An Empirical Study of Web Cookies

Aaron Cahn
comScore
acahn@comscore.com

Scott Alfeld
University of Wisconsin-Madison
salfeld@cs.wisc.edu

Paul Barford
comScore & University of Wisconsin-Madison
pb@cs.wisc.edu

S. Muthukrishnan
Rutgers University & Microsoft India
muthu@cs.rutgers.edu

ABSTRACT

Web cookies are used widely by publishers and 3rd parties to track users and their behaviors. Despite the ubiquitous use of cookies, there is little prior work on their characteristics such as standard attributes, placement policies, and the knowledge that can be amassed via 3rd party cookies. In this paper, we present an empirical study of web cookie characteristics, placement practices and information transmission. To conduct this study, we implemented a lightweight web crawler that tracks and stores the cookies as it navigates to websites. We use this crawler to collect over 3.2M cookies from the two crawls, separated by 18 months, of the top 100K Alexa web sites. We report on the general cookie characteristics and add context via a cookie category index and website genre labels. We consider privacy implications by examining specific cookie attributes and placement behavior of 3rd party cookies. We find that 3rd party cookies outnumber 1st party cookies by a factor of two, and we illuminate the connection between domain genres and cookie attributes. We find that less than 1% of the entities that place cookies can aggregate information across 75% of web sites. Finally, we consider the issue of information transmission and aggregation by domains via 3rd party cookies. We develop a mathematical framework to quantify user information leakage for a broad class of users, and present findings using real world domains. In particular, we demonstrate the interplay between a domain's footprint across the Internet and the browsing behavior of users, which has significant impact on information transmission.

1. INTRODUCTION

Web cookies were invented in 1994 as a mechanism for enabling state to be maintained between clients and servers. A cookie is a text string that is placed on a client browser when it accesses a given server. The cookie is transmitted back to that server in the header of subsequent requests. Initial support for web cookies was provided in 1994 in pre-1.0 versions of the Mosaic browser [9], and the first standard for web cookies was published in 1997 [19].

Over the years, cookies have remained a central component in the web and their use has expanded as application needs have evolved. 1st party cookies (*i.e.,* cookies placed by the domain shown in the browser's address bar) are commonly used in ecommerce applications and enable *e.g.,* shopping cart persistence. 3rd party cookies (*i.e.,* cookies placed by a domain that is different than what is shown in the browser's address bar) are routinely deployed by data brokerage firms (*e.g.,* Axiom, Datalogix and Epsilon), online advertisers and tracking applications (*e.g.,* Google Analytics).

While cookies are an intrinsic element of web applications, their use has important implications on user privacy. Indeed, one of the primary goals of data brokerage firms and online advertisers is to amass as much information as possible about users toward the goal of delivering targeted ads. Concerns over user privacy and web cookies have been voiced over the years in the popular press (*e.g.,* [28]), in research literature (*e.g.,* [18]) and at the US FTC [4]. This has led to the development of a variety of tools that facilitate cookie management and removal (*e.g.,* [24]), as well as laws that govern user privacy (*e.g.,* [12]).

In this paper, we present an empirical study of web cookies. Despite their ubiquity, importance and potential risks, we are aware of no prior large-scale studies on cookie deployments or cookie characteristics. We seek to answer questions such as what are the range of attributes used for cookies? what is the prevalence of deployments and use of 1st and 3rd party cookies? and can we develop models of cookie deployments to better reason about privacy issues?

Our study consists of a data set collected by implementing and deploying a specialized web crawler that identified 1st and 3rd party cookies placed on client browsers by the Alexa top 100K web sites. This data set comprises over 3.2M cookies. We also appeal to Cookiepedia's category index and a website genre database provided by Blue Coat Systems as a means for grouping domains by interests.

Our analysis of cookie data begins with an examination of the their general characteristics. As expected, we find that 3rd party cookies dominate 1st party cookies in terms of placements. Cookiepedia categories indicate that a large percentage of cookies are placed for the purpose of *performance assessment* and in a way that does not expose user identity details. Our examination of cookie attributes reveals a wide variety of interesting characteristics. In particular, **nearly 80%** of the cookies our crawler harvested were sent insecurely with maximal permissions, as we describe in Section 5.

Next, we drill down on how individual websites set cookies. We find diverse behaviors including over 20 sites that each set over 200 cookies! When we break down sites by genre, we find that sites related to tech/Internet, shopping, news/media set the largest number of cookies. We also consider how 3rd parties place cookies across a spectrum of websites. Our genre-based analysis shows that the majority of 3rd party cookies come from tech/Internet, business/e-

WWW 2016, April 11–15, 2016, Montréal, Québec, Canada.
ACM 978-1-4503-4143-1/16/04.
http://dx.doi.org/10.1145/2872427.2882991.

conomy, and search engines/portals and that the majority are used for targeted advertising. We also find that **less than 1%** of the domains associated with 3rd party cookies have placement capability on **over 75%** of websites we observed.

Based on our empirical findings, we develop a model that elucidates the issue of information leakage. Our approach is based on the notion that a visit to a unique website leaks information to 3rd party services on that site. We then develop a framework to compute the expected value of information leaked as a user browses the web. Using three different models for user browsing behavior, based on website genres, we show how much information several real-world domains are expected to gather as 3rd party agents.

In summary, this paper makes two main contributions: First, we present a first-of-its-kind empirical study of web cookie characteristics by examining over 3.2M cookies collected through targeted crawls. The results of our analysis highlight the broad scope of cookie characteristics as well as setting and placement methods. Second, based on our empirical study, we develop a model for information leakage due to 3rd party cookies. We use this model to show how specific 3rd party cookie entities can amass information about users based on their current footprints and different user browsing patterns.

2. WEB COOKIE OVERVIEW

A cookie is a formatted string consisting of semi-colon separated *key-value* pairs. Throughout this paper we use the terms web cookie and cookie interchangeably. Note, we do not discuss other forms of cookies such as flash, zombie, or edible cookies [29]. A simple cookie would appear as follows:

Name=Value; Host=example.com; Path=/account; Expires=Tue, 1 Dec 2018 10:12:05 UTC; Secure;

Name. The name attribute contains the name given to a cookie sent by a particular server. This uniquely identifies cookies to a particular server.

