, 34]. Data brokers collect personalised data about you to sell it to interested parties, such as advertisers, reference checking companies and even governments [34, 35, 36]. Once collected, your data is often augmented with additional information which may not have been supplied by you, nor even be wholly specific to you, before being shared with other third party providers or sold on for profit [34]. This augmentation of data is quite easy to achieve. For example, assuming you are not anonymising your web surfing habits, your IP address will provide your location to a general area. Information regarding the demographic of an area such as average age, level of disposable income and home ownership information is readily available and can be used to supplement the data collected about you to provide more targeted advertising. Some companies even purchase information about your offline life to enrich the data collected about your online life [37, 38]. If you have used a service that requires a full postcode when searching, such as when using a car location service like `http://autotrader.co.uk`, then the likelihood you restricted your search to a specific area by providing your full postcode is quite high. A quick read of the autotrader.co.uk cookie policy states third parties will also set cookies when you visit the website [39]. To make matters worse, autotrader.co.uk does not appear to support HTTPS, meaning your postcode may potentially be sent over the network in clear text making it observable directly from the network traffic.

Some forms of web tracking work well in the fight against fraud, particularly behavioural biometrics and device fingerprinting, both of which are discussed later. Behavioural biometrics identify individuals based on their interactions with devices, such as a fingerprint scanner on

a smartphone, or the typing characteristics of a user [40, 41]. Based on device interaction, behavioural biometrics can be used to continuously authenticate a user, a practice known as *active authentication* [42]. This convenience could come at a cost though, as, in theory at least, it would become possible to track individuals based solely on their actions in front of a keyboard or mouse.

Fingerprinting techniques produce a unique value based on the attributes and configuration of a device or web browser. This method has proven to be extremely accurate [19, 16, 22, 23] so would, for example, allow a bank to identify a single machine being used to manipulate multiple bank accounts, which could indicate a potential fraud.

Governments and intelligence agencies also have an interest in tracking users. In their paper *Cookies That Give You Away: The Surveillance Implications of Web Tracking*, Englehardt et al discuss how the accuracy of third party advertising cookies have made them an attractive proposition to intelligence agencies as a way of tracking individuals [9, 11, 43]. This is made easier by third parties that do not use HTTPS to encrypt their web traffic, allowing cookies to be passed around the network in plain text.

The final reason discussed for tracking users is also, sadly, one of the most common: crime. Unfortunately, criminals are using web tracking techniques and compromised advertising networks to identify the system configuration of users in attempts to defraud by delivering malware with a greater chance of successful infection [44, 45, 46, 47]. This is the sadder side of the link between web tracking and crime. Thankfully, web tracking can also help to prevent crime as demonstrated by the use of fingerprinting as a way of identifying the same device accessing multiple bank accounts presented earlier.

4 Implications of Being Tracked

4.1 Use of Sensitive Data for Personalisation and Advertising

Personalisation is the process of tailoring content or services specifically to a user, such as seeing product suggestions when signing into your account on Amazon. It takes many forms, including personalised search results and social media posts, however, it is most commonly found in retail due to its ability to increase visitor conversions, and therefore revenues [48]. As an example, when Co-Operative Travel implemented personalisation on their website, traffic reportedly increased by 95% and revenues by a staggering 217%, all attributed to customer personalisation [48].

A study published in 2013, *Do Not Embarrass: Re-examining User Concerns for Online Tracking and Advertising*, found consumers tended to react favourably to targeted advertising, accepting it as the price to pay in exchange for free services and content [49]. A core aim of personalisation is to make the customer feel comfortable, even special, which in turn makes one-to-one marketing more easily accepted by the consumer thus making them easier to sell to. Unsurprisingly, this becomes more achievable given a suitable level of background information on your target audience. This is where potential privacy issues lie. The same study found the majority of participants feared losing PII and financial information to third parties. The same participants accepted the currency of free services is personal data, however the majority taking part in the study felt uncomfortable with sharing more than 25% of their browsing histories. This was due to a cross-section of users having already been presented with advertising in the past felt to be overly personal due to PII or other sensitive information being shared with third parties, as well as adverts felt to be embarrassing or of questionable content [49]. Similar research conducted on the leakage of sensitive data by social networks found the sharing of personal data with third party data aggregators, including PII, happened almost immediately upon signing up for an account [2]. The section *Collection, Leakage and Sharing of PII* covers the leak of PII further.

4.2 Price Discrimination

Price discrimination is the act of charging different prices for the same goods or services based on information known about the consumer. There are three common forms of price discrimination: first-degree, where a seller is able to charge the maximum possible price for every unit sold, second-degree, commonly seen as discounts for multiple purchases or bulk buys, and third-degree, based on market segmentation, for instance charging a lower price for children's cinema tickets compared to the adult equivalent [50].

First-degree price discrimination, also known as 'perfect price discrimination' [50] is unsurprisingly the Holy Grail for retailers as it enables them to charge the highest possible price for every unit sold, therefore maximising their profits. This form relies on the identification of particular information that can be used to segment users into markets based on particular criteria, such as location, demographic, web browsing history, transaction history, whether the user is logged in to their account or browsing as a guest, or the brand or type of laptop or smartphone you use to shop online.

Detection of price discrimination as it happens can be quite challenging. For example, most e-commerce sites require a user to create an account and sign-in before making a purchase, as well as registering a credit card and delivery address. All of these actions make sensitive, personal information available to the vendor, primarily to prevent fraudulent activity. The vendor can also purchase further data relevant to the individual from data aggregators and create an instant, comprehensive profile of their new customer. Going forward, each purchase made by the new customer will be stored and likely reviewed when displaying prices for products, as will the browsing activity of the customer while they are logged in. To make matters worse, both stateful and stateless tracking techniques are also capable of linking an individual to a specific account, meaning the user can be tracked relatively easily even when logged out. Thankfully, simple defences such as clearing cookies can work, as shown in Appendix A.

4.3 The Filter Bubble

One side effect of content personalisation is known as the filter bubble. The filter bubble is most obvious when using the same search engine or social network over a period of time. After a while, the content shown to the user will be partly decided by algorithms and other criteria, for instance in the case of social networks where the user's connections start play a part in deciding what content is displayed [51]. This has the potential to leave users in their own unique echo chamber of biased search results and social media content [52].

To deal with the huge amount of information on the internet today, companies like Google and

Facebook have become gatekeepers to the content delivered to our devices, using algorithms as a way of selecting the content they believe to be most pertinent to us [52, 51]. Web tracking plays a sizeable part in this as by storing information on our past browsing history and searches along with details of the links we click, articles we read and videos we watch, search engines and social networks are able to reduce an overly large set of results to a smaller, concise set more in keeping with our past behaviours. This may be driven by the need to retain users in order to stay in business, as the attention span of users not seeing applicable search results is generally quite short. A study conducted in 2009 by Alexander van Deursen and Jan van Dijk demonstrated this, finding 90% of users conducting a web search do not leave the first page of results, choosing to alter the search criteria and perform another search, and 36% of participants did not look at more than the first three results [53].

4.4 Malware Delivery

2016 witnessed a dramatic rise in malvertising, the practice of using compromised digital advertising networks to deliver malware directly to web browsers [44]. According to reports, a number of high-level websites including the New York Times and the BBC were found to be unwittingly serving ransomware to visitors due to compromised digital advertising networks used by the sites [45].

While both stateful and stateless techniques are useful to malvertisers, stateless fingerprinting becomes the technique of choice due to it's reach and abilities. Using JavaScript, fingerprinting techniques are capable of reliably identifying the make and version of both the web browser and operating system a user has installed. Under certain circumstances, device fingerprinting over the web has even been used to retrieve computer names and hard disk identifiers [15]. With such capabilities to reliably identify system information such as browser type and operating system, malware can be written for a specific browser version running on a particular OS version, vastly increasing the likelihood of successful infection.

4.5 Collection, Leakage and Sharing of PII

The loss of control of PII and sensitive information could be viewed as one of the biggest privacy impacts resulting from web tracking. Leakage of PII may be neither obvious nor expected by the user under most normal circumstances, such as creating a new user account with a website of completing an online application form for a job. Such sharing of personal data, without the owners consent or knowledge, could lead to unpleasant outcomes such as identity theft [2, 10, 54].

As seen earlier in the section *Use of Sensitive Data for Personalisation and Advertising*,

users do have concerns about losing control of their personal or sensitive data [49]. In most situations, there is no ability to see what is being shared let alone any ability to control or opt-out of the sharing. Once collected, it could well be the case that the user has little or no ability to recall or remove their data from the systems in which it resides, whilst the new owner can decide how best to monetise this potentially new asset class [54]. Sadly, even data we would expect to remain confidential like the compulsory data provided to HMRC for tax purposes, may not be exempt from being seen as a new revenue stream [55].

In other circumstances, the collection of personal data may be explicitly covered by privacy policies and statements of terms and conditions. This could well be the case for free services requiring the user to agree to conditions before being allowed to use the services, like the provision of free wifi on buses in Sydney, Australia. Catch Oz, the provider of the service, outline the information they may collect in their privacy policy. This includes: name, address, date of birth, telephone number, email address, location information, demographic information and browsing history [56]. The policy also explicitly states they will augment online data with data obtained from additional sources [56], presumable to produce a profile of sufficient granularity for marketing needs or resale. Catch Oz's privacy policy is clearly written, easily accessible and even suggests user's create a fake profile if they feel uncomfortable with the level of data collection. This would lead us to two possible conclusions: either users do not read the privacy policy, or they are agnostic in relation to privacy and organisations collecting and processing their PII data.

4.6 Affected Decision Outcomes

One example of decisions being affected by web tracking has already been discussed, price discrimination, however other areas such as employment, insurance and financial credit can be equally influenced by data collection. Using data collected this way as a basis for making important decisions could have disastrous effects, particularly in light of how the data is collected, the lack of input, knowledge or right to reply from the user, and also how the data may be open to manipulation by the new owner.

Once data is under the control of third parties it can be aggregated, enriched and sold or shared, potentially making it accessible to organisations for use in decision making. Classifications are added to the data as part of the aggregation process, bucketing individuals in segments depending on their browsing history [5]. One example of this classification is men buying chocolates and flowers online being labelled as *'men in trouble - presumed to have relationship problems'* [34]. It's easy to see how such poor classification of data could have damaging effects, and it also highlights the effect of the lack of context surrounding the data. Perhaps buying a mother's day gift in person for cash could benefit your credit rating after all.

Credit card companies have been known to set cookies when customers check their statements online, allowing the issuer to track the customer's online activities after leaving their website [34]. Indeed, in 2009 American Express reportedly reduced a customer's credit card limit by more than half after mining data about the credit worthiness of it's customers and applying the findings in a blanket fashion. When asked for an explanation, American Express allegedly told the customer 'other customers who have used their card at establishments where you recently shopped have a poor repayment history with American Express' [57].

5 Summary

In this opening part we presented a high level definition for web tracking and defined who is performing the tracking before looking at the potential side effects and impacts. In part II, *Web Tracking Mechanisms*, we will take a close look at how tracking is performed, examining techniques that store state on the user's device as well as those capable of tracking a user without needing to store anything. We'll show how tracking is ingrained with the fabric of the internet, based on the same technology used to provide content and services, and how this complicates trying to defend against it.

II Web Tracking Mechanisms

6 Stateful Tracking Mechanisms

This section looks at mechanisms that store state on the user's device in order to perform tracking.

6.1 HTTP Cookies

Cookies have been a popular approach to tracking since their invention in the mid-1990's [5, 3, 14, 15, 16, 10, 11]. As users visit sites with embedded advertising, a unique identifier is created by the advertiser and stored in a cookie on the visitor's device [58]. As the user accesses other websites with advertising provided by the same advertiser, the cookies are sent back to the advertisers thus creating a link between the ID stored in the cookie and the sites visited. A behavioural profile of the user's online activities can be created using this information, with the advertising ID stored in the cookie acting as a reference key.

Setting a cookie simply requires the website to include a Set-Cookie header, along with a name and value for the cookie, in response to a HTTP request for an asset such as a webpage [59]. Once set, the cookie will be included automatically with every future request to the owning domain until the cookie expires, which could be tens of years in the future.

6.1.1 Distinguishing between first and third party cookies

The distinction between first and third party cookies is an important one. Most web browsers include cookie management functionality, allowing users to single out third party cookies and prevent them being set as a way to improve privacy [60]. This is good for users and bad for trackers, as, from the trackers perspective, the cookie is lost and cannot be used to perform tracking. It therefore makes sense for trackers to favour first party cookies if at all possible. We will discuss this further shortly.

When a user requests a webpage, the transaction is between two parties: the user, or more specifically the user's web browser, and the website requested, which is the first party. If the requested webpage includes content from providers outside of the first party domain, such as news feeds, weather applets or videos, the transaction is no longer between two parties, as

third parties have been introduced. This is easier to explain with a diagram, so Figures 6.1 and 6.2 should help clarify the explanation (*source of both images: Clearcode* [58]):

Figure 6.1: First party cookies

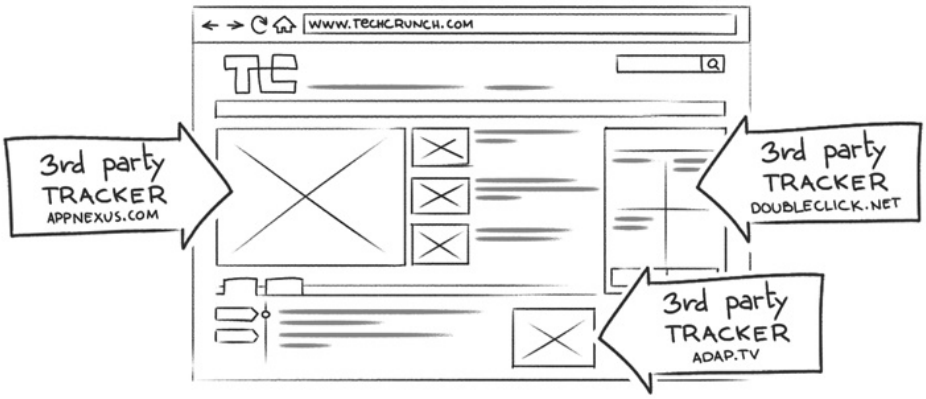


Figure 6.2: Third party cookies

6.1.2 The Same Origin Policy (SOP) and how to circumvent it

To improve security and stop websites hoovering up each other's cookies, cookies are governed by a principle known as the Same Origin Policy. This principle governs how cookies are accessed, and ensures they remain private. If `http://sitea.com` sets a cookie, the Same Origin Policy prevents `http://siteb.com` from accessing the cookie, either directly or by using JavaScript within the page if the HttpOnly flag is also set [61, 62]. However, it's possible to circumvent the Same Origin Policy by fooling the browser into believing the cookie originates from the first party domain, and not the third party that is actually setting it. One way to do this is by using a DNS Canonical Name (CNAME) record to map the

third party domain name to a name the browser will believe as belonging to the first party [63]. This technique was found to be prevalent in the wild by Krishnamurthy and Wills as a way of sharing cookies as far back as 2006, so is definitely not new [1, 3]. It works as follows: `http://mysite.com` wishes to share cookies with `http://tracker.net`, but the Same Origin Policy prevents `http://tracker.net` access to the cookies directly. To overcome the Same Origin Policy, a DNS CNAME record is created with a name containing the Top Level Domain (TLD), such as `http://tracker.mysite.com`, and pointed to the third party domain of `http://tracker.net`. The new domain `http://tracker.mysite.com` is then used by `http://mysite.com` to share the cookies as the web browser assumes both URLs are part of the same TLD, and does not enforce the Same Origin Policy.

Another approach to circumventing the Same Origin Policy and set first party cookies is to redirect to the tracking site, set the desire cookies, and then redirect back to the requested site. This allows first party cookies to be set as the user has actually visited the tracker's site directly, although not by choice [14, 10].

6.1.3 Cookie leaking and syncing

As we have seen, the Same Origin Policy (SOP) makes sure cookies owned by one domain are not directly accessible to another. Cookie leaking and cookie syncing are two more ways to circumvent the SOP and share cookies between unrelated domains. Leaking a cookie to another domain can be achieved quite simply by including the cookie to be leaked in the request in some way, for instance in a parameter in the query string of a GET request or as part of the URL in the referer header. Either of these approaches would allow a first party to leak cookies to a third party so they can be used for tracking [10].

Cookie syncing is slightly different to leaking, and is the process of mapping a shared cookie to multiple behavioural profiles stored on different systems managed by different entities. In this situation, syncing cookies ensures the unique identifier on all systems references the correct behavioural profiles for the user [58], allowing multiple profiles to be aggregated into a single profile encompassing the data from all systems. Sharing cookies out of sight is another potential approach, for example by two parties performing a direct transfer of data between systems.

6.1.4 Replacements for advertiser unique ID

As users start to adopt multiple devices it becomes harder for third parties to track cookies and attribute them to a single user. An example would be devices used in different contexts, such those used at home and work. Each device would contain cookies with different unique advertiser identifiers, making attribution to a single user much more complex [3]. This issue is

trivial to overcome, as other identifiers can be used to replace the advertiser ID and solve this issue. The most obvious candidates are email addresses, social handles and mobile telephone numbers. All of these tend to be reused for convenience, making them ideal for aggregating data across systems. Many online services now require the user to create an account in order to access the service - even free services - typically using an email address or telephone number as the identifier or for verification purposes. These replace the advertising identifiers stored in cookies, making the aggregation of the cookies across multiple systems easier and permitting the user's behavioural profile to contain information gathered from all the user's devices [3].

Social networks are a prime example of this approach. Facebook requires people to use their real name when creating an account [64] and will only enable an account once the user has verified it using a code sent to a mobile telephone. Leaking this kind of account information may result in the user being deanonymised and their PII or actual identity becoming associated with their behavioural profile. As pointed out by Krishnamurthy and Wills, once in possession of identifiable information it becomes trivial to augment behavioural profiles with publicly available records such as date of birth and home address [3].

6.2 The Adobe Flash Plugin

6.2.1 Flash Local Shared Objects (LSO) a.k.a Flash Cookies

Flash Local Storage Objects (LSOs) are a particularly insiduous type of cookie used to store data on the user's computer [10]. For tracking, Flash cookies have several advantages over HTTP cookies: they are stored outside of the web browser so are not part of cookie management, can hold 100KB of data compared to 4KB for HTTP cookies and, in particular, they have the ability to reconstruct HTTP objects after deletion [65]. The final item, a process known as *cookie respawning*, can make Flash cookies particularly troublesome as respawning goes against the user's wish to clear cookies by automatically recreating them once they have been deleted. Research conducted by Ayeson et al. in 2011 found respawning allowed tracking to continue, even when using private browsing modes with cookies disabled [65].

Flash LSOs are accessible to all software using a Flash plugin. This means they are shared between multiple browsers installed on the same machine, private and non-private browsing modes, and can even populate a clean browser install as soon as the Flash plugin is installed [10].

6.2.2 Flash LocalConnection objects

Another object from the Flash stable providing data sharing capability is the Flash LocalConnection object. LocalConnection objects allow data to be shared between Flash movie files, as the movies play. Interestingly, this approach can be used to provide a data-sharing conduit between normal and private browsing modes, providing a means to track individuals using PBM [10].

6.2.3 Other uses of Flash

Flash LSOs and LocalConnection objects do not tell the whole story when in comes to the use of Flash for tracking. As we will soon see in the *Browser Fingerprinting* section, Flash also provides handy mechanisms for retrieving the fonts installed on a system, allowing them to be used as attributes in browser fingerprints. Thankfully, the use of Flash was identified as being in decline in a study published in 2011 by McDonald and Cranor [66].

6.3 Highly-Persistent Cookies a.k.a Zombie Cookies and Evercookies.

Zombie cookies have the ability to detect their own deletion and automatically recreate themselves. This makes them extremely hard to permanently delete as backups of the data needed to reconstruct the cookie can be stored outside of a browser's normal storage locations. A particularly prominent version of this cookie is the Evercookie, a JavaScript API used to replicate cookie data across multiple locations in order to ensure it persists, making it almost impossible to locate and delete all the disparate data [67, 68]. Samy Kamkar, the inventor of the Evercookie, states in a blog post that Safari's Private Browsing mode repels the creation of Evercookies, however the post dates from 2010 and the Evercookie Github repository is constantly updating, so it's unclear if this remains the case today [67].

6.4 Caches and Storage

A number of alternatives to cookies exist for storing state locally on the user's device, including caches and structured databases. HTML5 also introduced new storage structures, providing yet more ways for web developers to persist data.

6.4.1 Web cache

To improve performance, web browsers retain a cache of the most recently received documents. When a request is made for a resource, the cache is checked to see if the resource

has already been retrieved and is present in the cache. If found, the cached version is used to prevent the need for a roundtrip to retrieve the resource. Browsing histories can be reconstructed using JavaScript to request webpages from the web cache. The way it works is to query the cache using a set list of URLs. If the resource corresponding to the URL is present in the cache, the user has visited the page and part of their browsing history has been recovered [10].

6.4.2 SQLite and HTML5's IndexedDB databases

SQLite and HTML5's IndexedDB are particularly interesting approaches to client-side data storage. Both make powerful features available to web developers with their ability to store large amounts of data and retrieve it using structured database queries, allowing complex data to be processed quickly [69]. Understandably, trackers take advantage of these capabilities for tasks such as respawning deleted cookies [10].

6.4.3 HTML5 local storage

HTML5 introduced the Storage API, with three storage lifecycle options: global, local and session. Browsers tend not to support global storage mechanisms as their global nature violates the Same Origin Policy. The opposite end of the scale is session storage which is short-lived and cleared automatically when the browser closes, making it less usable for tracking beyond the current session [10]. Local storage is the most practical of the three, retaining objects until they are physically cleared by the user. Local storage provides key-value entries up to 5MB in size, and requires nothing more than JavaScript to access [10]. Unlike HTTP cookies, entries added to local storage do not expire, making it an obvious location for storing tracking data.

6.5 Social Buttons

Social buttons, such as those offered by Facebook, LinkedIn and Twitter (among many others) allow websites to create associations with social networks as a way of increasing website traffic. These are the small widgets many of us are familiar with, a selection of which are shown in Figure 6.3.

Figure 6.3: Example social buttons

Social buttons are little more than snippets of JavaScript embedded within the webpage, however they have tracking implications due to the way they are rendered. As the webpage loads, a HTTP request is made to the social network to render the button in the page. This approach allows any cookies for the social network to be included in the HTTP request, providing details of the visited site to the social network and allowing the gathered data to be associated with specific members via their social cookies [14].

6.6 Other HTTP Mechanisms

Besides cookies, HTTP provides a number of other mechanisms that can be used for user identification and tracking. As cookies have already been discussed at some length in the earlier *HTTP Cookies* section, here we will look at the remaining mechanisms and how they are used to perform tracking.

6.6.1 Etag and the Last-Modified headers

The Etag and Last-Modified headers are designed to aid client side caching of resources such as images, to reduce the load on the network from unnecessary roundtrips. Both the Etag and Last-Modified headers can persist any form of data that can be represented as a string, making them viable for storing unique identifiers for tracking purposes [10]. Both headers are sent as part of the process of retrieving web resources such as images. On the next request for the same resource, the values for these headers are sent as part of the request, therefore sending any identifiers used for tracking to the server. This approach can be used in conjunction with small 1x1 pixel images embedded in web pages, allowing each site visited to be recorded by using the Etag or Last-Modified header to return a unique identifier when requesting the tracking pixel image.

6.6.2 Referer header

The referer header contains the address of the link used to navigate to the current page [70]. Figure 6.4 shows an example of how the referer header can be used to leak information:

```
GET http://tracker.com/track.php HTTP/1.1
Referer: http://site.com/?email=user@here.com
```

Figure 6.4: Example use of referer header for leaking information

The example shown in Figure 6.4 would require a little ingenuity to achieve. This is due to referer being a forbidden header, meaning the value is set by the web browser or user

agent and cannot be set programmatically [71]. This offers some privacy protection at least, although forming the original URL accordingly would enable the user's email address to be leaked. Concerns about privacy have been expressed since the referer header was introduced, particularly in relation to leaking browser history [70].

6.6.3 User-agent header

The user-agent header is appended to outgoing HTTP requests and is used to identify the system and software being used to perform the request for a web resource. The user agent header is quite an old construct, used originally to identify the capabilities of the web browser being used so content tailored to particular capabilities of specific browsers could be created. The user-agent consists of a combination of items, including the host operating system, details of the system architecture, and details of the web browser or agent making the HTTP request. An example user-agent string is shown below in Figure 6.5:

```
Mozilla/5.0 (Macintosh; Intel Mac OS X 10.12; rv:53.0) Gecko/20100101 Firefox/53.0
```

Figure 6.5: Example user-agent string

The user-agent provides a form of identification for the device requesting the web resource, however, it's not unique enough alone to track users' actions. It's feasible for multiple devices with the same configuration to issue the same user-agent header. User-agents are primarily used for tracking in two ways: in combination with another primitive such as the user's IP address, as seen in the upcoming *IP and Mac Addresses* section [5, 15, 11], and as a parameter used for fingerprinting [15, 72]. Indeed, the HTTP 1.1 specification now discourages long, detailed user-agent strings because of their usage in fingerprinting [72].

6.6.4 Request URL

The request URL can be used quite easily as a tracking mechanism simply by passing data as parameters of the URL path or query. For instance, if `http://site.com` wishes to send the email address of a user to `http://tracking.com` using the request URL it would look similar to Figure 6.6 below:

```
GET http://tracker.com/track.php?s=http://site.com/?email=user@here.com
```

Figure 6.6: Using a request URL to leak details of a logged in user

This technique is not affected by browsing mode, or the unavailability of cookies. However,

if cookies are available they can also be included as a parameter in the same way, freeing up the HTTP cookie header for other uses.

6.7 HTML and JavaScript

6.7.1 Third party tracking code snippets

We saw in the earlier *Social Buttons* section how small snippets of JavaScript can be used to enable tracking. This approach is not restricted to social buttons, with the same approach capable of being used to set cookies or download third party scripts to perform fingerprinting or other tracking actions. Figure 6.7 shows an example snippet of tracking code, in this case used for Google Analytics (*Source: Google Analytics* [5, 73]):

```
<!-- Google Analytics -->
<script>
(function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
})(window,document,'script','https://www.google-analytics.com/analytics.js','ga');

ga('create', 'UA-XXXXX-Y', 'auto');
ga('send', 'pageview');
</script>
<!-- End Google Analytics -->
```

Figure 6.7: Example tracking code - Google Analytics snippet

The website developers add the JavaScript snippets to their webpages so that the script executes as the page loads and contacts the third party analytics or tracking site. Once contacted, the third party site stores any data sent and potentially performs other actions such as setting cookies in the user's browser [14, 11, 13]. As the user continues to browse the internet, the cookies are returned to the tracker each time the tracking code executes, thus providing the details of the webpages visited by the user.

Using JavaScript also allows tracking code to be embedded into the page dynamically at run-time and provides the flexibility of executing code within the browser environment, which provides access to the page's Document Object Model (DOM). Scripts loaded into the first party domain context have complete access to the main document's DOM, and can therefore manipulate the page to add hidden elements containing unique identifiers, manipulate web forms, set first party cookies or search for evidence of browser add-ons.

6.7.2 Web forms

Web forms can be used in a number of ways to collect user information. One of the easiest approaches is to simply ask the user for information and allow them to manually submit

it along with the form. This is an easy way of obtaining potentially sensitive data such as names, email addresses and telephone numbers, and is simple to incorporate into sign-up forms for creating new user accounts.

Some third party companies offer services to manage HTML web forms on behalf of website administrators. Such services tend to include the codification of the web form, such as the generation and styling using HTML and JavaScript, along with handling the submission and receipt of the form data [8]. Once the user clicks the submit button on the form, the form data is sent to the third party first before being passed onto the website administrator, effectively turning the third party into a data aggregator. In their study *Are you sure you want to contact us? Quantifying the leakage of PII via website contact forms* Oleksii Starov et al discovered in some cases it is not necessary for the user to even click submit on the form, with JavaScript libraries available that provide the functionality to leak user information in real-time as the user types into the form.

6.7.3 Document Object Model (DOM) window.name attribute

The window.name property of the Document Object Model (DOM) can be used as a message exchange mechanism between parties [74]. It provides much more storage capability than a HTTP cookie, 2MB vs 100KB, allowing considerable amounts of information to be exchanged [10]. Using the attribute for data exchange is simple. The first party sets the data to be exchanged in the window.name property and notifies the third party. To retrieve the data, the third party uses JavaScript to read the value of the window.name attribute and submit the data to their server.

6.7.4 Tracking Pixels a.k.a Web Beacons, Web Bugs, Pixel Tags or Pixel Beacons

Tracking pixels are usually single-pixel resources hosted by a third party tracker designed specifically for tracking purposes [5, 11]. Nothing more than a small image, they are often transparent in appearance so as to be invisible to the user when rendered on the page. Being part of the anatomy of the webpage, a request is made to the third party for the tracking pixel when the page loads. This request can be used to provide tracking information in a myriad of ways, such as via the request URL query string, referer header or an Etag. Figure 6.8 shows an example request:

```
GET /nameofhost/trackingpixel.png?moretrackinginfo HTTP/1.1
Host: www.trackingcompany.com/
User-Agent: Mozilla/5.0..Firefox/53.0
Referer: http://www.sitebeingtracked.co.uk/?nameofloggedinuser
If-Modified-Since: lastmodifiedheadervalue
If-None-Match: Etagheadervalue
```

Figure 6.8: Example request for a tracking pixel

The above example sends tracking information to the third party including user-agent, referer and last-modified and Etag values, using the request query string and referer query string to send additional information. The IP address of the device making the request may also be available to the the third party, which can be combined with the user-agent string as a means of user identification. It is likely any cookies set by an earlier HTTP response would be included in the request for the tracking pixel. Tracking pixels have also been found in webforms as a way of sharing data from the form with third parties [8].

6.7.5 Page elements: HTML script, iframe, image and anchor elements

Any HTML element with the ability to load from, or point to, an external server can be used in the same way as the pixel beacon describe above. The iframe is of particular interest here as it can contain a completely separate HTML document. As such, an invisible iframe can be embedded into the page and configured to load content from the third party tracking site [10]. This will be invisible to the user, and result in the third party's cookies being set in the user's browser without the user knowingly visiting the site.

An important aspect governing the security of iframes is the SOP. This ensures iframes execute in a third party domain context and prevents access to the main document, which prevents the setting of first party cookies [75, 14].

7 Fingerprinting

This section looks at mechanisms that do not need to store state to track or identify users, a process generally referred to as Fingerprinting.

Fingerprinting is a means of identifying users without the need for storing state on the user's device, making it an effective method even if cookies have been disabled or when PBM is in use. Many of the tracking methods already identified can be used as attributes of a fingerprint, including the user-agent string and other HTTP headers, as well as IP and Media Access Control (MAC) addresses.

7.1 Browser Fingerprinting

Possibly the most common use of fingerprinting is to identify a user's browser. This is not surprising, given the web browser is generally the user's access to the internet. For a site performing browser fingerprinting, when the browser connects to the web server to retrieve a webpage, the JavaScript fingerprinting code is downloaded and executed in the browser. The code queries the public attributes of the browser and uses a cryptographic function to produce a hash value of the results, which becomes the fingerprint of the device. Given sufficient entropy, such fingerprints are capable of acting as globally unique identifiers [76]. Here we will look at some of the attributes that can be used to generate browser fingerprints.

7.1.1 JavaScript engines

The implementation of JavaScript engines tends to differ between browsers, allowing the actual engine executing the JavaScript to be included in the fingerprint. A 2013 study by Mulazzani et al. demonstrated this approach as a way of reliably identifying the user's browser, even in the event of the user-agent string being spoofed by other means [77].

7.1.2 DOM objects

DOM objects can be used as attributes in browser fingerprints. Due to the differing implementations of JavaScript engines between browsers, the enumeration of a DOM object's method changes between different web browsers. To do this, the methods are retrieved from the DOM object, and the order in which they are returned is used as the fingerprinted attribute. Publicly accessible DOM objects, such as Navigator and Screen, are commonly used with this approach [15, 23].

7.1.3 Installed browser add-ons and extensions

Many users install add-ons to extend the functionality of their browser to increase security or protect privacy. Some browsers now block the enumeration of plugins as a privacy measure [78], however, many add-ons and extensions leak information into the DOM of the webpage, allowing JavaScript to be used to query the DOM for these telltale signs and identify well known add-ons and extensions [79, 76, 80]. Other add-ons and extensions add customer headers to outgoing HTTP requests, providing another means of identification [10, 81].

Browser add-ons and extensions can quite easily work against you if you find yourself in the unfortunate position of having installed a rogue one. Some rogue add-ons have been identified in the wild enabling device fingerprinting to be carried out using system attributes, such as the serial number of hard disks and installation date of the operating system [15]. Understandably, this would produce a very strong fingerprint and prove extremely difficult to defend against without making considerable changes to the underlying system.

7.1.4 Browser fonts

The fonts installed on the device is a popular attribute to include when fingerprinting browsers. The fonts within a web browser can be quite transient. For example, visiting a webpage which uses a custom font may prompt the user to download and install the custom font, thus increasing the uniqueness of the list of fonts available on the device. Font enumeration is made considerably easier given a little help from the Adobe Flash plugin [76, 15, 80].

7.1.5 HTML5 Canvas API

HTML5 introduced new graphics capabilities in the form of the `<canvas>` element [82]. Rendering graphics to the `<canvas>` element uses the device's underlying graphics hardware, which allows fingerprinting to include characteristics of the underlying graphics device, and

has become a common approach to browser fingerprinting [19, 81, 80]. As with JavaScript interpreters, browser graphics rendering engines tend to be bespoke to browser manufacturers, which provides the distinctive attributes fingerprinting relies on [80]. The general approach is to render graphics to a `<canvas>` element, typically strings of letters and numbers in different fonts, sizes and colours. Font effects such as colour depth and transparency can also be added to increase entropy. A cryptographic hash of the result is generated and becomes the fingerprint [19]. An example `<canvas>` is shown below in Figure 7.1 (Source: *Beauty and the Beast: Diverting Modern Web Browsers to Build Unique Browser Fingerprints* [81]):

Figure 7.1: Example canvas fingerprint

The `<canvas>` element may also be used as a replacement for font enumeration if Flash is not installed on the device. Although quite time-consuming, a list of different fonts can be drawn to a `<canvas>` element to identify the fonts available on the client's device, allowing a list of installed fonts to be determined [19].

7.1.6 HTML5 AudioContext API

The HTML5 AudioContext API provides similar benefits as the `<canvas>` element. As with the Canvas API, the AudioContext API relies on the underlying hardware, which again makes for an effective fingerprinting target [83]. Most devices have minor changes in configuration, such as hardware types and driver file versions, making them relatively unique even amongst similar version of the same device. To generate a fingerprint using the AudioContext API, different sounds are created using JavaScript, but rather than being played through the device's speakers they are sent to a software buffer which is cryptographically hashed to produce the fingerprint [84].

7.1.7 HTML5 Battery Status API

The final HTML5 API commonly found in fingerprinting code is the Battery Status API [83]. Unsurprisingly, this API gives statistics relating to the device's battery, such as charge level and whether the device is currently running on battery or is plugged into mains power [85]. Due to the transitive nature of the battery API it's unlikely to be a reliable attribute for longer term fingerprinting, as the fingerprint would change if, for example, the user plugged their device into the mains. However, it has been shown as effective in fingerprinting devices over shorter timescales such as sessions [20].

7.1.8 Emojis

Emojis are a good example of how fingerprinting relies on the differences between devices and browsers to generate a reliable fingerprint. Laperdrix et al demonstrated how many platforms create bespoke graphic emoticons which can be used as a way of platform identification via fingerprinting [81]. An example of this can be found in Appendix A.