Value. The value attribute contains the data the cookie is responsible for transmitting between client and server. Value data may be clear text, but is generally encrypted, or obfuscated for security and privacy reasons.

Host. The host attribute identifies the cookie's origin server. This allows a browser to send cookies back to the proper server during subsequent communication. It also distinguishes 1st and 3rd party cookies.

Path. The path attribute restricts when a browser sends a cookie back to a host. The value in the path attribute must exist in the URL of the web site being requested by the browser.

Expires. The expiration attribute contains a datetime string announcing when the cookie should be invalidated. The value in the expires attribute distinguishes session and persistent cookies. Note, Firefox converts the datetime string into an expiry time in seconds.

Secure. The secure attribute is a flag which specifies whether a cookie be transmitted securely via SSL and HTTPS.

HttpOnly. The HttpOnly attribute is a flag which specifies whether a cookie can be accessed programmatically client side.

isDomain. The isDomain attribute is a flag which specifies if a cookie should be sent to a host for URL requests to any subdomain within a site. Note, isDomain is determined by the host attribute and is specific to Firefox.

3. COOKIE AND GENRE COLLECTION

To automate the collection of cookies, we developed a lightweight web crawler. Our crawler communicates with the Firefox extension *Firefly*, manipulating the nsICookie2 and nsICookieManger Javascript XPCOM interfaces. Firefly automates the collection of cookies set via HTTP Set-Cookie headers, embedded JavaScript web beacons, and tracking pixels.

3.1 System Components

Our crawling infrastructure consists of three major components implemented in Python; the *Controller*, *Crawler*, and *Cruncher*.

Controller. The Controller initiates and continuously monitors the Crawler. The Controller is a script that invokes a Crawler in a subprocess with the appropriate configurable parameters. The Controller blocks while the Crawler is running. When the Crawler exits the Controller interprets the return code and performs the appropriate action. This allows the Controller to properly handle errors encountered during the Crawler's execution and restart the Crawler in the event of these errors or a Crawler crash.

Crawler. A Crawler is invoked from the Controller with a workload (*i.e.,* the set of sites to visit). The Crawler spawns an instance of Firefox with the Firefly extension installed. This allows the Crawler to communicate with and manipulate the Firefox browser (*e.g.,* visiting a site). The Crawler harvests cookies by sending a harvest command to Firefly. When the onLoad event fires within the browser, Firefly harvests the current set of cookies in Firefox's cookie jar. Firefly transmits the cookie dump back to the Crawler which then logs the dump in a database for latter processing. After a successful log of the cookie dump, the Crawler issues a command to Firefly to clear Firefox's cookie jar before proceeding to the next site.

Cruncher. The Cruncher runs independently of the Crawler and Controller components of the crawling infrastructure. The Cruncher is responsible for processing the raw cookie dumps logged by the Crawler into a useful format for analysis. To facilitate this, the Cruncher inserts each cookie and a small amount of meta-data as a row in a separate table. Note, unless otherwise stated, this table is the source of all data for the analysis and results we present in the following sections.

3.2 Site Genre Collection

Pulser. Blue Coat Systems[1] provides an interface to query their database with a web site and have it return a set of genres which cumulatively define that web site's genre (*i.e.,* the type of content the site publishes). Blue Coat Systems maintains this database as part of its web security and content filtering services. In total, Blue Coat Systems provides a set of 85 genres. We automate the task of querying the database for genres with Pulser, a script which utilizes Selenium Webdriver. Pulser submits a query web page, scrapes the returned genres, and logs them in a database table. Note, we chose a random sample of sites to manually verify the accuracy of genres Blue Coat Systems returns. We found genres from Blue Coat Systems align with site content for the sample set.

4. COOKIE DATA SETS

The cookies obtained from the crawl campaign were modified with a small amount of meta-data to create a clean, concise data set for analysis. First, where appropriate, we map multiple distinct hosts to a single host. This is done by removing the subdomain and public suffix portion of the host name [2]. We motivate this with the realization that some distinct hosts are, in fact, not distinct. We term a group of semantically non-distinct hosts a "one-off host set". We define "one-off host set" as the set of hosts which differ strictly by subdomain. For example, *www.google.com*,

[1] `https://sitereview.bluecoat.com/sitereview.jsp`

[2] `publicsuffix.org` provides a comprehensive list.

mail.google.co.uk, and *maps.google.com* are all in the same "one-off host set". Note, *mygoogle.com* would not map to the above set. This drastically reduces the number of distinct hosts in the data set. We do not consider entity relations in this study (*i.e.,* Google owns both *google.com* and *doubleclick.net*). Second, we mark every cookie either 1st or 3rd party. This is accomplished by comparing the cookie's host attribute with the domain of the URL the Crawler was visiting when the cookie was set. A match is marked as 1st party.

This data set forms the basis for the analysis and results in Section 5. For efficiency we produce a matrix A of aggregate statistics on cookie attributes. This matrix guides the discussion in Section 5.2.

4.1 Cookie Feature Matrix

The matrix A is a condensed representation of the cookie data set. Each row a_i of A represents an element from the set R where $R = (X \cup Y)$. X is defined as the set of sites visited where a cookie was set and Y is defined as the set of hosts which place one or more cookies on a site in X. Row a_i consists of two feature sets F_{set} and F_{place}. F_{set} is based on the cookies set on a client's browser when visiting site a_i. F_{place} is based on the 3rd party placement behavior of host a_i. Note, host a_i represents a "one-off host set". F_{set} is used in Section 5.4, and F_{place} in Section 5.5. For F_{set} and F_{place}, features are extracted as follows:

Total Cookies. For F_{set}, total cookies is the number of cookies set on the browser while visiting site a_i. For F_{place}, total cookies is the number of 3rd party cookies host a_i places over X.

1st Party. For F_{set}, 1st party is the number of cookies set on the browser by host a_i while visiting site a_i. 1st party has no context in F_{place}.

3rd Party. For F_{set}, 3rd party is the number of cookies set on the browser by a host other than a_i while visiting site a_i. In F_{place}, 3rd party is equivalent to total cookies.

isDomain. For F_{set}, isDomain is the number of cookies with isDomain set true while visiting site a_i. In F_{place}, isDomain is the number of 3rd party cookies placed by host a_i with isDomain set true.

isSecure. For F_{set}, isSecure is the number of cookies on site a_i with the secure flag set. In F_{place}, isSecure is the number of 3rd party cookies host a_i places with the secure flag set.

isHttpOnly. For F_{set}, isHttpOnly is the number of cookies on site a_i with the HttpOnly flag set. In F_{place}, isHttpOnly is the number of 3rd party cookies host a_i places with the HttpOnly flag set.

Path Depth. We define a binary categorization for the path attribute. If the path attribute is set to root, we classify it as path depth == 1. Otherwise, we classify it as path depth > 1. For F_{set}, path depth == 1 and path depth > 1 are the number of cookies on site a_i in each category respectively. For F_{place}, path depth == 1 and path depth > 1 are the number of 3rd party cookies host a_i places over X in each category respectively.

Persistent and Session. For F_{set}, session is the number of cookies on site a_i with expiration set to zero. Persistent is the number of cookies on site a_i with expiration set greater than zero. For F_{place}, session is the number of 3rd party cookies host a_i places with expiration set to zero. Persistent is the number of 3rd party cookies host a_i places with expiration set greater than zero.

Cookiepedia Category. We use the Cookiepedia categories to assign each cookie a category. The categories are *strictly necessary* (required for core functionality), *performance* (to measure site usage statistics), *functionality* (to enhance user experience), *targeting/advertising* (to track users and provide targeted ads), and *unknown*. A complete description for each category is provided on Cookiepedia's web site [3]. For F_{set}, these are the number of cookies on site a_i in each category. For F_{place}, these are the number of 3rd party cookies host a_i places in each category.

4.2 Blue Coat Genres Data Set

The data gathered by Pulser allows each site a_i to be labeled with a set of genres. Blue Coat Systems maintains a description of each genre on their website.

5. COOKIE CHARACTERISTICS

The cookies gathered from our crawl of the top 100K Alexa web sites are summarized in Table 1[3]. Due to page request timeouts, our Crawler successfully visited 95,220 (95,311) web sites. Note, the set of web sites that caused a timeout is likely an artifact of our crawl. This is because web site availability is an on/off function with respect to time. Therefore, the availability of the 100K web sites we visited is biased by the time our Crawler made a request to a specific site. This speaks to the highly dynamic nature of web site availability. Even among the top 100K Alexa web sites, downtime is not uncommon.

Throughout the rest of this section we adopt the nomenclature XX% (YY%) to discuss the results from our two crawl campaigns simultaneously. The first percentage represents the crawl conducted in April of 2015 and the second percentage (*i.e.,* the percentage in paranthesis) reflects the results from the crawl conducted in November of 2013.

5.1 Cookie Collection Overview

Of the web sites our Crawler successfully visited, 94.6% (94.38%) result in at least one cookie being set on a user's browser. This is a substantial increase from 67.4% reported in a 2009 study conducted on a data set obtained using a similar crawl methodology [31] and 43% reported in [33]. Note, only 1st party cookies were collected in the latter study. This highlights a significant increase in the use of cookies over the past few years. In total, our Crawler collected 1,895,023 (1,490,619) cookies.

Note, the choice to visit only landing pages does introduce some bias into our data collection. For instance, perhaps the landing page of a site may set no cookies, but other pages within the site will. In this case, traversing beyond the landing page will produce a different set of results. We leave such a study to future work.

5.1.1 1st Party Cookies

Our corpus contains 31.9% (36.58%) 1st party cookies, whereas [31] report 56.6%. However, this does not necessarily suggest an universal shift away from 1st party cookies. Rather, embedding 3rd party content into web sites, which then set 3rd party cookies has increased rapidly. This growth simply overshadows the use 1st party cookies significantly.

To provide deeper insight into the use of 1st party cookies on web sites we split along Cookiepedia categories and then along cookie attributes within each category. Figure 1 highlights these insights. For instance, an overwhelming majority (among known categories) fall under the performance category. We posit the use of site analytic services (*e.g.,* Google Analytics) are responsible. In contrast, very few 1st party cookies fall under targeting/advertising because there are many viable, simple to use 3rd party targeting/advertising services.

Interestingly, Figure 1 reveals a majority of performance and targeting/advertising cookies are persistent, with root level path and isDomain set. This allows a user to be tracked as they move be-

[3]The crawl campaigns were conducted in April of 2015 and November of 2013. The results in the table reflect the April 2015 crawl.

Table 1: Raw cookie data from the crawl of the top 100K Alexa sites.

	Total	Session	Persistent	isSecure	isDomain	isHttpOnly	Root Level Path
Total Cookies	1,895,023	271,166	1,623,857	6,790	1,543,646	123,866	1,864,062
1st Party Cookies	605,922	150,100	455,822	3,531	426,314	32,329	602,289
3rd Party Cookies	1,289,101	121,066	1,168,035	3,259	1,117,332	91,537	1,261,773

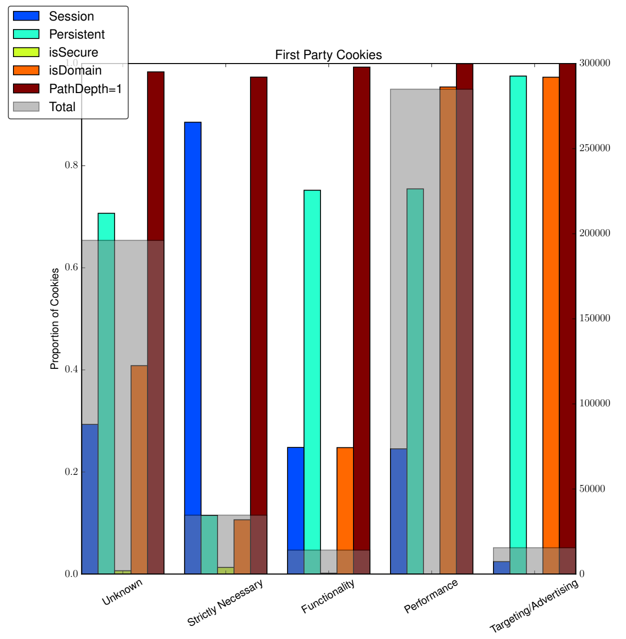

Figure 1: Distribution of first party cookies by Cookiepedia category. The grey bars, using the right-hand scale, show the total number of 1st party cookies. The colored bars, using the left-hand scale, show the normalized values of total first party cookies with different attributes.