7.1.9 SSL/TLS handshake fingerprinting

For HTTPS network traffic, it is possible to consistently identify users by fingerprinting the parameters exchanged as part of the SSL/TLS handshake protocol, such as the cipher suite details [86]. This method has also proved reliable in guessing the user-agents involved in the exchange [86].

7.1.10 WebRTC fingerprinting

WebRTC, a protocol which allows real-time communication between web browsers, assigns a unique device ID for each multi-media device found in the system [87]. As these IDs are unique to each device, such as a camera or microphone, they can be used as fingerprinting attributes [80]. Studies have found different browsers treat device IDs differently, with some browsers renewing device IDs with each browser session, making them less useful for tracking purposes [80].

In addition, WebRTC has been found to leak the local IP address of devices, and can be used to track users even when anonymising technologies such as Virtual Private Networks (VPNs) are in use [88].

7.2 Behavioural Biometrics

A more recent approach to user identification comes in the form of behavioural biometrics. Behavioural biometrics identify individuals based on their interaction with devices, for instance, the fingerprint reader on an Apple iPhone or the unique characteristics of the way the user types or uses a mouse [40, 41]. The image in Figure 7.2 below shows example modalities used for active authentication (*source: BehavioSec* [42]):

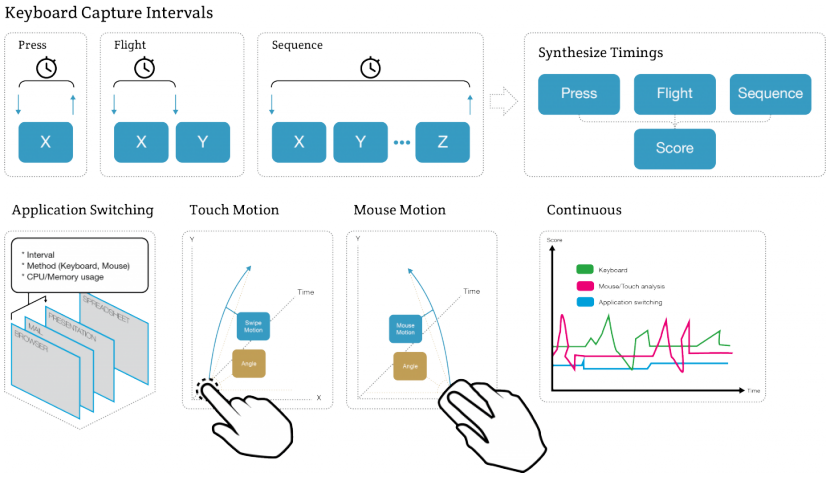


Figure 7.2: Example modalities of behavioural biometrics

Behavioural biometrics have a distinct application as an out-of-band second factor authentication mechanism and have the potential to help prevent fraud and account takeover by means of active authentication of the user [89]. Active authentication also makes for a viable stateless tracking mechanism [42]. Profiling users in this way is particularly potent, potentially having the ability to deanonymise The Onion Router (TOR) users [90], and would permit advertisers another way to track users regardless of the devices they choose to use to access web-based services.

8 Summary

That completes our look at the technology and approaches used to perform tracking. In this part, we've looked at a considerable amount of tracking mechanisms, from the humble HTTP cookie to highly persistent versions with the ability to resurrect other cookies from the dead. We also know how simple techniques for exchanging data can be based on the name given to a browser window, and how difficult it can be defend against the different forms of fingerprinting. From here, we will go on to look at the defences presented by academia, before taking a selection of these defences and testing them to find out how effective they are in practice.

III Defences

9 Defences Based on the Web Browser

Over the next few chapters we will look at the choices available to help defend against web tracking. As demonstrated in the previous part of this study, tracking techniques can be flexible enough to overcome certain limitations put in place by users, such as the blocking of third party cookies, or using private browsing mode. Such a layered approach to tracking requires a similar defence-in-depth approach to mitigation. To make it easier, the next few chapters will present defences categorised by their ease of implementation. We will start off by looking at easily implemented defences based on the web browser, placing them in reach of the majority of users regardless of technical ability.

9.1 Browser Selection

Web browser selection is likely the first line of defence for most users. Current web browsers tend to feature built-in privacy features as standard, such as private browsing mode or Firefox's Tracking Protection [91], as well as the ability to implement security and privacy further through configuration. There are a considerable number of offerings in the web browser market, with the three most popular being Google Chrome, Microsoft Edge and Mozilla Firefox [92]. Perhaps the most practical approach is to select one of the well-known browsers as a starting point.

9.1.1 Advantages

The browser's in-built tracking protection mechanisms are usually easy to configure and offer good protection. Some browsers, such as Mozilla's Firefox, are also open-source [93], meaning the underlying source code is under constant review which tends to result in improved functionality and security. Another benefit of selecting one of core browsers is the convenience with which add-ons and extensions can be installed to extend the functionality of the browser.

9.1.2 Disadvantages

The effectiveness of particular features can differ from browser to browser, so some level of research is still required to ensure the selected browser meets your needs. Private browsing mode is an example of how features can differ by manufacturer, as demonstrated by Aggarwal et al in their study *An Analysis of Private Browsing Modes in Modern Browsers* [25].

9.2 Tracking Protection (TP)

Many browsers now come with some form of tracking protection by default. A good example of this is the Tracking Protection feature of Mozilla's Firefox which is tested later in this study. Tracking protection tends to work in a similar way to list-based privacy add-ons, by comparing the requested URL of each outgoing request to a list of known trackers and hidden nodes, and blocking the request or warning the user if the URL is on the list.

9.2.1 Advantages

Having tracking protection built into the browser means privacy protection is available without the need to install any add-ons or extensions. For Firefox, the Tracking Protection feature can be configured so it is active all the time or only in Private Browsing Mode. Firefox uses Tracking Protection Lists (TPLs) from the Disconnect privacy add-on, offering both standard and strict TPLs [94].

9.2.2 Disadvantages

One disadvantage is that tracking protection may not be enabled by default, meaning the user needs to be aware of the feature in order to enable it. Enabling tracking protection may also have adverse effects on the page, such as preventing content from loading or breaking some functionality. This downside is not unique to tracking protection, with many defences breaking the functionality of web page as a result of blocking requests to third parties.

9.3 Private Browsing Mode (PBM)

Not so much a direct defence against tracking, private browsing mode is designed to protect the browsing history on the local machine rather than prevent the user from being tracked whilst browsing [30, 43]. All three of the main browsers offer some form of private browsing mode.

9.3.1 Advantages

Private Browsing Mode reduces the amount of client-side state stored, particularly cookies, and enforces privacy protections automatically when the browsing session ends, such as clearing cookies and other stored state. PBM can be used in combination with defences such as tracking protection, and can also be configured to be used by default in many web browsers.

9.3.2 Disadvantages

Not all private browsing modes are made equally, with the effectiveness of PBM being reliant on the browser being used [43].

9.4 Extending Browser Capabilities with Add-Ons and Extensions

One of the most convenient methods for implementing protection against tracking is by installing pro-privacy browser add-ons and extensions [1, 3, 28, 8, 17, 43, 95]. Browser add-ons and extensions allow a user to easily extend the capabilities of the web browser by installing new functionality. Once installed, each HTTP request will be directed through the add-on allowing the request to be inspected and altered or blocked if needed. All three of the main browser manufacturers offer web stores of add-ons and extensions for easier discovery and installation, as well as the functionality to manage add-ons and extensions from within the browser.

9.4.1 Advantages

Add-ons and extensions can offer good tracking protection while being simple to install. The ease with which add-ons can be added and removed means users are able to test different functionality and configurations quickly and easily. The support of plug and play architectures within browsers allows users to create custom add-ons tailored to their own specific needs. Add-ons can offer fine-grained control over the browser's default settings, such as disabling JavaScript on a page-by-page basis rather than having to resort to disabling JavaScript altogether.

9.4.2 Disadvantages

Although add-ons can offer good levels of flexibility and protection against tracking, there are often costs associated with this convenience. With large numbers of free browser extensions available, it can be a challenge to locate and evaluate add-ons, all of which takes time and

effort on the part of the user [11]. As there are no limitations on the development of browser extensions there may be a lack of formal testing, security evaluation or approach to patching bugs, which can result in add-ons becoming targets for attack.

Perhaps more concerning is the ability for fingerprinting scripts to identify add-ons via the structural changes they make to the webpage, and use the details of the add-ons as part of the fingerprint. In this case, rather than protecting the user, add-ons start to erode their privacy by making them more trackable [95, 80].

To make matters worse, add-ons operate in a privileged environment with the ability to inspect and alter HTTP requests and responses, giving them access to the web traffic flowing in both directions. As a result, add-ons can have access to sensitive information such as PII, making them an attractive proposition for misuse [96].

9.5 Removing the Adobe Flash Plugin

As seen in the *The Adobe Flash Plugin* section earlier, Flash is particularly useful for tracking. It allows cookie respawning using Flash LSOs, is used by Evercookies to persist data, and is also a convenient way to enumerate installed fonts when browser fingerprinting.

9.5.1 Advantages

Removing the plugin prevents the use of Flash LSOs for tracking and cookie respawning [43] and Flash LocalConnection object from sharing information between Flash movie files. Removing Flash also helps reduce the fingerprintable surface of the web browser [15].

9.5.2 Disadvantages

Removing the Flash Plugin may result in a loss of media functionality on some websites.

10 Defences Based on Web Browser Configuration

The next category of defences we will look at are those based on web browser configuration. As with *Defences Based on the Web Browser* these defences require very little technical knowhow to implement and should therefore be in reach of most users, regardless of their technical prowess.

10.1 Disallowing Cookies

Restricting the setting of cookies offers the user some protection against being tracked, particularly when setting the browser to reject third party cookies by default [3, 14, 43]. All three of the main web browsers provide cookie management capabilities allowing the user to reject cookies by default or clear cookies automatically when the browser closes. In addition, many browser add-ons and extensions are available for managing cookies.

10.1.1 Advantages

Preventing cookies from being set, particularly third party cookies, reduces their ability to be used for tracking purposes.

10.1.2 Disadvantages

As users start to deny the setting of third party cookies, the use of first party cookies increases. Denying first party cookies may result in a loss of functionality offered by a service, meaning the user may have to accept some tracking as the cost of using the service.

10.2 Disabling JavaScript

Disabling JavaScript is potentially one of the most effective defences against web tracking, but also one of the most inconvenient due to the inevitable loss of functionality experienced when visiting websites with JavaScript disabled [3, 18, 14, 43]. For the three main browsers,

JavaScript can be disabled using browser settings or by installing an add-on or extension such as NoScript Security Suite[1].

10.2.1 Advantages

Many of the mechanisms used to track users rely on JavaScript in some form, such as cache and storage manipulation as well as the more insidious cookie types like Evercookies [43]. Possibly the main benefit of disabling JavaScript is the reduction/prevention of the ability to fingerprint the web browser [95].

10.2.2 Disadvantages

The main disadvantage of disabling JavaScript is the inevitable loss of functionality the user will experience as most of today's web has a heavy reliance on JavaScript [3, 14, 18]. This impact can be reduced to some degree by using an add-on such as NoScript Security Suite which allows selective control over enabling/disabling JavaScript on a per-page basis.

A second disadvantage with disabling JavaScript is the fact it does not prevent tracking, only some forms of tracking. Users can still be tracked quite easily even with JavaScript disabled. A common workaround is to embed a `<noscript>` element containing a tracking pixel into the webpage as a replacement for the loss of JavaScript [14].

10.3 Disabling the Referer Header

Disabling the referer header will help prevent leakage of sensitive information included in the URL of a site being passed to a third party. It will also help prevent browsing history being determined using the referer header. The main three browsers allow the referer header to be disabled either via configuration, a command line switch or by installing a specified browser extension.

10.3.1 Advantages

Disabling the referer header prevents it being used as a means of leaking browsing activity and potentially sensitive information.

[1] https://noscript.net/

10.3.2 Disadvantages

May result in loss of functionality if the referer header value is required in order to provide some form of service. For instance, it is common to check the value of the referer header as a means of preventing cross-site request forgery [97].

10.4 Disabling WebRTC

WebRTC provides two tracking mechanisms: unique IDs for media devices which can be used as part of fingerprinting, and the potential leakage of local IP address when behind a VPN. The easiest approach to preventing these two tracking mechanisms is to disable WebRTC in the browser via settings or by installing a browser add-on to help manage WebRTC.

10.4.1 Advantages

Disabling WebRTC will prevent tracking via media device IDs and local IP addresses.

10.4.2 Disadvantages

Disabling WebRTC will likely result in the loss of some media functionality.

10.5 Enabling Do Not Track (DNT)

Do Not Track (DNT) is a HTTP request header managed by the browser to indicate the user's wish to opt-out of being tracked [3, 14, 11, 23, 43, 18]. As DNT is managed by the web browser, it can be enabled via browser configuration for the three main browsers.

10.5.1 Advantages

As DNT is managed by the browser it is simply a case of enabling the feature and letting the web browser handle sending the header with each request. Enabling DNT signals your wish to opt-out of tracking, allowing sites honouring the DNT request to set opt-out cookies as a future reminder to not perform tracking.

10.5.2 Disadvantages

Lack of legislation and law governing DNT means trackers can ignore the flag without penalty. DNT has no means of actually preventing tracking, and relies on an ethical approach by

trackers to honour the request [14, 11, 23]. Enabling DNT may also result in an increase in cookies being set, and the header can also be used as a browser fingerprinting attribute.

11 Technical Defences

The final batch of defences we present are the most technical, requiring a level of expertise and understanding to implement and use effectively. Not for the faint hearted, these defences are included for completeness.

11.1 Virtual Private Networks (VPNs)

Virtual Private Networks provide security by encrypting the connection over which traffic flows, making the traffic appear scrambled to eavesdroppers [3, 43]. VPNs can be setup and configured locally, however the most convenient approach is to use a web-based VPN service.

11.1.1 Advantages

VPNs provide security as well as privacy by ensuring all traffic is encrypted between the user and the VPN service. Prevents IP tracking by ISPs as the ISP sees only the IP address of the VPN service provider, and not the details of the URLs requested by the user [43].

11.1.2 Disadvantages

There is potential for the VPN service to track users and sell their browsing history. VPNs can also be weakened by other technologies, such as WebRTC leaking the user's local IP address from behind the VPN service, so a level of understanding is required to use the technology effectively.

11.2 The Onion Router (TOR)

TOR is a network of servers designed specifically to provide privacy and anonymity. It uses multiple layers of encryption and privacy protection to protect network traffic as it passes from node to node, which masks the source and destination of the traffic [98]. TOR is most

commonly accessed using a specially adapted version of Mozilla Firefox, available from the TOR project[1].

11.2.1 Advantages

TOR is potentially the most effective way to defend against online tracking, including fingerprinting [43, 95], as TOR is built specifically to provide anonymity online and contains anti-tracking provisions [99].

11.2.2 Disadvantages

A level of technical knowledge and understanding is required to us TOR effectively without compromising the protections offered. In addition, TOR has also been proved comprisable in recent years [100].

[1] `https://www.torproject.org/`

12 Summary

We're now starting to get to the crux of the study, having reviewed the reasons and implications of being tracked as well as the mechanisms used to perform tracking. In this part, we reviewed a large selection of defences and divided them into three categories: *Defences Based on the Web Browser*, *Defences Based on Web Browser Configuration* and *Technical Defences*. For each defence we looked at advantages and disadvantages to help provide a little more colour. In the next part, *Testing Browser Based Defences*, we will select a number of defences from the list, build an automated testing framework in Python and test the defences against a selection of the sites known for performing tracking.

IV Testing Browser Based Defences

13 Testing Defences

So far we've looked closely at the impacts and outcomes of being tracked, how the trackers go about their craft, and also compiled a comprehensive list of defences. We will now start getting to the nitty gritty of the study by taking a handful of defences and testing them to evaluate their effectiveness. All of the defences chosen are easy to implement and need next to no technical understanding or expertise to provide good, solid privacy protection.

Before continuing, it's appreciated that words such as *easy* hold different meanings for different people, so it would be worthwhile clarifying this word by defining a target audience. All of the defences tested are browser based and consist of either changes to settings and configuration, or installing a browser add-on from the browser's online store. Anyone capable of using a web browser to perform a search and follow simple step-by-step guides should be comfortable implementing, understanding and using the defences tested.

13.1 Selecting Defences

The earlier part *Defences* presented a wealth of information regarding the options available to defend against tracking. Here, we will take two categories from that section, *Defences Based on Web Browser Configuration* and *Extending Browser Capabilities with Add-Ons and Extensions* and test how they can effect your online privacy. Englehardt and Narayanan found browser configuration and browser add-ons to be practical forms of defence [17], which provides a good basis from which to work. Evaluating browser configuration allows us to easily incorporate a number of other defences featured in the *Defences* section into our testing, including enabling Do Not Track, disabling JavaScript and disallowing cookies.

Before deciding which configuration items and browser extensions to evaluate, decisions need to be made regarding what to actually test, and the criteria for judging effectiveness. As seen in the section *Stateful Tracking Mechanisms*, HTTP cookies are the most common way to perform tracking, so a measure of persistence of HTTP cookies would seem a sensible starting point. The *Stateful Tracking Mechanisms* chapter also distinguished between first and third party cookies and their relevance to tracking. To save backtracking, here's a gentle reminder. Every web transaction features at least two parties: the user requesting the webpage and the

website providing the page, known as the first party. A large number of websites today rely on content served by others, such as videos hosted by YouTube or Vimeo, or web forms or surveys managed by external companies. Displaying this content introduces a third party into the mix, and this party often brings with them cookies and a considerable amount of overhead in additional HTTP requests for content. These third party cookies allow the content provider to track users across the websites they visit by storing unique user identifiers within the cookies and then retrieving the cookies from each site loading content provided by the cookie owner.