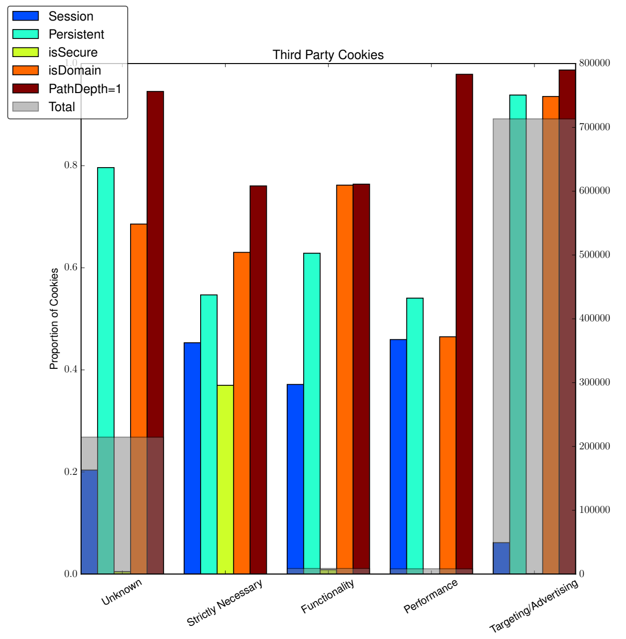

Figure 2: Distribution of third party cookies by Cookiepedia category. The grey bars, using the right-hand scale, show the total number of 3rd party cookies. The colored bars show the normalized values of 3rd party cookies with various attributes.

tween intra-domain pages and correlate this information over multiple browsing sessions. With this information, comprehensive user profiles can be built based on user behavior within the site. These profiles can then be used to provide targeted advertising or tailor the information presented to a user while on the site.

5.1.2 3rd Party Cookies

Our corpus contains 68.0% (63.42%) 3rd party cookies, an increase from 43.4% reported in [31]. This growth can largely be attributed to the adoption and popularity of 3rd party services such as targeted advertising, site analytics, and social media widgets. Interestingly, this space is controlled primarily by a small set of entities. In our corpus, over 60% of 3rd party cookies come from a set of only 50 hosts. In fact, the collection of more and more user information by a decreasing number of 3rd party entities is well documented in the literature [18, 23, 25, 26].

Figure 2 provides a breakdown of 3rd party cookies similar to Figure 1. Perhaps expected, yet still alarming, is the dominance of persistent, targeting/advertising 3rd party cookies with a root level path and isDomain set. With these cookies, 3rd party entities are able to track users over long periods of time, across domains, and on all pages within a domain. These cookies routinely cause alarm in the popular media due to implications regarding user privacy [28].

5.2 Cookie Attributes

In this section we analyze our cookies stratified by attributes. The analysis offers compelling insights into the *(i)* pseudo-standardized nature of cookies and *(ii)* the rapid consolidation of players placing cookies in a 3rd party context.

5.2.1 The Name Attribute

Our corpus contains 153,312 (108,059) cookies with distinct names. The low variation in cookie names is due to *(i)* the use of standard names (*e.g.,* UID, SESSIONID, etc.) by hosts and *(ii)* the fact that a relatively small number of entities constitute the majority of cookie placements on the web. To motivate the latter, consider the five most popular names used: *UID*, *__utma*, *__utmb*, *__utmc*, & *__utmz*. Of these five, the last four all come from Google Analytics, which provides web sites with metrics on how users are visiting their site[4].

5.2.2 The Host Attribute

Our corpus contains 86,835 (96,891) distinct hosts. The top ten 3rd party hosts are all targeting/advertising entities. As seen in Figure 2, only in rare cases, does a 3rd party provide functionality or performance services. In fact, PayPal is the only 3rd party service from our corpus labeled Strictly Necessary.

5.2.3 The Expiration Attribute

Our corpus contains 14.3% (17.2%) and 85.7% (82.8%) session and persistent cookies respectively. Persistent cookies appear over **six times** as frequently as session cookies. This divide is even more stark amongst 3rd party cookies. We see 9.4% (10.03%) and

[4]developers.google.com/analytics/devguides/collection/analyticsjs/cookie-usage

90.6% (89.97%) 3rd party session and persistent cookies respectively. Therefore, a user is **nine times** as likely to be tracked by a 3rd party service over multiple browsing sessions. In fact, DoubleClick cookies appeared on 42.1% (41.88%) of sites we successfully crawled during the crawl campaign.

5.2.4 *The Secure Attribute*

In our corpus only 0.36% (0.24%) of cookies have the secure flag set. Therefore, 99.64% (99.76%) of the cookies were sent unencrypted in HTTP headers. Although there is performance overhead associated with HTTPS [10], HTTP leaves cookies vulnerable to cross site scripting attacks and cross site request forgery [5,8,13]. In other words, these cookies represent a potential attack vector for an adversary to exploit. Note, different browsing behavior (*e.g.,* logging in to a site) may result in the collection of more cookies with the secure flag set due to the sensitive nature of such actions. We do not consider such behavior in this study.

5.2.5 *The HttpOnly Attribute*

Only 6.5% of cookies in our corpus have the HttpOnly flag set. As a result, 93.5% of the cookies are accessible programmatically client side. Similar to the secure flag, these cookies expose a potential attack vector (*e.g.,* cross site scripting and cross site request forgery attacks). These exploits can lead to session hijacking, allowing a malicious user to masquerade as someone else; potentially stealing sensitive information or performing damaging actions [7, 27]. Considering this attack vector can be nullified if web developers simply set the HttpOnly flag during cookie creation, the lack of adoption is astonishing [34]. Note, we did not collect data on the HttpOnly attribute during the first crawl campaign.

5.2.6 *The isDomain Attribute*

Our corpus contains 81.5% (80.47%) of cookies with the isDomain flag set. As discussed above, cookies set while visiting one subdomain are subsequently sent during page requests for all other subdomains. This allows a user to be identified and tracked by a company across any subdomain of a given site. In the case of a 3rd party cookie, the user may be completely unaware they are being tracked because, by virtue of it being a 3rd party cookie, the cookie does not originate from the site the user is currently on. In fact, it is extremely likely a user may have never visited the site of the 3rd party service placing the cookie.

5.2.7 *The Path Attribute*

Our corpus contains 98.4% (98.17%) of cookies with a root level path. By setting a cookie's path root, a user can be identified and tracked while visiting any page within a domain. The sentiments expressed in the previous subsection apply directly here as well.

5.3 Security and Privacy Concerns

In total, **80.6% (79.73%)** of the cookies harvested have maximal permissions *e.g.,* a root level path, isDomain set, and secure not set. Note, for the second crawl we also add HttpOnly not set to the definition which changes the percentage from 80.6% to 76.1%. The issue with maximal permission cookies is two fold.

In the 1st party context, security is the primary concern. Setting cookies with maximal permissions increases an attacker's ability to sniff a cookie during transmission between browser and server. Since 1st party cookies are generally used to provide user functionality (*e.g.,* login), interception of these cookies can lead to session hijacking and if the site has a vulnerability, a cross site scripting attack. **68.0%** of the 1st party cookies in the corpus are set with maximal permissions.

In contrast, in the 3rd party context, privacy is the main concern. The considerable use of such cookies creates an environment where cookies are continuously sent between browser and server. This behavior magnifies the diffusion of user information and unnecessarily escalates potential interception by an adversary. **80.0%** of the 3rd party cookies in the corpus are set with maximal permissions.

These results have several implications. First, users could make better judgments about how they manage their browser if they had more visibility into how cookies being set by both 1st and 3rd parties are being utilized. Currently, there are a few client side tools that provide this functionality *e.g.,* Ghostery [11]. Second, it is unlikely that the use cases for the majority of these cookies justifies their attributes being set this way. This speaks to the need for broader conversations about best practices on cookie usage among developers.

5.4 Cookie Setting Behavior

The 1st and 3rd party cookies set on the browser while visiting a web site defines that web site's cookie setting behavior. In particular, this section utilizes F_{set}.

Our corpus contains 67,587 (65,258) sites which set both 1st and 3rd party cookies. Cookie setting behavior varies drastically from site to site. Specifically, cookie setting on a web site ranges from 0 to 312 with an average of 19 cookies being set per web site.

There is similar variability amongst the use of 1st and 3rd party cookies. 17,458 (19,172) sites exclusively set 1st party cookies and 5,034 (5,526) sites exclusively set 3rd party cookies. This differentiates sites which exclusively track their own users and those which delegate the task to 3rd party entities.

In the following two subsections we stratify our cookie collection by genre and then again by category to further highlight cookie setting behavior.

5.4.1 *Setting Behavior by Genre*

Figure 3 depicts the proportion of cookies set on sites in row r's genre which came from a host in column c's genre. Each row sums to one and column stripes illuminate the fact that most site genres set cookies from a few host genres. For example, the majority of cookies a site sets come from hosts labeled with Technology/Internet, Shopping, News/Media, Business/Economy, and Entertainment genres.

5.4.2 *Cookie Category Setting Behavior by Genre*

Figure 4 shows the distribution of Cookiepedia categories within the most common genres. Targeting/Advertising cookies are highest in all genres followed by performance, with only a few Strictly Necessary and Functionality cookies. Note, Government/Legal is the one of the few genres where Performance cookies are set almost as much as Targeting/Advertising.

5.5 Cookie Placement Behavior

A host's cookie placement behavior is defined by the cookies the host places on web sites. This set of web sites induces what we define as the host's footprint. Note, this section utilizes F_{place} and therefore deals exclusively with 3rd party cookies.

We provide a quantification of 3rd party cookie placement in Table 2. The footprints of the largest 3rd party entities demonstrate their extensive reach across the Internet. In fact, in aggregate, these ten 3rd entities cover **51.90%** of the sites we crawled. This means that much of the identifiable information users leak as they browse the web (as discussed in Section 5.2) is collected by a small group of 3rd party entities. That is, a minority of 3rd party services gather user information from a majority of web sites. This inversion of power allows for extensive data aggregation, which can then be sold and distributed as seen fit by these 3rd party services. We explore this phenomenon further and provide a framework to quantify the amount of information gathered based on user behavior in Sections 6 and 7.

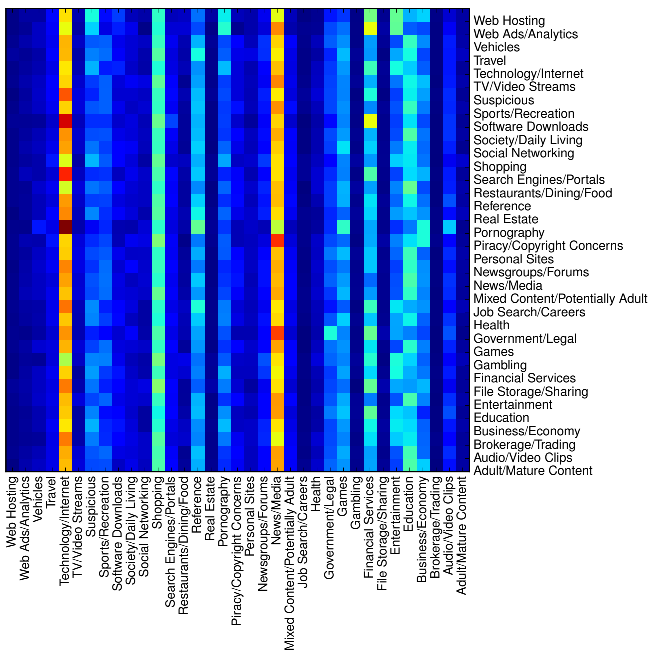
Figure 3: A heatmap showing the distribution of cookies set. The (r, c)-th entry is the proportion of cookies set on genre r sites from genre c hosts. For example, the square corresponding to the Travel row and Health column is the proportion of cookies set on Travel sites which are from Health hosts.

3rd Party Host	% Unique Web Sites
doubleclick.net	42.1% (41.88%)
scorecardresearch.com	15.8% (18.19%)
yahoo.com	14.6% (7.53%)
google.com	13.1% (16.24%)
mathtag.com	11.4% (7.48%)
twitter.com	11.2% (15.36%)
openx.com	10.7% (-)
rubiconproject.com	8.9% (-)
pubmatic.com	8.3% (-)
turn.com	7.6% (6.46%)

Table 2: Top 10 3rd Party hosts in terms of the percentage of unique web sites touched. Note the percentages in paranthesis are the values from the November 2013 crawl. A dash means that host was not previously in the top 10.