To capture the activity of third parties, we can record all the HTTP traffic exchanged between the user's web browser and the first and third parties and use this information as an evaluation criteria by looking inside the HTTP requests for the telltale signs of tracking. We will come back and take a closer look at these criteria shortly, however that is sufficient information to define which specific defences to test.

13.2 Defences Based on Browser Configuration

Disabling cookies would seem a sensible starting point. This will be split into two tests and evaluated separately by first disabling third party cookies and then all cookies. Englehardt and Narayanan used this as a criteria in their study of tracking spanning a hefty one million sites [17].

Other studies such as those by Roesner et al [14] and Mayer and Mitchell [101] put Do Not Track (DNT) under the spotlight. It will be interesting to include DNT in the tests as it's not a defence as such, more a wish, as it relies on ethical compliance to honour the user's request to not be tracked.

Many of the popular browsers now incorporate some form of tracking protection by default. The lead of Kontaxis and Chew [26] will be followed and Firefox's Tracking Protection will be included in the tests. Firefox's Tracking Protection is based on Tracking Protection Lists (TPLs), and uses the same TPLs as the Disconnect add-on also selected for testing [94]. In this case, the standard TPL will be tested opposed to the stricter version of the two.

Firefox offers Tracking Protection as a standalone defence and also as a configuration option for Private Browsing Mode (PBM), so both will be tested. Including PBM should be an interesting test, as it's predominantly a protection against discovery of browsing activity on the local machine, rather than setting out to prevent tracking [30, 43].

The final configuration changes to be added to the list are disabling JavaScript and disabling the referer header. Disabling JavaScript has been found to be an effective deterrent to tracking, but also one of the most troublesome to work with as it's likely to break functionality

on the webpage [3, 18, 14, 43]. The *Referer header* and *Tracking Pixel a.k.a Web Beacon, Web Bug, Pixel Tag or Pixel Beacon* sections show how the referer header can be used to leak information to third parties, so examining the effects of disabling it is a logical test to conduct. Disabling JavaScript and the referer header brings the total number of configuration changes to seven. Table 13.1 summarises the defences based on configuration changes:

Defences Based on Configuration Changes Selected for Testing
Disabling third party cookies
Disabling all cookies
Enabling Do Not Track (DNT)
Enabling Tracking Protection (TP)
Enabling Tracking Protection and Private Browsing Mode (PBM)
Disabling JavaScript
Disabling referer header

Table 13.1: Defences based on configuration changes selected for testing

13.3 Defences Based on Browser Add-Ons

Now the configuration changes have been selected, it is time to take a closer look at which add-ons to test. The relevant literature in this area is a good place to start. Five extensions are commonly mentioned: Ghostery [26, 17], Disconnect [26, 17], AdBlock Plus [26], UBlock Origin [17] and Privacy Badger [26]. Out of these, four are based on Tracking Protection Lists (TPLs) so need no learning period before providing protection. The fifth, Privacy Badger[1], uses an heuristics based approach which requires a learning period to become really effective. Unfortunately, it's not practical to provide Privacy Badger with this learning period as the tests will be conducted on clean Firefox profiles using virtualisation so the environment can be returned to a known clean state after each test, which would reset Privacy Badger to it's original state.

Ghostery[2], Disconnect[3], AdBlock Plus[4] and UBlock Origin[5] are all installed directly into the web browser and work by monitoring outgoing HTTP requests and comparing them to lists of known trackers. If the HTTP request is to a member on the list, the request is blocked. Ghostery incorporates a second mode which only monitors the outgoing HTTP requests and

[1] `https://www.eff.org/privacybadger`
[2] `https://www.ghostery.com`
[3] `https://disconnect.me`
[4] `https://adblockplus.org`
[5] `https://www.UBlock.org`

informs the user of matches, but does not block the request by default. Users need to be aware of the default mode, otherwise Ghostery will not be configured to block tracking requests.

One final add-on will be added to the list. NoScript Security Suite[6] allows more control over the scripts executing on the page compared to disabling JavaScript in the browser completely. Trusted sites can be added to a whitelist so functionality is not affected, whilst retaining the default mode of blocking scripts executing from untrusted sources. Table 13.2 summarises all the add-ons selected for testing:

Defences Based on Browser Add-Ons Selected for Testing
Ghostery, configured in blocking mode
Disconnect
AdBlock Plus
UBlock Origin
NoScript Security Suit

Table 13.2: Defences based on browser add-ons selected for testing

13.4 Browser Selection

As all the defences selected are browser based, Mozilla's Firefox was selected as the basis for the defences. This decision was made for a number of reasons. First, Firefox has a built-in automation engine, Marionette [102], which allows the browser to be driven using Python [103]. Marionette provides extensive functionality to drive Firefox and manipulate the page being tested, such as entering text into controls and clicking buttons which permits actions such as logging into accounts or executing JavaScript directly in the page.

Another significant benefit of Firefox is it's configurability. Firefox supports the creation of multiple profiles, with each one segregated from the others, using a utility known as the Profile Manager. The Profile Manager was used to create thirteen separate profiles: one profile for each defence and one profile to act as a benchmark against which the defences can be compared. After creation, each profile was configured to a default state using a script to ensure all the profiles were identical apart from the defence being tested (see Appendix E). Once each profile was configured, the associated defence was installed.

The final reason for selecting Firefox is the use of SQLite databases to store session data. SQLite can be queried using standard SQL syntax, allowing complex queries to be created if

[6] `https://noscript.net`

required. SQLite also provides a Python driver, making it very easy to query databases from scripts.

13.5 The Crawler Framework

A bespoke framework was written in Python to drive Firefox using the Marionette driver. Dubbed Crawler, the Framework automates all the tasks needed to perform the tests, including managing the logging of HTTP requests, requesting webpages, signing into social network accounts, querying the Firefox SQLite databases and analysing results before converting them into a format understandable by a spreadsheet. The framework is available on GitHub at `https://github.com/darrelln/crawler/`.

13.6 Test Data Set

In order to test how well the chosen defences work, a number of test URLs are needed. Ideally, these should be URLs with a known history of using tracking and behavioural advertising. One candidate fits this description perfectly: the news site.

13.6.1 News Sites

In their study, *Online tracking: A 1-million-site measurement and analysis*, Englehardt and Narayanan found news sites regularly attracted a considerable number of trackers [17]. To some degree, this is to be expected as they use advertising to help fund free content for readers [26, 104]. One very recent study by Merzdovnik et al went a step further in their investigation of news sites, finding more trackers on news articles rather than the landing pages [105]. With this in mind, news sites would appear the perfect candidates for testing. A news article from each of the ten most popular news sites ranked by Alexa[7] was selected as the test data set. The full list of URLs can be found in Appendix C.

13.6.2 Potential Trackers and Social Networks

To provide a clearer picture of tracking taking place, two additional sets of data were added to the test data set. The first set consists of the 'known trackers' identified by Englehardt and Narayanan in their *Online tracking: A 1-million-site measurement and analysis* study [17], from here on referred to as *potential trackers*. The second set contains the domains of

[7] `http://www.alexa.com/topsites/category/news`

the Top Five social networks, according to Alexa[8], and expanded slightly to include a .co.uk URL if available. The inclusion of social networking domains in the test data set is based primarily on studies carried out by Krishnamurthy and Wills on the leakage of information stemming from social networks [2, 3]. The perspective taken for this study is to assess to what extent the defences block the sharing of information such as cookies with social networks, for example, via requests made to social networks containing cookies or leaking information via the referer header. To encourage tracking activity from the social networks, an account was created with each of the Top Five, namely Facebook, Twitter, Google+, LinkedIn and Pinterest, and logged into at the start of each test. The full details of both sets are shown in Appendix D.

13.7 Criteria for Evaluation

As mentioned in the *Selecting Defences* section, the main criteria we will use to judge effectiveness are the reduction in cookies and the reduction in HTTP requests.

13.7.1 Reduction in cookies

The total number of first and third party cookies will be counted and compared against the benchmark values. This approach is similar to the one used by Kontaxis and Chew to evaluate Firefox's Tracking Protection [26], however we will extend this by subdividing third party cookies further to assess whether they belong to either the set of known potential trackers or social media sites. This will allow us to examine whether potential trackers and social media sites are able to continue tracking with the defence in place.

13.7.2 Reduction in HTTP requests

The second criteria looks at the HTTP requests made between the user and first and third parties. Assessing HTTP requests will follow a similar approach to the evaluation of cookies by measuring the number of requests, again taking a close look at the requests flowing between the potential trackers and social networks. Having access to the HTTP requests means we can peek inside for signs of information leakage via the referer header and request URL. We will evaluate this by looking at the number of requests to potential trackers and social networks which include a cookie as part of the request, and also those requests including a cookie and mentioning the first party in the referer header or request URL.

[8] `http://www.alexa.com/topsites/category/Computers/Internet/On_the_Web/Online_Communities/Social_Networking`

13.8 Test Environment and Methodology

Thus far, we have defined which defences to test along with the criteria against which they will be judged. The final piece of the puzzle is how the tests will be conducted.

13.8.1 Test environment

A Linux virtual machine (VM) was created based on Oracle's Virtual Box[9] and the Linux Mint distribution[10]. Linux Mint is a relatively light, fully-featured distribution making it ideal for use as a guest VM. Helpfully, it comes complete with Mozilla Firefox and Python installed by default.

13.8.2 Test methodology

A completely automated approach to testing was taken, based on the Crawler framework introduced earlier (see section *The Crawler Framework*). As a refresher, each defence was installed in a separate Firefox profile. This approach provides the benefit of segregation: each defence being tested is segregated from all others, as are the results of each test. For each Firefox profile, Crawler first signs into the five social network accounts before iterating through each news article in turn. Once all the profiles have been exhausted, the results of the tests are analysed and formatted into a CSV file for further investigation in a spreadsheet. Table 13.3 shows the complete test steps in pseudocode:

[9] `https://www.virtualbox.org/`
[10] `https://www.linuxmint.com/`

Start virtual machine

For each Firefox profile:

    Start Firefox with the profile under test and Marionette support

    Enable HTTP request logging for social networking requests

    For each social network:

        Sign in social network account

    End for

    Disable HTTP request logging for social network requests

    Enable HTTP request logging for test requests

    For each test URL:

        Request URL and wait for page to fully load

    End for

    Disable HTTP request logging for test requests

    Close Firefox and marionette

    Analyse HTTP requests

    Analyse Firefox cookies

    Produce results

End for

Create CSV results file

Table 13.3: Pseudocode representation of test steps

14 Evaluation and Appraisal

With our tests complete, we can now evaluate the results and appraise how well the defences performed. In total, thirteen individual tests were completed: seven tests based on web browser configuration, five tests based on web browser add-ons and one test with no defences installed to act as a benchmark.

14.1 Results of Benchmark Test

To provide a basis from which to evaluate test results, the first set of numbers we will look at is the results of the benchmark test. As a reminder, a benchmark test was included based on a Firefox profile with all forms of protection against tracking disabled (see Appendix E for the Firefox configuration script). The benchmark test produced a total of 549 cookies, of which 136 were first party and 413 were third party cookies. Of the 413 third party cookies, 53 had a domain in the potential trackers list, and 64 had a domain from the social networks list. This is a ratio of just over 3:1 third party cookies to first party cookies. Figure 14.1 shows this breakdown of cookies by owner.

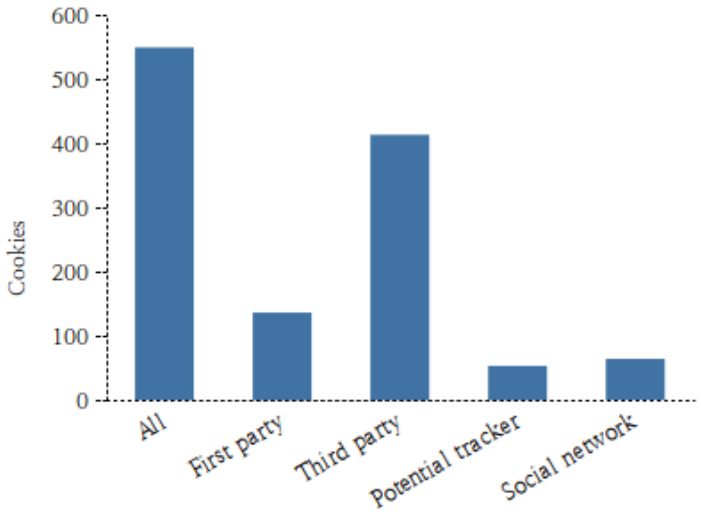


Figure 14.1: Breakdown of cookies by owner

The benchmark HTTP requests, shown in Figure 14.2 followed a similar trend, if not magnified. Out of 3,551 requests, only 351 were first party with the remaining 3,200 being requests to third parties, producing a ratio of 10:1 third party to first party requests. This may explain why some URLs in the test data set took almost five minutes to fully load.

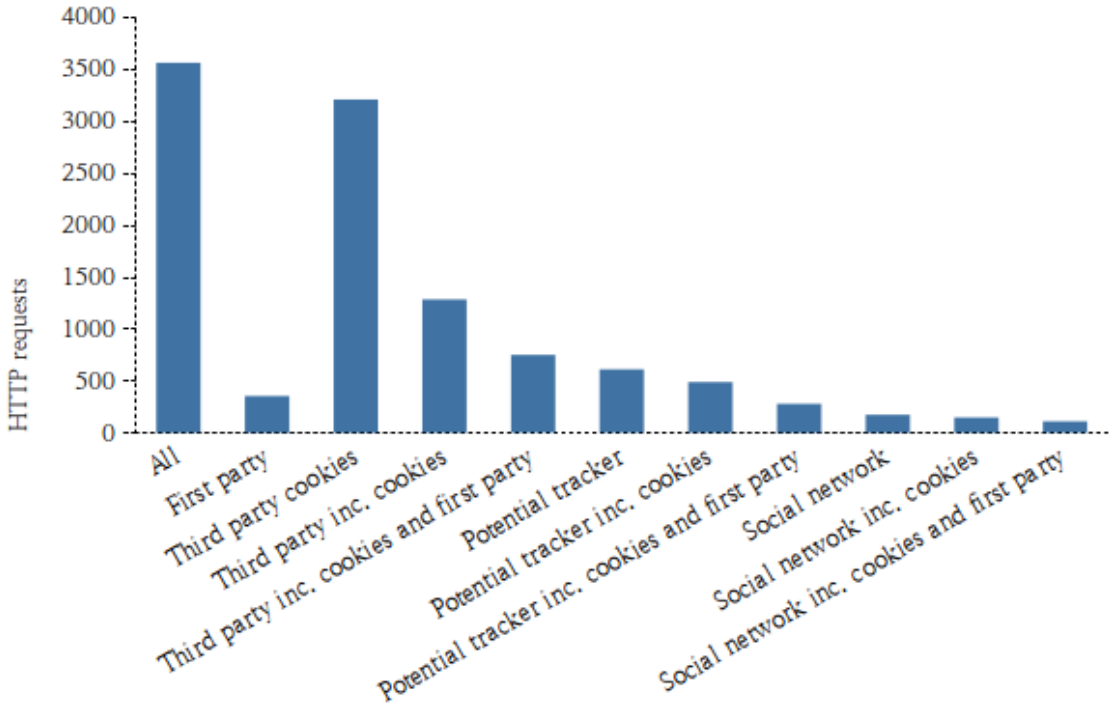


Figure 14.2: Breakdown of HTTP requests by owner

Figure 14.2 also shows a breakdown of third party requests including a cookie, and third party requests including a cookie and details of the first party in either the referer header or request URL. Interestingly, although requests were made to social networks including a cookie and first party details (i.e. the referer header or request url mentioned a first party by name), the total number of requests is quite small. These requests can be seen at that far right hand side of Figure 14.2, under the heading *Social network inc. cookies and first party.*

The benchmark results show a strong bias in favour of third parties, for both cookies and HTTP requests. As highlighted in separate studies by Kontaxis and Chew [26] and Schelter and Kunegis [104] news sites are a prime example of free content subsidised by advertising revenue, which results in large numbers of third parties tracking the articles read by visitors.

14.2 Results of Tests Based on Browser Configuration

14.2.1 Reduction in cookies

With the benchmark results defined, we can start to evaluate how well the defences performed, starting with those based on web browser configuration. Figure 14.3 contains the results related to cookies:

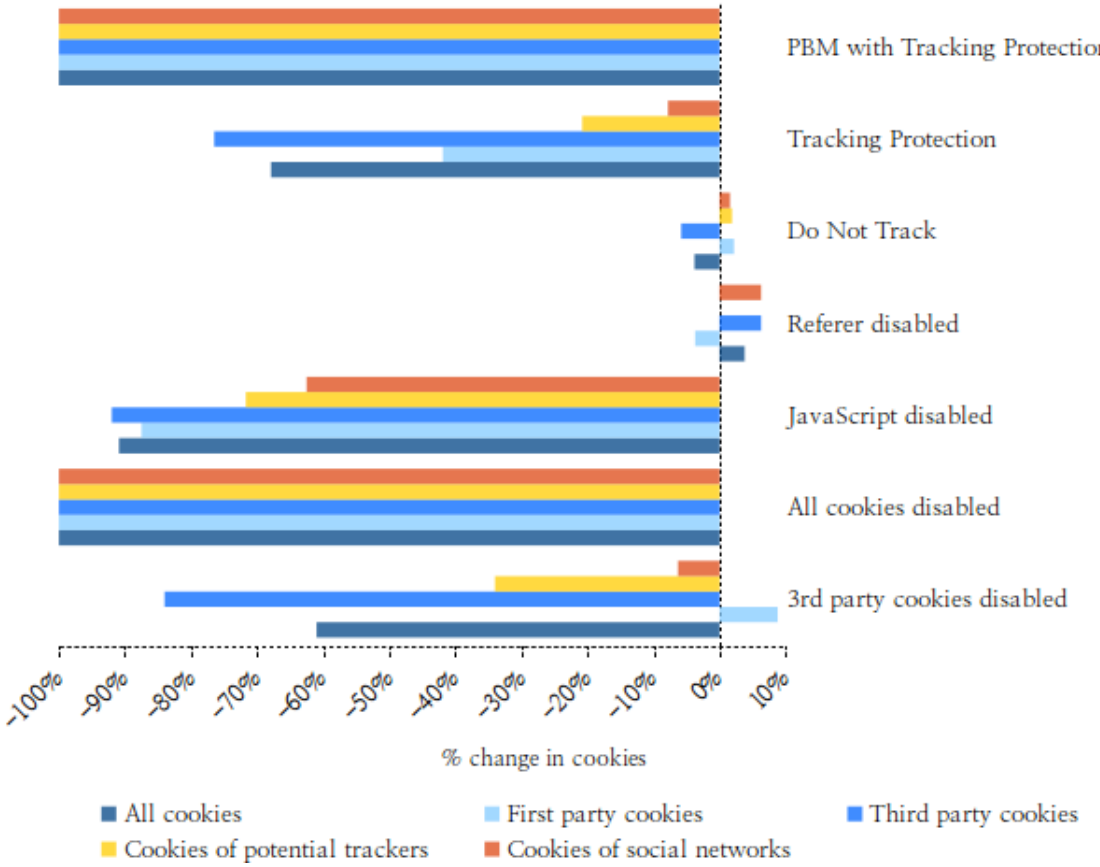


Figure 14.3: Effect of web browser configuration on cookies

Figure 14.3 shows the most effective defence to reduce cookies was to simply disable them completely. Unfortunately, disabling cookies completely had adverse effects on the functionality of many of the pages in the test, such as preventing log in for most of the social network accounts. Disabling JavaScript had a similar effect on functionality, and produced a less pronounced reduction in cookies to boot. It's worth bearing in mind the tests were conducted in some resemblance of isolation as the evaluation criteria is quite narrow, looking only at the reduction in cookies and HTTP requests. Evaluating the effectiveness of disabling JavaScript against such narrow criteria fails to take into account possible effects it would have on other

tracking technologies, such as it's assumed effectiveness at reducing browser fingerprinting.