5.5.1 *Placement Behavior by Genre*

Figure 5 depicts the proportion of cookies placed by column c's genre which appear on sites of genre r. Each column sums to one and row stripes illuminate the fact that most host genres place cookies on a few site genres. For example, it is clear most cookies are placed (regardless of host genre) on sites with genres Technology/Internet, Shopping, News/Media, Entertainment, and Business/Economy. This correlates well with Figure 3

5.5.2 *Cookie Category Placement Behavior by Genre*

Figure 6 shows the distribution of hosts amongst genres and categories. The concentration of 3rd party cookies in a few genres (Technology/Internet, Business/Economy, and Search Engines/Portals) is indicative of the ubiquitous use of a few major 3rd party services. Specifically, these 3rd services overwhelming provide Targeting/Advertising. This stands in contrast to Figure 4 which has a more even distribution among genres.

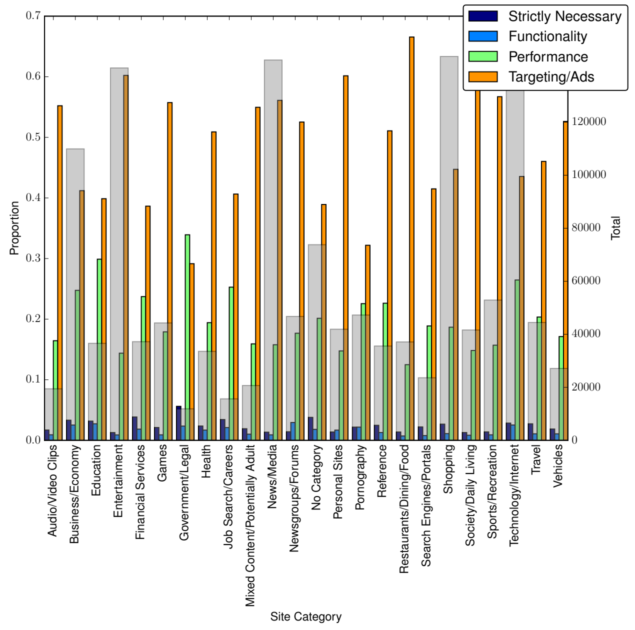
Figure 4: Cookiepedia category setting behavior by site genre. The grey bars, using the right-hand scale, show the total number of cookies set. The colored bars, using the left-hand scale, show the proportion of cookies of various Cookiepedia categories.

Domain	WFPS (%)	FPS (%)	Difference
google	0.080 (8.0%)	12,499 (13%)	-0.051
baidu	0.033 (3.3%)	6,086 (6.4%)	-0.031
twitter	0.082 (8.2%)	10,682 (11.2%)	-0.030
youtube	0.037 (3.7%)	6,136 (6.4%)	-0.028
addthis	0.039 (3.9%)	6,141 (6.4%)	-0.025
bing	0.019 (1.9%)	1,078 (1.1%)	0.008
qq	0.011 (1.1%)	224 (0.24%)	0.009
sina	0.012 (1.2%)	242 (0.24%)	0.010
microsoft	0.010 (1.0%)	40 (0.04%)	0.010
live	0.014 (1.4%)	16 (0.02%)	0.013

Table 3: The Weighted Footprint Size (WFPS), and Footprint Size (FPS) and difference for 10 domains. The first five domains in the table are those in the top 1,000 domains with the lowest footprint difference. The second five are those with the highest footprint difference. The percentage with WFPS is the percentage of traffic the domain is expected to observe, and the percentage with FPS is the percentage of sites within the domain's footprint.

We note two exceptions to the distribution described above. First, the majority of cookies from the Financial Services genre are categorized as Strictly Necessary. We posit this is because the Financial Services genre using cookies to transmit sensitive information and provide critical site components. Second, Business/Economy, Health, and Shopping genres place a considerable amount of Functionality cookies. Hosts associated with these genres provide embedded content such as interactive widgets or shopping cart services.

6. IDENTIFYING DOMAIN FOOTPRINTS

We define the *footprint*, $C^{(D)}$, of a domain D to be the set of sites on which D places a cookie (not including D itself), and the *footprint size* (FPS) of the domain to be the number of sites within its footprint, $|C^{(D)}|$. The footprint size of D divided by the total number of sites in our study yields the *normalized footprint size* – the proportion of sites on which D places cookies. The normalized footprint size of several hosts is extensive, and as a result during a typical browsing session a user is likely to have one or more of

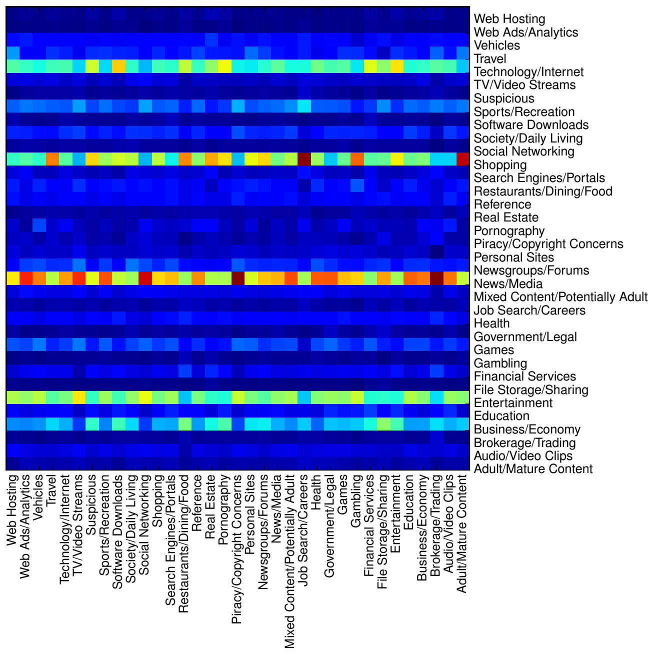

Figure 5: A heatmap showing the distribution of cookies placed. The (r, c)-th entry is the proportion of cookies placed by genre c that are on genre r sites. For example, the square corresponding to the Travel row and Health column is the proportion of cookies Health hosts placed that are on Travel sites.

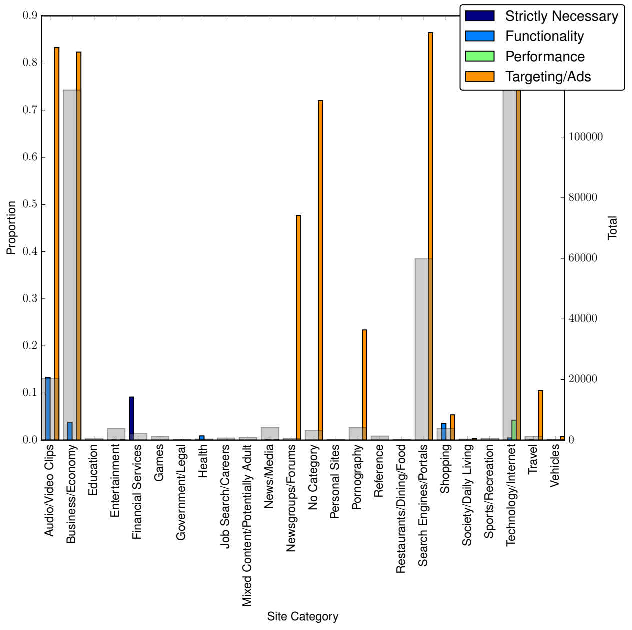

Figure 6: Cookipedia category placing behavior by site genre. The grey bars, using the right-hand scale, show the total number of cookies placed per genre. The colored bars show the proportion of cookies placed of each Cookiepedia category.

these hosts place a persistent 3rd party cookie allowing tracking of the user over a long period of time. For example, DoubleClick sets cookies on over 40% of the top Alexa sites, while Twitter, Google, and ScorecardResearch cover over 15% each. We find that **less than 1%** of domains cover in aggregate **over 75% of web sites**.

To measure the proportion of traffic rather than sites, we define the *weighted footprint size* (WFPS). We model users as making independent and identically distributed page visits. That is, a user U makes a visit v_1 by selecting a page s_1 based on some distribution $\mathscr{D}_U$, and then selects a visit v_2 by drawing independently from the same distribution, and so on. Given such a distribution over sites, we define the weighted footprint size to be the probability that U visits a site on which D places a cookie: $P(s \in C^{(D)})$, where s is drawn from $\mathscr{D}_U$. Phrased differently, this is the expected proportion of unique site visits by U to those setting cookies from D. Note that this is a proxy for the amount of inter-domain information U leaks to D. We explicitly calculate the amount of information leakage, and illustrate the connection to weighted footprint size in section 7.

A weighted footprint can be calculated using any distribution $\mathscr{D}_U$. For this work, we model a user's distribution based on the Alexa ranks so as to approximate actual user profiles. For the top ranked Alexa sites, we compute the footprint size and weighted footprint size under the assumption that $P(v = s_i)$ is a Zipf distribution with parameter 1: $P(v = s_i) \propto 1/\text{rank}(s_i)$.

We define the *footprint difference* of a domain to be its weighted footprint size minus its normalized footprint size. That is, the proportion of users it reaches minus the proportion of sites on which it places cookies. The footprint difference therefore ranges from -1 to 1 where high values indicate many users receive cookies while relatively few are placed, and low values indicate that many cookies are placed with relatively few set on users' browsers. Table 3 shows the FPS and WFPS of the top and bottom five domains in terms of footprint differences in our data set. In general we see that domains with low (negative) footprint differences have extremely large footprints (*e.g.,* Google and Twitter), and those with high (positive) footprint differences have extremely small footprints (*e.g,* Microsoft and Live). A notable exception is Bing, with its relatively large FPS (1,078) but positive difference. This indicates that it reaches considerably more traffic (as estimated by weighted footprint size) than those of similar footprint size (for example Rambler, FPS: 1,024, WFPS: .005 and Histats FPS: 1380, WFPS: .004).

7. USER INFORMATION LEAKAGE

A simple definition of *user information leakage* in the context of web browsing is a user's unintentional transmission of information about their browsing behavior. Our empirical analysis serves as the basis for developing a model for user information leakage via cookies. Specifically, we develop a mathematical model that enables quantification of user information leakage based on browsing behavior and cookie placement footprint. We then relate the weighted footprint size of a domain to the information gleaned from our model.

As a user U browses the Internet, various entities gather information about $U's$ behavior. This information comes in two forms: intra-domain and inter-domain. Intra-domain information pertains to what U does within a domain, and consists of what intra-domain links are clicked, what U searches for, and other page-related activities. Inter-domain information comes from a user visiting pages of different domains. Our motivation for studying user information leakage is to enable the assessment of user privacy and to aid the development of effective systems for preserving privacy.

We first describe the mathematics of calculating the amount of information leakage for an arbitrary user and domain in our framework. We then define several user profiles and examine the rate of information leakage for five domains selected from Table 3: Google, Twitter, Youtube, Yahoo, Rubiconproject. We show a rich and varied relation between users, domains, and the rate at which information is obtained. Specifically, we examine an example where Google obtains more information from a user for the first 4,000 individual page visits, soon after which Twitter is expected to have obtained more information. We further demonstrate that our previously described measure of weighted footprint size serves as a proxy for the more in-depth (and computationally costly) measures.

7.1 Mathematical Framework

Consider a user U and a domain D. As before, we model a user as making n *i.i.d.* page visits $V = v_1, \ldots, v_n$, drawn from a distribution $\mathscr{D}_U$. We let the amount of information D obtains about U (from the sequence of visits) be denoted as $h_D(V)$, for which we make the following assumptions. *(i)* Visiting a site s leaks information to D only if D places a cookie on s ($s \in C^{(D)}$). *(ii)* Any visit to s, after the first, yields no additional information. *(iii)* The order in which U visits pages is independent of the amount of information leaked. *(iv)* h_D can be decomposed into a linear combination of functions on individual sites. Assumptions *(i)*, *(ii)*, and *(iii)* are to fit a natural model of information leakage via cookies. Specifically, we are measuring inter-domain information leakage, not intra-domain. Assumption *(iv)* is for mathematical simplicity.