Firefox's Tracking Protection (TP) reduced third party cookies by 77% without having an adverse effect on functionality, however cookies owned by potential trackers and social networks only reduced by 21% and 8% respectively. The impressive 100% reduction in cookies shown by using Tracking Protection in conjunction with Private Browsing Mode (PBM) is a result of PBM clearing cookies automatically when the browser is closed. This could be viewed in two ways: either an effective defence as no cookies were stored once the browser closes, or a defective test, again because no cookies were stored. We would be inclined to view the results on par with those produce by using Tracking Protection in isolation, along with the added benefit of cookies being cleared automatically by Firefox when the browser closes.

Surprisingly, three of the configuration changes tested resulted in an increase in cookies. Disabling the referer header and enabling Do Not Track (DNT) both had an adverse effect and increased cookies. What was not expected was the 9% increase in first party cookies when disabling third party cookies. The reasons for this increase are unclear, particularly as disabling third party cookies produced a corresponding reduction in HTTP requests, although it may be the case that DNS canonical naming records (CNAME) were in use [1, 84]. Another surprise was the effect disabling third party cookies had on third party cookies: it did not reduce them by anywhere near 100%. In fact, third party cookies were reduced by 84% and social network cookies by a meagre 6%.

14.2.2 Reduction in HTTP requests

All the defences based on web browser configuration reduced the overall number of HTTP requests. As expected, some defences produced better results than others, with Tracking Protection, both standalone and when combined with Private Browsing Mode, and disabling JavaScript reducing third party requests by 80% and 94% respectively, as can be seen in Figure 14.4:

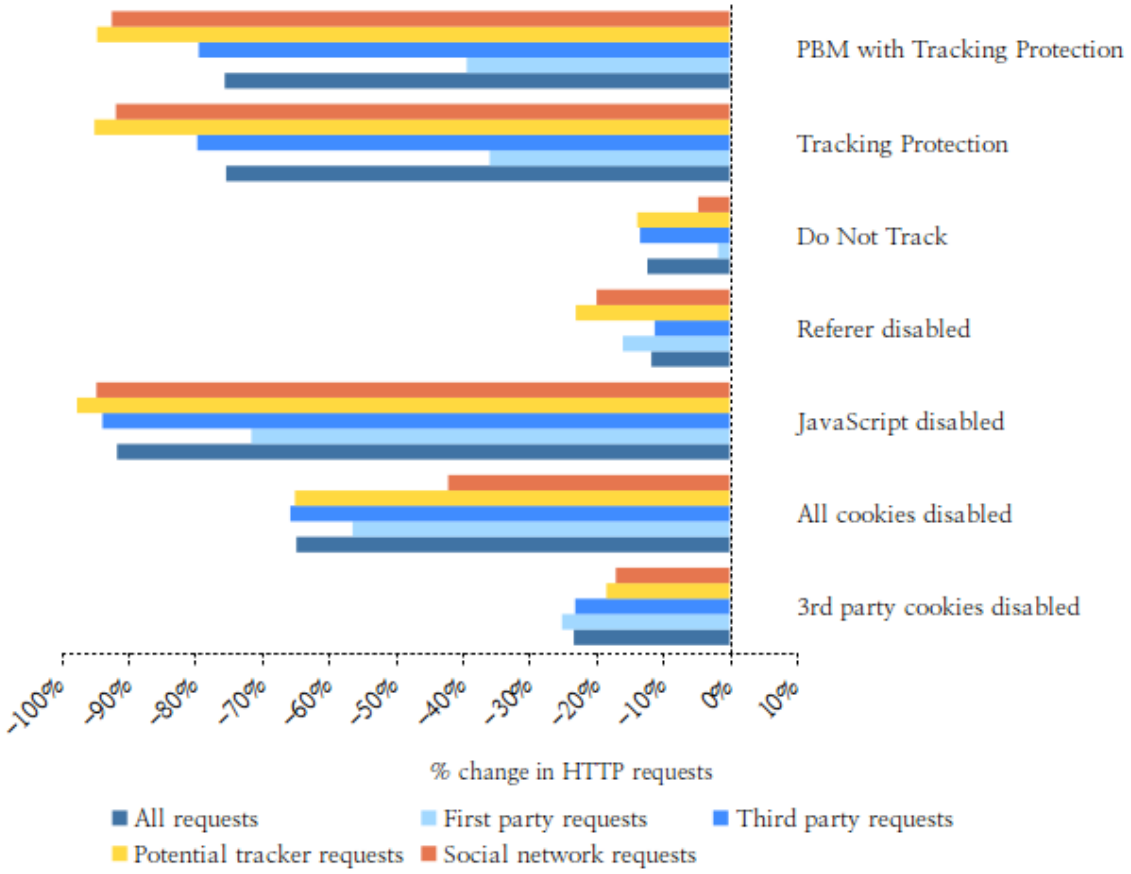


Figure 14.4: Effect of web browser configuration on HTTP requests

Enabling Do Not Track and disabling the referer header both produced a reduction in HTTP requests, however the reductions are not as strong as those seen by disabling JavaScript or enabling Tracking Protection.

Completely disabling cookies reduced third party requests by 66%, however it had adverse effects on the page's functionality, so a percentage of the drop in requests could well be due to the page failing to fully load. In many ways, assessing HTTP requests in relation to cookie-based defences may well be seen as trying to compare apples with oranges, however it's clear that reducing cookies does effect the number of HTTP requests made to third parties.

14.2.3 Reduction in HTTP requests including cookies and first party

We will now take a closer look at HTTP requests for signs of information leakage to identify requests to third parties that include a cookie, and also those that include a cookie and mention any of the first parties in either the referer field or as part of the requested URL. What we're looking for in this instance is an indication of information leakage using these

mechanisms rather than to identify all occurrences of leakage, as there are simply too many ways information could be obfuscated to make it unobservable in a request URL.

With the exception of Do Not Track (DNT), Figure 14.5 would indicate all the defences based on browser configuration did a good job of reducing information leakage. This may not be as pretty a picture as we'd hoped, as the number of requests potentially leaking information is quite small compared to the total number of requests made. For example, out of 3,200 third party requests, 107 were potentially leaking information to social networks, therefore an overall reduction in third party requests may have unwittingly prevented requests that would have otherwise leaked information, rather than the defence knowingly blocking the requests to prevent information leakage.

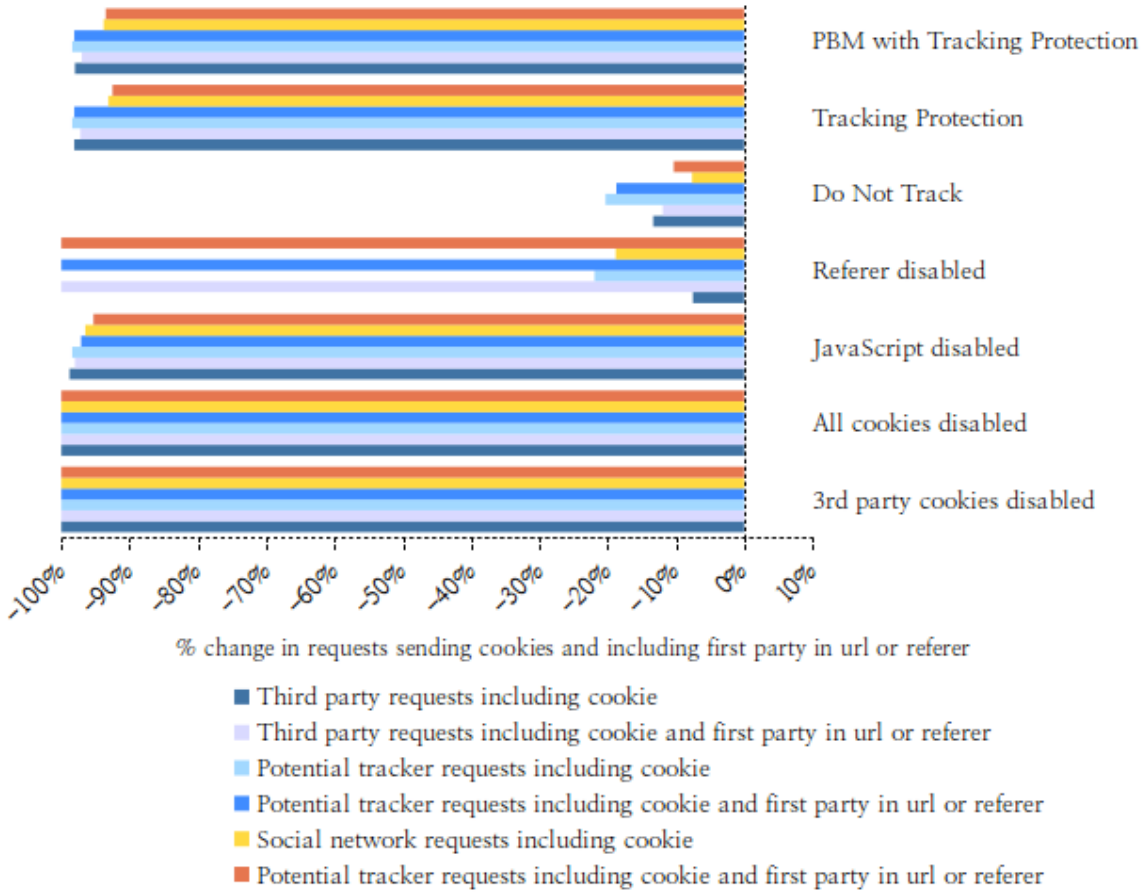


Figure 14.5: Effect of web browser configuration on HTTP requests sending cookies and first party details

14.3 Results of Tests Based on Browser Add-Ons

We will now turn to the results from the second category of tests, those based on web browser add-ons. Something to note when interpreting these results is that one add-on, NoScript Security Suite, is not designed for general tracking protection, rather it provides protection against execution of untrusted scripts and plug-ins, such as JavaScript and Adobe Flash.

14.3.1 Reduction in cookies

Figure 14.6 shows the impact the add-ons had on cookies. Disconnect, UBlock Origin, Ghostery and AdBlock Plus all produced similar results, reducing cookies across the board, with Ghostery configured in blocking mode reduced cookies overall by 79% and third party cookies 83%. Although the most effective of the four at reducing cookies, the drop in cookies of potential trackers and social networks was not up to par compared with the reduction in third party cookies, at 32% and 20% respectively. In our tests, AdBlock Plus blocked the least amount of cookies, reducing third party cookies by 60% but having no effect on cookies owned by social networks.

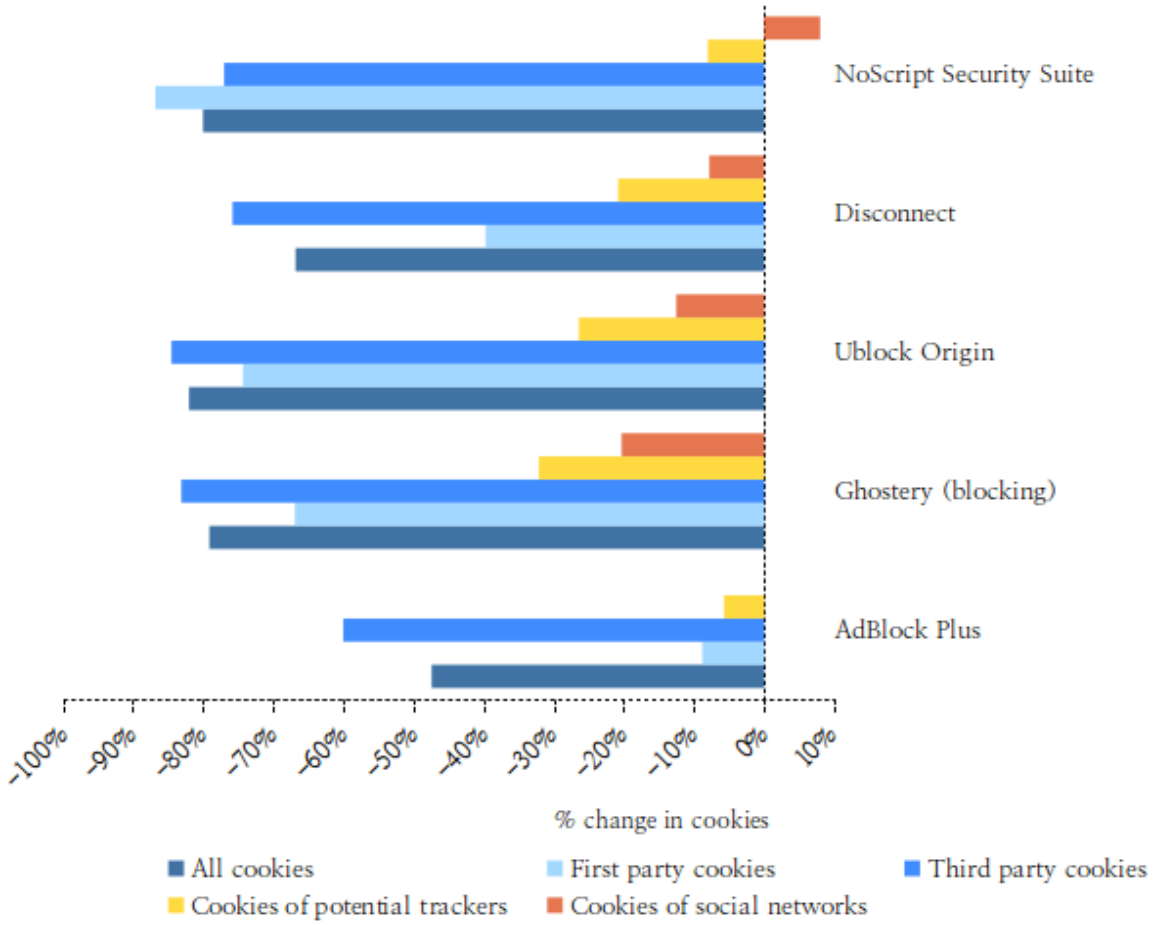


Figure 14.6: Effect of web browser add-ons on cookies

One concerning outcome is the 8% increase in social network cookies resulting from installing NoScript Security Suite. It's unclear why this occurred, and further testing and closer investigation is needed to get to the bottom of the increase.

14.3.2 Reduction in HTTP requests

Looking at the reduction in HTTP requests shows a similar picture to that of the reduction in cookies. As shown in Figure 14.7, all four list-based add-ons performed to similar level, although Ghostery once again performed particularly well with an 83% reduction in HTTP requests compared to AdBlock Plus with the least blocked requests at 65%. NoScript Security Suite reduced overall HTTP requests by the most with a fall of 93%, taking it close to the 94% fall in HTTP requests seen from disabling JavaScript. This is a good indication of the reliance on JavaScript to drive today's internet.

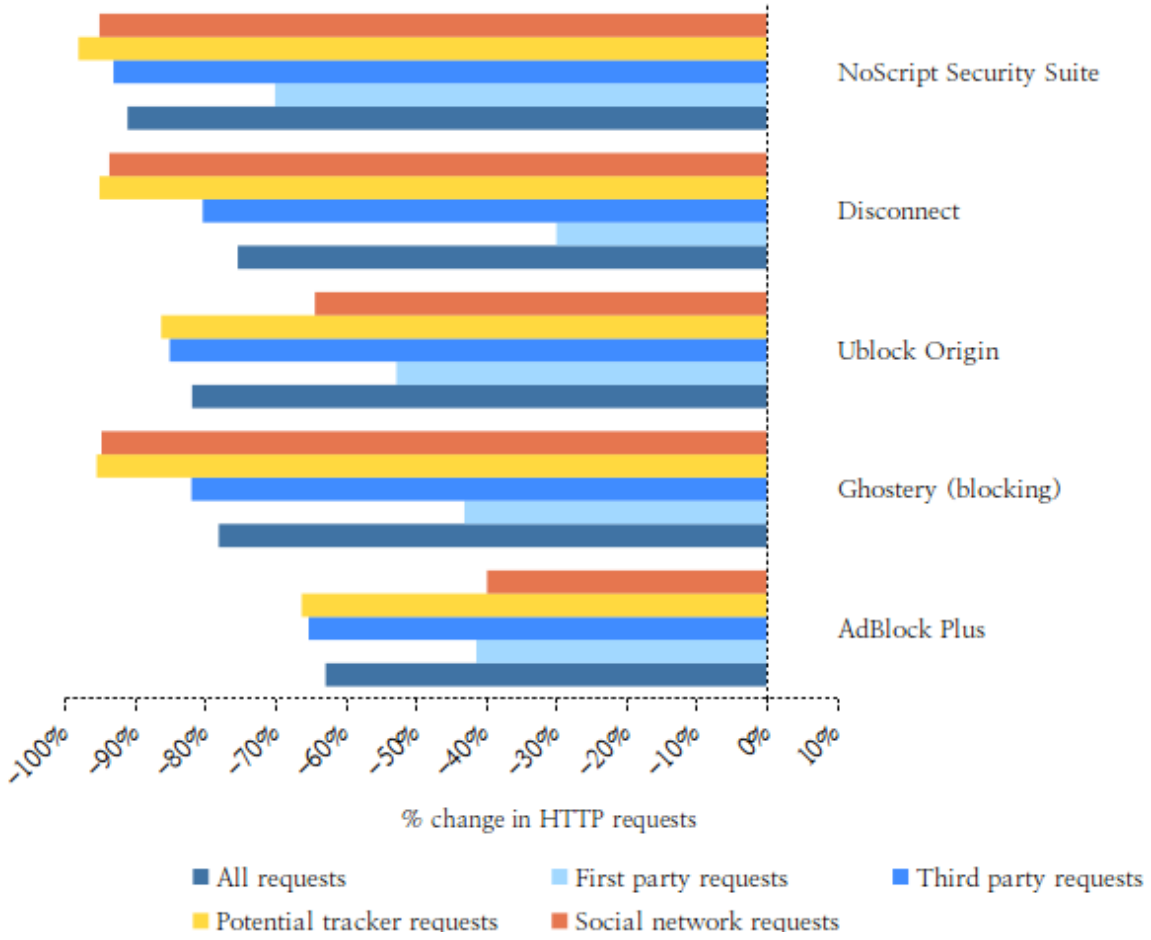


Figure 14.7: Effect of web browser add-ons on HTTP requests

14.3.3 Reduction in HTTP requests including cookies and first party

The reductions in HTTP requests potentially leaking information is very similar to those produced by the defences based on web browser configuration. Once again, a likely explanation for the drop is as a side effect of an overall reduction in HTTP requests, rather than specifically identifying and blocking requests capable of leaking information. Figure 14.8 shows the test results:

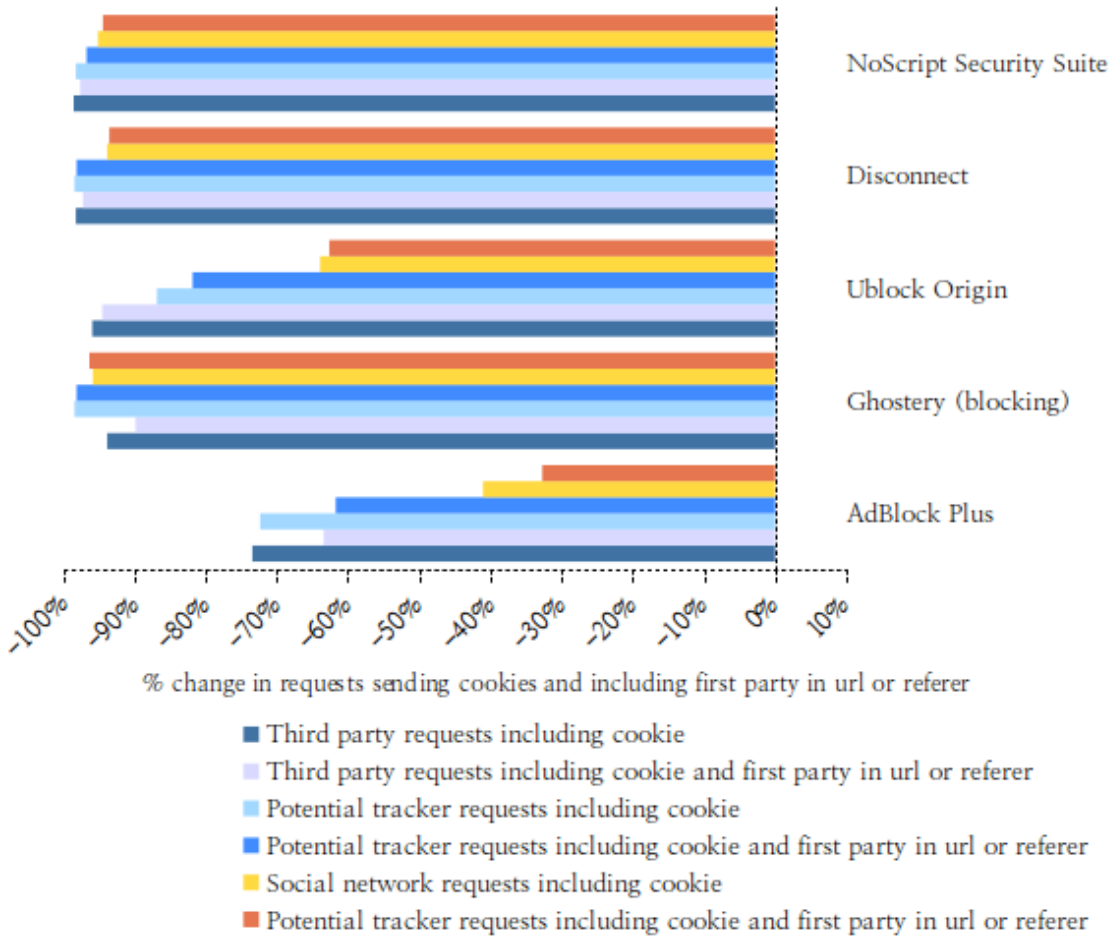


Figure 14.8: Effect of web browser add-ons on HTTP requests sending cookies and first party details

Once again, NoScript Security Suite and Ghostery reduced requests by a considerable amount, both reducing potential leakage to social networks by 95%. AdBlock Plus blocked the least amount of requests, resulting in a 40% reduction in potential information leakage to social networks.

14.4 Appraisal of Tested Defences

Both categories produced respectable reductions in cookies and HTTP requests, although none of the defences tested were completely effective in their own right. This isn't particularly surprising, as there remains a need to strike a balance between privacy and usability, therefore defences that consistently break webpages are unlikely to remain in place for very long. Unfortunately, it would appear the most effective defences are also the most destructive, for example, disabling cookies and JavaScript prevented log in to the social networks in our test

data set, which is no surprise.

With this in mind, many of the defences tested here proved effective at reducing both cookies and HTTP requests, and therefore reduced the amount of tracking taking place whilst also improving performance - potentially a win/win situation. Sadly, reducing tracking and preventing tracking are distinctly different requirements, and it would appear, preventing tracking is very difficult to achieve under normal circumstances.

14.4.1 Defences based on browser configuration

Of the browser configurations tested, Firefox's Tracking Protection proved effective at reducing tracking without adversely affecting the functionality of the webpage, reducing HTTP requests to third parties by 80%. One potential drawback is the lack of reduction in cookies, particularly the cookies of potential trackers and social networks. Using Private Browsing Mode with Tracking Protection, whilst not preventing cookies being set and exchanged within the browsing session, did prevent cookies persisting once the session closed which will help reduce tracking.

It was surprising to see some configuration changes result in more cookies being set, with Do Not Track (DNT) being an obvious candidate. An increase in cookies when using DNT may not necessarily be a negative: it's possible the additional cookies were set as a result of first and third parties respecting the request to not perform tracking by setting opt-out cookies in the user's browser, however, this would require further tests to verify.

Few of the configuration changes managed to reduce the cookies of potential trackers and social networks by any significant amount, with only disabling JavaScript and disabling all cookies having a positive outcome. Even disabling third party cookies did not see significant drops in the cookies from these two categories, which is slightly puzzling.

14.4.2 Defences based on browser add-ons

All the add-ons based on Tracking Protection Lists (TPLs) provided similar protection to varying degrees. Of these, Ghostery configured in blocking mode provided the most effective protection, reducing third party cookies by 83% and third party HTTP requests by 82%. It also proved the most efficient at reducing cookies from potential trackers and social networks, with a 32% and 20% reduction respectively. Disconnect produced results close to Firefox's built-in Tracking Protection, which is as expected as they share the same Tracking Protection Lists (TPLs) [94].

NoScript Security Suite proved the most effective at reducing third party HTTP requests, and was the only add-on to reduce third party, potential tracker and social networking requests

by over 90%. Based on blocking page elements rather than tracking domains, NoScript allows finer grained control over blocking JavaScript. This should help reduce the number of pages breaking as a result of blocking scripts whilst providing the advantages gained from disabling JavaScript completely, such as reducing or preventing Evercookies and fingerprinting.

14.5 Limitations

Although we feel our study has successfully evaluated the effectiveness of the defences tested, we acknowledge there are limitations to the tests. The evaluation criteria is quite narrow, examining cookies and HTTP requests only, and does not evaluate the impact of the defences against other forms of tracking such as browser fingerprinting. Perhaps a better approach would be to evaluate the defences against a cross-section of tracking techniques and score each defence in such a ways as to provide a quantitative measure of effectiveness. This would prove a better approach to identifying all round defences.

The timeframe over which the tests were conducted could also be viewed as a limitation, as can the limited size of the test data set. On reflection, a better approach may be to run a larger test data set multiple times and average the results to remove outliers. This would also give sufficient time to evaluate defences based on heuristics, such as PrivacyBadger.

15 Recommendations

We are now almost at the end of our study. So far, we have examined the methods and techniques used for tracking, defences against these methods, and also tested and appraised how well a selection of these defences perform. Here we will provide a number of recommendations to help reduce being tracked online, based on the research reviewed for this study as well as the tests we conducted.

To provide recommendations to as wide an audience as possible, the recommendations will be split into two: *Easily Implemented Recommendations* and *Technical Recommendations*. None of the recommendations presented here require a high level of technical expertise, and, with a little time, patience and experimentation, even the technical recommendations are within easy reach of the less technically inclined user.

15.1 Easily Implemented Recommendations

The recommendations in this section concentrate on selecting and configuring a good browser, and being mindful of your actions whilst online.

15.1.1 Select a good browser

Using a good browser can have a profound impact on the level of privacy experience whilst online. Before deciding on a particular browser, a number of variables should be examined. These variables will depend on the your requirements, such as the level of privacy required, how well supported the browser is by add-ons and extensions, and whether the browser offers complete anonymity or just a level of privacy. Our recommendation would be to select one of the main three browsers, either Mozilla Firefox, Microsoft Edge or Google Chrome. All these browsers are feature rich and and well supported, meaning security issues are patched quickly. One approach would be to install a selection of browsers and test them against each other so see which one best fits your needs.

As an alternative, many smaller companies offer enhanced privacy browsers based primarily

on the Chromium project[1] which forms the basis of Google Chrome, amongst others. If moving away from the main three browsers, if would be wise to perform some research before adopting one of the privacy-enhancing browsers to ensure it meets your needs.

15.1.2 Configure your browser appropriately

The results of the tests based on browser configuration, shown in the section *Results of Tests Based on Browser Configuration*, show how changes to web browser configuration can effect privacy. A defence recommended by Englehardt and Narayanan [17], understanding the configuration options available in your selected browser can improve your online privacy significantly. A good starting point would be to disable third party cookies [17], disable the referer header to reduce information leakage [3, 18, 14, 43], and enable tracking protection if your selected browser supports it [26]. For configuration liable to break webpages, such as disabling JavaScript or disabling cookies completely, the changes can be contained in different browser profiles to reduce the impacts.

15.1.3 Consider installing a privacy focused browser add-on

Our study reviewed five privacy focused browser add-ons and found them to provide an acceptable level of protection against cookies and HTTP requests to third parties. Section *Results of Tests Based on Browser Add-Ons* shows both Ghostery and Disconnect provide good privacy protection, so would be a reasonable starting point. As with web browsers, some add-ons may require configuration to work at their optimum level [29]. Ghostery is an example of this as it does not block third parties by default, requiring configuration to do so.

Ghostery and Disconnect also provide handy functionality to visualise the tracking on a particular page. This can be a helpful way of understanding what's happening on a page and also verifying how well defences are working. Figures 15.1 and 15.2 show screenshots of the page visualisation of both:

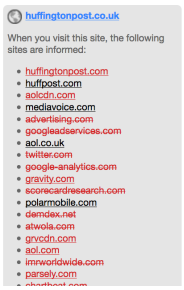
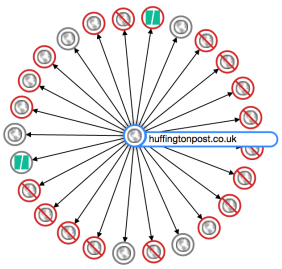


Figure 15.1: Disconnect's page visualisation



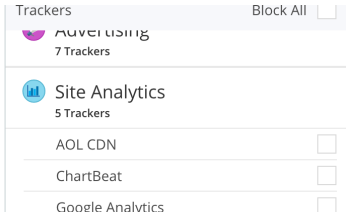


Figure 15.2: Ghostery's page visualisation

[1] `https://www.chromium.org/`

Unfortunately, as when installing any software from potentially untrustworthy sources, a word of warning is required. Browser add-ons have been used for malicious purposes, as seen in Michael Weissbacher's study *These Browser Extensions Spy on 8 Million Users* [96]. Care should to be taken when installing any software, and, for want of a better phrase, it can be beneficial to stick with the crowd when selecting browser add-ons and extensions.

15.1.4 Avoid reusing emails, social handles and telephone numbers for identification

As discussed earlier in *Replacements for advertiser unique ID*, reusing a constant identifier such as an email address or social handle encourages the aggregation of data across systems and profiles owned by different tracking companies, potentially resulting in a single giant behavioural profile containing all the browsing activity of the user. For this reason it is recommended to either log into accounts on a designated machine, or use different email address for different accounts [3].

15.1.5 Avoid volunteering information unnecessarily

Starov's, Gill's and Nikiforakis's study *Are you sure you want to contact us? Quantifying the leakage of PII via website contact forms* [8] shows how contact forms can be used to easily obtain and leak information. A moments thought before clicking the submit button on an online contact form can prevent leaking personal information. As seen in many studies reviewed here, once PII has been entered into a form, control over that data is potentially lost forever [1, 2, 49, 8], sometimes without even needing to click the submit button [8].

15.2 Technical Recommendations

The recommendations in this section are for the more technically adventurous, and should be seen as an extension rather than a replacement for the easily implemented recommendations.

15.2.1 Segregated browser profiles

Mozilla's Firefox provides the facility to create and manage multiple, independent profiles, each of which is isolated from the others. This means changes made to a profile's configuration are not reflected in the other profiles. This type of facility can be useful for providing specific contexts for different activities, such as for online banking and general web activity, or to test the effects of potentially breaking changes like disabling JavaScript. From a technical stand point and using Firefox as an example, profiles are managed using Firefox's Profile

Manager. To access the Profile Manager, start Firefox from the command line using the `-p` switch. This will show the Profile Manager interface, shown below in Figure 15.3:

Figure 15.3: Firefox's Profile Manager

16 Summary

This has been a hands-on part, involving the development of a Python framework to automate testing and the building of a virtual environment in which to run it. Finally, we tested the defences selected in the *Testing Defences* chapter. Of the defences tested, those based on web browser add-ons provided an acceptable level of privacy protection without impacting the functionality of the webpage. Disabling JavaScript and cookies demonstrated both the need for compromise between privacy and functionality, as both defences had detrimental effects on webpages, and also how tracking techniques are ingrained with the reliance on JavaScript and cookies, both of which can be seen to form the backbone of today's internet.

Few of the defences performed a really good job at reducing the cookies of potential trackers and social networks, with some actually increasing the cookie tally: enabling Do Not Track, disabling the referer header and the NoScript Security Suite add-on. Do Not Track (DNT) did not perform as well as hoped, with only a slight reduction in both cookies and HTTP requests, sadly leaving us to conclude it is insufficient a defence to be used on its own.

We presented a selection of recommendations based on the reviewed research and the outcome of the study, all of which provide a solid starting point for users to reduce tracking whilst online. The easily implemented recommendations outlined in this section are within reach of all users, regardless of their technical inclination, while the technical recommendations are a result of the testing carried out for this study and aim to augment rather than replace the easily implemented recommendations.

V Conclusions and Closing Remarks

17 Conclusions

The aim of this study was to answer one question:

Is it feasible to defend against being tracked whilst online?

To answer this question, we started by understanding who's behind web tracking and the reasons why users are tracked throughout their internet journeys, finding many parties involved and also many reasons for tracking. We showed how web tracking has a place in fighting crime by helping to identify fraud, how it can make life easier and more secure through active authentication, and how it allows personalised experiences to be delivered to shoppers through product recommendations. We also witnessed how this personalisation can result in price discrimination and steering in some cases, and showed how trackers go about creating extensive behavioural profiles on individuals by collecting vast amounts of data before augmenting these profiles with data from our offline lives, often including our real names, addresses and contact details. We also found how the data classifications of these profiles can easily be incorrect, and what effects this may have on us in the real world.

From here we moved to defences, reviewing the current crop offered by academia. We selected a number of defences for closer investigation based on the recommendation by Englehardt and Narayanan of selecting defences based on browser configuration and add-ons [17]. To evaluate these, tests were constructed and results interpreted with a view of presenting as quantitative an approach as possible within the constraints of the work. Although by no means highly-scientific, the tests provided an insight into the effectiveness of the defences and to what extent they reduce third party cookies and HTTP requests.

One critique of the study is the lack of testing around defending against device fingerprinting. Effort was made to identify fingerprinting taking place in the wide, however it was not to be and reliably identifying fingerprinting in action proved elusive. We have already seen the advantages of identifying users based on stateless technology like device fingerprinting in the *Why Track Users?* section and there is no denying it's effectiveness. This alone is quite scary, as defending against such an efficient technology will likely prove to be very troublesome, especially as one of the most effective defences at preventing fingerprinting, disabling

JavaScript, tends to have such a destructive impact to the functionality of a webpage. Fingerprinting could potentially make preventing tracking an impossible task for the average user.

The ingrained nature of tracking technology could also make complete prevention of tracking difficult. For example, tracking relies on exactly the same technologies and mechanisms as those for delivering web content, such as cookies, images, JavaScript, and third parties like Content Deliver Networks (CDNs). This blurs the lines as to how to prevent tracking, as simply blocking content and HTTP requests is not a practical solution. It doesn't take the adaptability of the trackers into account either. The adaptability of trackers has been seen in the past with many now using first party cookies in preference to third party cookies, possibly due to more users blocking third party cookies as awareness of their use increases. As most of the add-ons tested in this study prevent tracking by blocking HTTP requests to domains appearing on Tracking Protection Lists (TPLs), the obvious response would be to move tracking content such as scripts and images to allowable third parties such as CDNs. This would be a safe bet from the trackers' standpoint as blocking the CDNs would prevent webpage content being delivered as well.

Turning our attention back to the original question, this study would suggest it is practical to reduce but not completely prevent web tracking. This should be caveated with the phrase *for the user of average technical ability* as the study did not delve deeply into particularly complex solutions such as using VPNs and TORs to reduce or prevent tracking.

18 Future Work

The scope for future work to extend the study is quite large, with a number of areas and avenues emerging as worthy of closer investigation. Work centred around identifying and preventing device fingerprinting would be beneficial, as would longer term testing of browser add-ons, particularly to compare heuristic-based against list-based add-ons. Comparing the effectiveness of different Tracking Protection Lists (TPLs), such as testing Disconnect's strict TPLs with Firefox's Tracking Protection, would be worthy of future investigation, as work by Mayer and Mitchell found TPL selection to alter the effectiveness of list-based add-ons [101]. However, we feel more benefit would result from examining two other areas related to tracking.

18.0.1 Develop a privacy focused browser extension

Rather than conduct another study on tracking we would be inclined to develop a browser extension to act as a test bed for further investigation. This approach would allow both manipulation and observation of the webpage in real time, rather than needing to wait for the page to load before quantifying results such as how many cookies were set. This has the advantage of being able to monitor the page as it is being constructed, potentially allowing manipulations of the page's DOM to be identified.

18.0.2 Tracking on mobile platforms

We would also like to investigate the tracking being conducted on smartphones via mobile browsers and apps. Apps on smartphones provide another avenue for tracking users and is worthy of a study in it's own right. Mobile web browsers are generally less functional versions of their desktop cousins and tend not to offer the same support for privacy protection via browser add-ons. Once again, we would be more inclined to develop a smartphone app or similar piece of functionality to form the basis of study.

19 Closing Remarks

It's no surprise to arrive at the end of this work with a list of questions longer than I started with. Seemingly, each question answered resulted in two, three or four new questions being posed. While I feel I take a lot away from completing the study, I also feel the journey has only just started, and the actual size and complexity of the topic is only now becoming apparent.

Intentionally Blank

Bibliography

[1] B. Krishnamurthy and C. E. Wills, "Generating a privacy footprint on the internet," in *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '06. New York, NY, USA: ACM, Oct. 2006, pp. 65–70, accessed 09/06/2017. [Online]. Available: http://doi.acm.org/10.1145/1177080.1177088

[2] ——, "On the leakage of personally identifiable information via online social networks," in *Proceedings of the 2nd ACM workshop on Online social networks*. ACM, aug 2009, pp. 7–12, accessed 01/06/2017. [Online]. Available: http://conferences.sigcomm.org/sigcomm/2009/workshops/wosn/papers/p7.pdf

[3] B. Krishnamurthy, K. Naryshkin, and C. E. Wills, "Privacy leakage vs. protection measures: the growing disconnect," in *Proceedings of the Web*, vol. 2, May 2011, pp. 1–10, accessed 09/06/2017.

[4] A. Korolova, "Privacy violations using microtargeted ads: A case study," in *2010 IEEE International Conference on Data Mining Workshops*, Dec. 2010, pp. 474–482, accessed 17/02/2017.

[5] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas. (2010) Adnostic: Privacy preserving targeted advertising. Accessed 09/06/2017.

[6] M. Bilenko, M. Richardson, and J. Tsai. (2011, Sep.) Targeted, not tracked: Client-side solutions for privacy-friendly behavioral advertising. Accessed 09/06/2017. [Online]. Available: https://papers.ssrn.com/abstract=1995127

[7] J. Parra-Arnau, "Pay-per-tracking: A collaborative masking model for web browsing," *Information Sciences*, vol. 385?386, pp. 96–124, Apr. 2017, accessed 09/06/2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025516322587

[8] O. Starov, P. Gill, and N. Nikiforakis, "Are you sure you want to contact us? quantifying the leakage of pii via website contact forms," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 1, pp. 20–33, Jan. 2016, accessed

08/06/2017. [Online]. Available: https://www.degruyter.com/view/j/popets.2016.2016.issue-1/popets-2015-0028/popets-2015-0028.xml

[9] S. Englehardt, D. Reisman, C. Eubank, P. Zimmerman, J. Mayer, A. Narayanan, and E. W. Felten, "Cookies that give you away: The surveillance implications of web tracking," in *Proceedings of the 24th International Conference on World Wide Web*. ACM, 2015, pp. 289–299, accessed 06/06/2017.

[10] T. Bujlow, V. Carela-Español, J. Solé-Pareta, and P. Barlet-Ros, "Web tracking: Mechanisms, implications, and defenses," *arXiv preprint arXiv:1507.07872*, 2015, accessed 05/06/2017.

[11] T. Libert, *Exposing the hidden web: An analysis of third-party http requests on 1 million websites*, arXiv, 2015, accessed 06/06/2017. [Online]. Available: https://timlibert.me/pdf/Libert-2015-Exposing_Hidden_Web_on_Million_Sites.pdf

[12] A. Lerner, A. K. Simpson, T. Kohno, and F. Roesner, "Internet jones and the raiders of the lost trackers: An archaeological study of web tracking from 1996 to 2016," in *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Aug. 2016, accessed 16/02/2017. [Online]. Available: https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_lerner.pdf

[13] S. Schelter and J. Kunegis, "On the ubiquity of web tracking: Insights from a billion-page web crawl," *CoRR*, vol. abs/1607.07403, Jul. 2016, accessed 09/06/2017. [Online]. Available: http://arxiv.org/abs/1607.07403

[14] F. Roesner, T. Kohno, and D. Wetherall, "Detecting and defending against third-party tracking on the web," in *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, Apr. 2012, pp. 12–12, accessed 16/02/2017. [Online]. Available: https://www.usenix.org/system/files/conference/nsdi12/nsdi12-final17.pdf

[15] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "Cookieless monster: Exploring the ecosystem of web-based device fingerprinting," in *Security and privacy (SP), 2013 IEEE symposium on*. IEEE, 2013, pp. 541–555, accessed 08/06/2017. [Online]. Available: https://lirias.kuleuven.be/bitstream/123456789/393661/1/cookieless_sp2013.pdf

[16] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, "The web never forgets: Persistent tracking mechanisms in the wild," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '14. New York, NY, USA: ACM, Nov. 2014, pp. 674–689, accessed 09/06/2017. [Online]. Available: http://doi.acm.org/10.1145/2660267.2660347

[17] S. Englehardt and A. Narayanan, "Online tracking: A 1-million-site measurement and analysis," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, Oct. 2016, pp. 1388–1401, accessed 16/02/2017.

[18] I. Sanchez-Rola, X. Ugarte-Pedrero, I. Santos, and P. G. Bringas, "The web is watching you: A comprehensive review of web-tracking techniques and countermeasures," *Logic Journal of the IGPL*, vol. 25, no. 1, p. 18, Feb. 2017, accessed 09/06/2017. [Online]. Available: http://dx.doi.org/10.1093/jigpal/jzw041

[19] K. Mowery and H. Shacham, "Pixel perfect: Fingerprinting canvas in html5," *Proceedings of W2SP*, pp. 1–12, May 2012, accessed 09/06/2017. [Online]. Available: https://cseweb.ucsd.edu/~hovav/dist/canvas.pdf

[20] C. Diaz, L. Olejnik, G. Acar, and C. Casteluccia, "The leaking battery: A privacy analysis of the html5 battery status api," in *Lecture Notes in Computer Science*, vol. 9481. Springer, Jan. 2015, pp. 254–263, accessed 18/02/2017. [Online]. Available: https://eprint.iacr.org/2015/616.pdf

[21] W. Meng, B. Lee, X. Xing, and W. Lee, "Trackmeornot: Enabling flexible control on web tracking," in *Proceedings of the 25th International Conference on World Wide Web*, ser. WWW '16. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, Apr. 2016, pp. 99–109, accessed 16/02/2017. [Online]. Available: https://doi.org/10.1145/2872427.2883034

[22] P. Laperdrix, W. Rudametkin, and B. Baudry, "Mitigating browser fingerprint tracking: Multi-level reconfiguration and diversification," in *Proceedings of the 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, ser. SEAMS '15. Piscataway, NJ, USA: IEEE Press, May 2015, pp. 98–108, accessed 09/06/2017. [Online]. Available: http://dl.acm.org/citation.cfm?id=2821357.2821378

[23] S. Luangmaneerote, E. Zaluska, and L. Carr, "Survey of existing fingerprint countermeasures," Sep. 2016, accessed 09/06/2017. [Online]. Available: https://eprints.soton.ac.uk/401902/

[24] N. Bielova, F. Besson, and T. Jensen, "Using javascript monitoring to prevent device fingerprinting," *ERCIM News*, 2016, accessed 09/06/2017. [Online]. Available: https://hal.inria.fr/hal-01353997/file/final.pdf

[25] G. Aggarwal, E. Bursztein, C. Jackson, and D. Boneh, "An analysis of private browsing modes in modern browsers." in *USENIX Security Symposium*, Aug. 2010, pp. 79–94, accessed 10/06/2017. [Online]. Available: https://cdn.elie.net/publications/an-analysis-of-private-browsing-modes-in-modern-browsers-slides.pdf

[26] G. Kontaxis and M. Chew, "Tracking protection in firefox for privacy and performance," *CoRR*, vol. abs/1506.04104, Jun. 2015, accessed 09/06/2017. [Online]. Available: http://arxiv.org/abs/1506.04104

[27] X. Pan, Y. Cao, and Y. Chen, "I do not know what you visited last summer: Protecting users from third-party web tracking with trackingfree browser," in *Proceedings of the 2015 Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA*, Feb. 2015, accessed 09/06/2017. [Online]. Available: http://www.yinzhicao.org/TrackingFree/trackingfree_NDSS15.pdf

[28] N. Tsalis, A. Mylonas, and D. Gritzalis, *An Intensive Analysis of Security and Privacy Browser Add-Ons*. Cham: Springer International Publishing, Apr. 2016, pp. 258–273, accessed 09/06/2017. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-31811-0_16

[29] A. Gervais, A. Filios, V. Lenders, and S. Capkun, "Quantifying web adblocker privacy," Cryptology ePrint Archive, Report 2016/900, Sep. 2016, accessed 09/06/2017. [Online]. Available: http://eprint.iacr.org/2016/900

[30] N. Tsalis, A. Mylonas, A. Nisioti, D. Gritzalis, and V. Katos, "Exploring the protection of private browsing in desktop browsers," *Computers & Security*, vol. 67, pp. 181–197, Mar. 2017, accessed 10/06/2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404817300597

[31] (2017, Apr.) Full year 2016 digital adspend results — iab uk. Accessed 11/07/2017. [Online]. Available: https://iabuk.net/research/library/full-year-2016-digital-adspend-results

[32] T. Hobbs. (2017, Apr.) Uk ad spend hit 21.4bn in 2016 as digital continues to dominate. Accessed 11/07/2017. [Online]. Available: https://www.marketingweek.com/2017/04/25/uk-ad-spend-digital/

[33] C. Castelluccia, M.-A. Kaafar, and M.-D. Tran, *Betrayed by Your Ads!* Springer Berlin Heidelberg, 2012, accessed 09/06/2017. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31680-7_1

[34] (2014, Sep.) Getting to know you. The Economist. Accessed 16/02/2017. [Online]. Available: http://www.economist.com/news/special-report/21615871-everything-people-do-online-avidly-followed-advertisers-and-third-party

[35] R. Siciliano. (2014, Apr.) Data brokers: What are they; how to get control of your name. Accessed 17/02/2017. [Online]. Available: http://www.huffingtonpost.com/robert-siciliano/data-brokers-what-are-the_b_5185127.html

[36] G. Anthes. (2015, Jan.) Data brokers are watching you. Accessed 17/02/2017. [Online]. Available: https://katie.mtech.edu/classes/csci340/Resources/DataBrokersAreWatchingYou.pdf

[37] J. Angwin, T. P. Jr, and S. Mattu. (2016, Dec.) Facebook buys information from data brokers about users' offline lives - business insider. Accessed 17/02/2017. [Online]. Available: http://uk.businessinsider.com/facebook-data-brokers-2016-12?r=US&IR=T

[38] M. Reilly. (2016, Dec.) Facebook knows a lot about your offline life that it isn?t telling you. Accessed 17/02/2017. [Online]. Available: https://www.technologyreview.com/s/603283/how-facebook-learns-about-your-offline-life/

[39] A. T. UK. Cookie policy - auto trader uk. Accessed 17/02/2017. [Online]. Available: http://www.autotrader.co.uk/cookie-policy

[40] J. Carter. (2015, Sep.) Behavioural biometrics ? the future of security. Accessed 11/07/2017. [Online]. Available: http://www.techradar.com/news/world-of-tech/future-tech/behavioural-biometrics-the-future-of-security-1302888

[41] Reuters. (2017, Apr.) Experian enlists behavioral biometrics startup to combat online fraud. Accessed 11/07/2017. [Online]. Available: http://fortune.com/2017/04/07/experian-biometrics-startup-fraud-online/

[42] Darpa: Active authentication program. BehavioSec. Accessed 14/06/2017. [Online]. Available: https://www.behaviosec.com/darpa-active-authentication-program/

[43] T. Bujlow, V. Carela-Espaol, J. Sol-Pareta, and P. Barlet-Ros, "A survey on web tracking: Mechanisms, implications, and defenses," *Proceedings of the IEEE*, vol. PP, no. 99, pp. 1–35, Mar. 2017, accessed 09/06/2017. [Online]. Available: http://ieeexplore.ieee.org/document/7872467

[44] I. Murphy. (2017, Feb.) Malvertising grows faster than advertising. Accessed 08/06/2017. [Online]. Available: https://www.enterprisetimes.co.uk/2017/02/01/malvertising-grows-faster-advertising/

[45] J. Segura. (2016, Mar.) Large angler malvertising campaign hits top publishers. Accessed 08/06/2017. [Online]. Available: https://blog.malwarebytes.com/threat-analysis/2016/03/large-angler-malvertising-campaign-hits-top-publishers/

[46] J. Wrolstad and B. Vengerik, "Pinpointing targets: Exploiting web analytics to ensnare victims," FireEye Threat Intelligence, Nov. 2015, accessed 05/06/2017. [Online]. Available: https://www2.fireeye.com/rs/848-DID-242/images/rpt-witchcoven.pdf

[47] "The waterbug attack group," Symantec, Jan. 2016, accessed 05/06/2017. [Online]. Available: https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/waterbug-attack-group.pdf

[48] A. Davies. (2012, Aug.) Three major benefits of using personalisation in marketing. Accessed 07/06/2017. [Online]. Available: http://econsultancy.com/blog/63212-three-major-benefits-of-using-personalisation-in-marketing/?utm_campaign=bloglikes&utm_medium=socialnetwork&utm_source=facebook

[49] L. Agarwal, N. Shrivastava, S. Jaiswal, and S. Panjwani, "Do not embarrass: Re-examining user concerns for online tracking and advertising," in *Proceedings of the Ninth Symposium on Usable Privacy and Security*, ser. SOUPS '13. New York, NY, USA: ACM, 2013, pp. 8:1–8:13, accessed 07/06/2017. [Online]. Available: http://doi.acm.org/10.1145/2501604.2501612

[50] Price discrimination as a profit maximising strategy. Accessed 06/06/2017. [Online]. Available: http://www.economicsonline.co.uk/Business_economics/Price_discrimination.html

[51] N. Zuo and M. Wending. (2017, Jun.) Election 2017: How can you pop your filter bubble? BBC. Accessed 07/06/2017. [Online]. Available: http://www.bbc.co.uk/news/av/magazine-40137211/election-2017-how-can-you-pop-your-filter-bubble

[52] E. Bozdag, "Bias in algorithmic filtering and personalization," *Ethics and Information Technology*, vol. 15, no. 3, pp. 209–227, 2013, accessed 07/06/2017. [Online]. Available: http://dx.doi.org/10.1007/s10676-013-9321-6

[53] A. J. van Deursen and J. A. van Dijk, "Using the internet: Skill related problems in users? online behavior," *Interacting with Computers*, vol. 21, no. 5-6, p. 393, 2009, accessed 07/06/2017. [Online]. Available: http://dx.doi.org/10.1016/j.intcom.2009.06.005

[54] (2011) Personal data: The emergence of a new asset class. World Economic Forum. Accessed 08/06/2017. [Online]. Available: http://www3.weforum.org/docs/WEF_ITTC_PersonalDataNewAsset_Report_2011.pdf

[55] R. Mason. (2014, Apr.) Treasury must urgently explain plans to sell taxpayers' details, says labour. The Guardian. Accessed 08/06/2017. [Online]. Available: https://www.theguardian.com/politics/2014/apr/20/treasury-plans-sell-taxpayers-details-labour

[56] *Privacy Policy*, Catch Oz, accessed 17/01/2017. [Online]. Available: https://catchoz.com/privacy/

[57] C. Cuomo, J. Shaylor, M. McGuirt, and C. Francescani. (2011, Jan.) Carded: Customers spied on then penalized. ABC News. Accessed 08/06/2017. [Online]. Available: http://abcnews.go.com/GMA/TheLaw/gma-answers-credit-card-companies-financially-profiling-customers/story?id=6747461

[58] M. Zawadzi?ski. (2015, Jul.) What is cookie syncing and how does it work? - clearcode blog. Clearcode. Accessed 13/06/2017. [Online]. Available: http://clearcode.cc/2015/12/cookie-syncing/

[59] Set-cookie. Mozilla Developer Network. Accessed 13/06/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie

[60] Disable third-party cookies in firefox to stop some types of tracking by advertisers — firefox help. Mozilla Developer Network. Accessed 13/06/2017. [Online]. Available: https://support.mozilla.org/en-US/kb/disable-third-party-cookies

[61] Same-origin policy. Mozilla Developer Network. Accessed 13/06/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy

[62] Httponly - owasp. OWASP. Accessed 13/06/2017. [Online]. Available: https://www.owasp.org/index.php/HttpOnly#What_is_HttpOnly.3F

[63] What is a cname record? - dnsimple help. dnsimple.com. Accessed 13/06/2017. [Online]. Available: https://support.dnsimple.com/articles/cname-record/

[64] Help centre. Facebook. Accessed 14/06/2017. [Online]. Available: https://en-gb.facebook.com/help/112146705538576?helpref=faq_content

[65] M. D. Ayenson, D. J. Wambach, A. Soltani, N. Good, and C. J. Hoofnagle, *Flash cookies and privacy II: Now with HTML5 and ETag respawning*, Jul. 2011, accessed 14/06/2017. [Online]. Available: https://papers.ssrn.com/abstract=1898390

[66] A. M. McDonald and L. F. Cranor, "A survey of the use of adobe flash local shared objects to respawn http cookies," *ISJLP*, vol. 7, p. 639, Jan. 2011, accessed 14/06/2017. [Online]. Available: http://heinonline.org/HOL/Page?handle=hein.journals/isjlpsoc7&id=651&div=&collection=

[67] S. Kamkar. (2010, Oct.) evercookie. Accessed 14/06/2017. [Online]. Available: https://samy.pl/evercookie

[68] ——. evercookie. Accessed 14/06/2017. [Online]. Available: https://github.com/samyk/evercookie

[69] Using indexeddb. Mozilla Developer Network. Accessed 16/06/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB_API/Using_IndexedDB

[70] Referer. Mozilla Developer Network. Accessed 14/06/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referer

[71] Forbidden header name. Accessed 11/08/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Glossary/Forbidden_header_name

[72] J. Reschke and R. Fielding. (2013, Jul.) Hypertext transfer protocol (http/1.1): Semantics and content. Accessed 14/06/2017. [Online]. Available: https://tools.ietf.org/html/draft-ietf-httpbis-p2-semantics-23#section-5.5.3

[73] (2016) Adding analytics.js to your site — analytics for web (analytics.js). Google Developers. Accessed 16/06/2017. [Online]. Available: https://developers.google.com/analytics/devguides/collection/analyticsjs/

[74] Window.name. Mozilla Developer Network. Accessed 16/06/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Window/name

[75] ¡iframe¿. Mozilla Developer Network. Accessed 16/06/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe

[76] P. Eckersley, *How Unique Is Your Web Browser?* Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 1–18, accessed 18/06/2017. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-14527-8_1

[77] M. Mulazzani, P. Reschl, M. Huber, M. Leithner, S. Schrittwieser, E. Weippl, and F. C. Wien, "Fast and reliable browser identification with javascript engine fingerprinting," in *Web 2.0 Workshop on Security and Privacy (W2SP)*, vol. 5, May 2013, accessed 09/06/2017. [Online]. Available: https://ai2-s2-pdfs.s3.amazonaws.com/bdb4/30d06f3e134b2df8965dd62c77ac87710064.pdf

[78] Navigatorplugins.plugins. Mozilla Developer Network. Accessed 18/06/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/NavigatorPlugins/plugins

[79] O. Starov and N. Nikiforakis, *XHOUND: Quantifying the Fingerprintability of Browser Extensions*, accessed 18/06/2017. [Online]. Available: https://www.securitee.org/files/xhound-oakland17.pdf

[80] N. M. Al-Fannah and W. Li, "Not all browsers are created equal: Comparing web browser fingerprintability," *arXiv preprint arXiv:1703.05066*, Mar. 2017, accessed 18/06/2017. [Online]. Available: http://arxiv.org/abs/1703.05066

[81] P. Laperdrix, W. Rudametkin, and B. Baudry, "Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints," in *2016*

IEEE Symposium on Security and Privacy (SP), May 2016, pp. 878–894, accessed 18/06/2017. [Online]. Available: http://ieeexplore.ieee.org/document/7546540/?reload=true#full-text-section

[82] Canvas api. Mozilla Developer Network. Accessed 18/06/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API

[83] F. Alaca and P. C. van Oorschot, "Device fingerprinting for augmenting web authentication: Classification and analysis of methods," in *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, ser. ACSAC '16. New York, NY, USA: ACM, Dec. 2016, pp. 289–301, accessed 18/06/2017. [Online]. Available: http://doi.acm.org/10.1145/2991079.2991091

[84] S. Englehardt and A. Narayanan. (2016, Oct.) Web privacy through transparency. Accessed 09/06/2017. [Online]. Available: http://senglehardt.com/presentations/2016_10_ccs_online_tracking.pdf

[85] Battery status api. Mozilla Developer Network. Accessed 18/06/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Battery_Status_API

[86] M. Husák, M. Čermák, T. Jirsík, and P. Čeleda, "Https traffic analysis and client identification using passive ssl/tls fingerprinting," *EURASIP Journal on Information Security*, vol. 2016, no. 1, p. 6, Feb. 2016, accessed 05/06/2017. [Online]. Available: http://dx.doi.org/10.1186/s13635-016-0030-7

[87] Webrtc leak test - ip address discovery - media device id fingerprint - browserleaks.com. Accessed 18/06/2017. [Online]. Available: https://browserleaks.com/webrtc#webrtc-device-id

[88] (2015, Jul.) How to disable webrtc in firefox and chrome. Accessed 05/07/2017. [Online]. Available: https://www.purevpn.com/blog/disable-webrtc-in-chrome-and-firefox/

[89] Continuous authentication. BehavioSec. Accessed 14/06/2017. [Online]. Available: https://www.behaviosec.com/

[90] D. Goodin. (2015, Jul.) How the way you type can shatter anonymity?even on tor. Accessed 14/06/2017. [Online]. Available: https://arstechnica.co.uk/security/2015/07/how-the-way-you-type-can-shatter-anonymity-even-on-tor/

[91] Tracking protection. Mozilla Developer Network. Accessed 04/07/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Mozilla/Firefox/Privacy/Tracking_Protection

[92] Market share for mobile, browsers, operating systems and search engines — netmarketshare. NetMarketShare. Accessed 04/07/2017. [Online]. Available: https://www.netmarketshare.com/

[93] Contributing to the mozilla code base. Mozilla Developer Network. Accessed 04/07/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Mozilla/Developer_guide/Introduction

[94] Tracking protection in private browsing — firefox help. Accessed 04/08/2017. [Online]. Available: https://support.mozilla.org/en-US/kb/tracking-protection-pbm

[95] N. Kaur, S. Azam, K. Kannoorpatti, K. C. Yeo, and B. Shanmugam, "Browser fingerprinting as user tracking technology," in *2017 11th International Conference on Intelligent Systems and Control (ISCO)*, Jan. 2017, pp. 103–111, accessed 09/06/2017. [Online]. Available: http://ieeexplore.ieee.org/document/7855963/

[96] M. Weissbacher. (2016) These browser extensions spy on 8 million users. Accessed 09/06/2017. [Online]. Available: https://www.ftc.gov/system/files/documents/public_comments/2016/09/00017-128978.pdf

[97] Cross-site request forgery (csrf) prevention cheat sheet - owasp. Accessed 11/08/2017. [Online]. Available: https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet#Checking_the_Referer_Header

[98] Tor project: Overview. Accessed 05/07/2017. [Online]. Available: https://www.torproject.org/about/overview.html.en

[99] Tor project: Anonymity online. Accessed 05/07/2017. [Online]. Available: https://www.torproject.org/

[100] K. Poulsen. (2014, Dec.) The fbi used the web's favorite hacking tool to unmask tor users. Accessed 05/07/2017. [Online]. Available: https://www.wired.com/2014/12/fbi-metasploit-tor/

[101] J. R. Mayer and J. C. Mitchell, "Third-party web tracking: Policy and technology," in *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 2012, pp. 413–427, accessed 26/07/2017. [Online]. Available: https://jonathanmayer.org/papers_data/trackingsurvey12.pdf

[102] Marionette. Mozilla Developer Network. Accessed 04/08/2017. [Online]. Available: https://developer.mozilla.org/en-US/docs/Mozilla/QA/Marionette

[103] Marionette python client ? marionette python client documentation. Accessed 04/08/2017. [Online]. Available: https://marionette-client.readthedocs.io/en/latest/index.html

[104] S. Schelter and J. Kunegis, "Tracking the trackers: A large-scale analysis of embedded web trackers." in *ICWSM*, Mar. 2016, pp. 679–682, accessed 09/06/2017. [Online]. Available: https://www.researchgate.net/profile/Sebastian_Schelter/publication/301222141_Tracking_the_Trackers_A_Large-Scale_Analysis_of_Embedded_Web_Trackers/links/570d504008aed31341cf76cd.pdf

[105] G. Merzdovnik, M. Huber, D. Buhov, N. Nikiforakis, S. Neuner, M. Schmiedecker, and E. Weippl, "Block me if you can: A large-scale study of tracker-blocking tools," in *Proceedings of the 2nd IEEE European Symposium on Security and Privacy (IEEE EuroS&P)*, 2017, accessed 24/06/2017. [Online]. Available: https://www.securitee.org/files/trackblock_eurosp2017.pdf

Appendices

A An Example of Price Discrimination

Figure 1: Example effects of browsing using PBM

B An Example of Fingerprinting Attributes

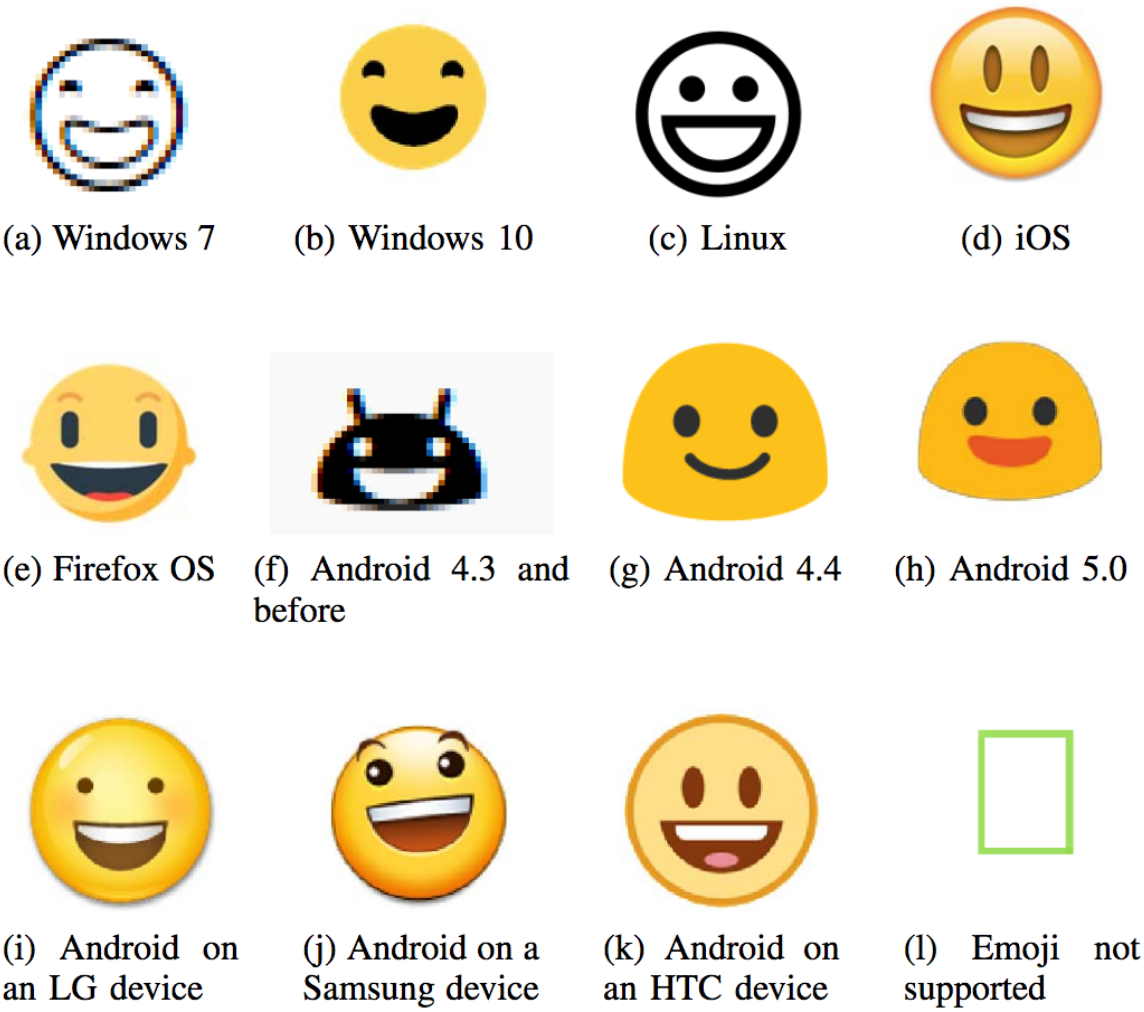


Figure 2: Example of how emojis can be used to aid fingerprinting

(Source: *Beauty and the Beast: Diverting Modern Web Browsers to Build Unique Browser Fingerprints* [81])

C URLs of News Articles Selected for Testing

One notable member is missing from the list: Google News. At the time of writing, Google News appears at number five in the Alexa ranking. However, as we will see shortly, Google domains also appear on known tracker lists. This made correlating results troublesome as Google domains appeared as both first party and third party sites, so Google news was substituted by the eleventh placed Huffington Post.

URLs of News Articles Selected for Testing
http://www.indiatimes.com/news/world/pakistan-supreme-court-disqualifies-nawaz-sharif-as-prime-minister-in-panama-papers-case-326736.html
https://www.nytimes.com/2017/07/27/us/politics/senate-russia-sanctions-trump.html?hp&action=click&pgtype=Homepage&clickSource=story-heading&module=first-column-region&region=top-news&WT.nav=top-news
https://www.reddit.com/r/funny/comments/6puha0/more_than_two_genders/
http://edition.cnn.com/2017/07/26/europe/russia-us-sanctions/index.html
https://www.theguardian.com/politics/2017/jul/28/philip-hammond-confirms-uk-will-seek-brexit-transitional-deal
http://www.huffingtonpost.co.uk/entry/brexit-things-lost_uk_597f086ae4b0da64e87a52ea?utm_hp_ref=uk
https://www.washingtonpost.com/news/innovations/wp/2017/07/29/heres-what-we-know-about-teslas-model-3/?utm_term=.7505b26e154c
https://www.forbes.com/sites/kenrapoza/2017/07/25/europe-fears-russia-sanctions-upgrade-but-will-go-along-with-washington-anyway/#64d77b011133
https://www.yahoo.com/news/senate-skinny-repeal-obamacare-falls-apart-senate-floor-mccain-defects-072042708.html
http://www.bbc.co.uk/news/uk-wales-mid-wales-40764862

Table 1: URLs of news articles selected for testing

D Potential Trackers and Social Networks Included in the Test Data Set

Potential Trackers and Social Networks Included in the Test Data Set	
Potential Trackers	*Social Networks*
http://www.google-analytics.com	https://twitter.com
http://www.gstatic.com	https://plus.google.com
http://www.doubleclick.net	https://pinterest.com
https://www.doubleclickbygoogle.com	https://pinterest.co.uk
https://www.google.com	https://linkedin.com
https://fonts.googleapis.com	https://facebook.com
https://www.facebook.com	
http://www.facebook.net	
https://ajax.googleapis.com	
http://www.fbcdn.net	
https://www.twitter.com	
http://www.googleadservices.com	
https://www.adnxs.com	
http://www.googleusercontent.com	
http://www.bluekai.com	
http://www.mathtag.com	
http://www.mediamath.com	
https://www.youtube.com/	
https://www.ytimg.com	
http://www.googletagmanager.com	
https://www.yahoo.com	

Table 2: Potential Trackers and Social Networks included in the test data set

E Firefox Profile Configuration Script

```
# Mozilla User Preferences

user_pref("browser.startup.homepage", "");
user_pref("browser.startup.page", 0);
user_pref("browser.cache.disk.enable", false);
user_pref("browser.cache.memory.enable", false);
user_pref("browser.safebrowsing.malware.enabled", false);
user_pref("browser.safebrowsing.phishing.enabled", false);
user_pref("browser.privatebrowsing.autostart", false);
user_pref("network.prefetch-next", false);
user_pref("network.dns.disablePrefetchHTTPS", true);
user_pref("network.dns.disablePrefetch", true);
user_pref("network.dnsCacheEntries", 0);
user_pref("network.cookie.cookieBehavior", 0);
user_pref("network.http.sendRefererHeader", 2);
user_pref("privacy.trackingprotection.enabled", false);
user_pref("privacy.trackingprotection.pbmode.enabled", false);
user_pref("privacy.donottrackheader.enabled", false);
user_pref("plugins.flashBlock.enabled", false);
user_pref("plugin.state.flash", 2);
user_pref("plugins.click_to_play", false);
user_pref("plugins.hide_infobar_for_outdated_plugin", true);
user_pref("javascript.enabled", true);
```