To formalize these assumptions, we restrict h_D to the form:

$$h_D(V) = \sum_{s \in \text{set}(V)} f_D(s) \tag{1}$$

where $f_D : S \to \mathbb{R}_+$ is a function measuring the value (in terms of how much information is provided) of site s and $\text{set}(V) = \{v \mid v \in V\}$. We denote the expected information gained as

$$H_U^{(D)}(n) = \mathop{\mathbb{E}}_{V \sim \mathscr{D}_U^n} h_D(V) \tag{2}$$

That is, **the expected value (over possible sequences V of n visits) of inter-domain information leaked by U to D.** Recall that $C^{(D)}$ is the footprint of D, and let $C^{(U)} \triangleq \{s \mid P(v = s \mid U) > 0\}$ be the set of all sites U could visit where $P(\cdot)$ denotes probability with respect to $\mathscr{D}_U$.

We model this as a balls-and-bins problem, where the bins are the sites $s_1, \ldots s_K$ and the balls are the visits. Each bin has a value $f_D(s_i)$, and we denote the bins corresponding to sites in $C^{(D)}$ as red. The question then becomes what is the sum of values of red bins which have at least one ball. Unlike the more common case of balls in bins which assumes a uniform distribution over bins, each of our bins has a (potentially) different probability.

Let Y_i be the indicator variable that bin s_i has at least one ball. We have

$$\begin{aligned} \mathbb{E}Y_i &= P(Y_i) = 1 - P(\neg Y_i) \\ &= 1 - (1 - P(v = s_i \mid U))^n \end{aligned} \tag{3}$$

where we have used the fact that the probability of the bin having zero balls is $(1 - P(v = s_i \mid U))^n$ because draws are *i.i.d.*. Note that the expected number of bins with at least one ball may be written as

$$\mathbb{E}\sum_{i=1}^{K} Y_i = \sum_{i=1}^{K} \mathbb{E}Y_i = \sum_{i=1}^{K} P(Y_i) \tag{4}$$

Incorporating the values $f(s_i)$ and the restriction to D's footprint, we obtain

$$\begin{aligned} H_U^{(D)}(n) &= \sum_{s_i \in C^{(U)}} \mathbb{1}_{(s_i \in C^{(D)})} f_D(s_i) P(Y_i) \\ &= \sum_{s_i \in C^{(U)}} \mathbb{1}_{(s_i \in C^{(D)})} f_D(s_i) \left(1 - \left(1 - P(v = s_i \mid U)\right)^n\right) \end{aligned} \tag{5}$$

where $\mathbb{1}_{(\cdot)}$ is the 0-1 indicator function.

7.2 Experimental Setup

To demonstrate our framework, we examine 3 hypothetical users leaking information to real world domains: the Social Networking user, the Education User, and the News/Media User. We define $\mathscr{D}_U$ for a genre g by applying a Zipf distribution with parameter 1 to the top 500 sites of genre g. For example the Education User visits the top 500 sites with genre Education, each with probability as determined by a Zipf distribution. That is, $P(v = s \mid U_g) \propto 1/\text{rank}_g(s)$, where $\text{rank}_g(s)$ is the rank of site s amongst sites of genre g as reported by Alexa. To remain agnostic and general, and to allow direct comparison with weighted footprint sizes, we let $f_D(s) = 1$ for all D, s. That is, all sites provide the same amount (1 unit) of information. Note that our framework is robust to any discrete, bounded distribution – we use Zipf here only for illustrative purposes. We then consider five domains, using their footprints over the 500 sites for each user. The genres and domains we selected are summarized in Table 4.

For these artificial user definitions and real world domains, we conducted a case study to demonstrate the power and flexibility of our framework. Given arbitrary (bounded, discrete) user behavior distributions and arbitrary domain footprints, our methods illuminate the amount of information gathered as the number of visits n increases.

For each user, we computed the expected amount of information leaked using equation (5) for $n = 1, \ldots, 5000$ visits. Depending on the information collecting entity, different values of n are important. In particular, values of n correspond to user browsing frequency. Suppose that the information that D collects is only valuable to D for one week. With the knowledge of how frequently a set of users visit pages, D can select the appropriate value of n to consider. For example, if users visit roughly 100 pages a day, then $n = 700$ will represent the number of visits during the time the information remains valuable.

While the raw $H_U^{(D)}$ values are illustrative in their own right, we use two additional methods of normalization to gain further insights into how information is being leaked. The three methods of normalizing the expected values of information leaked are: Raw, Total Information, and Footprint Size.

Raw. We call the expected value $H_U^{(D)}$ for a domain D and user U to be the "raw" amount of (inter-domain) information gathered. In words, "how much information in expectation will D gather after n visits by U?"

Total Information. Consider the case where U makes 100 visits and D places a cookie on every site on the Internet. U may only have visited 50 unique sites, and thus the raw information gathered is 50. However, D has obtained 100% of the information that U has revealed. This motivates normalizing by total information. Namely, we divide $H_U^{(D)}$ by the expected number of unique sites that U has visited ($\mathbb{E}|\text{set}(V)|$). In words, "of all the information U leaks in n visits, how much is D expected to gather?"

Footprint Size. Suppose D places cookies on 50 sites. After U has visited those 50 sites, D will never gain additional inter-domain information about U. However, when we normalize by Total Information, D's information will decrease (as U continues to visit new sites). We therefore also consider the amount of information D has gathered relative to the total amount of information they will ever gather. That is, we divide $H_U^{(D)}$ by the footprint size of D. In words, "of all the information D could gather about U, how much will it gather in expectation after n visits by U?"

In addition, for each user, domain pair, we constructed two synthetic domains, which we illustrate with an example. Suppose D has a footprint size of 100. We construct D_{high} and D_{low} with footprint sizes 100. We construct D_{high} by placing cookies on the 100 most probable sites, thus maximizing its weighted footprint size. Similarly, we construct D_{low} by placing cookies on the 100 least probable sites, minimizing its weighted footprint size. The infor-

	Google		Twitter		Youtube		Yahoo		Rubiconproject	
User Genre	FPS	WFPS	FPS	WFPS	FPS	WFPS	FPS	WFPS	FPS	WFPS
Business/Economy	55	0.0813	92	0.1432	31	0.0275	112	0.1903	61	0.1141
Education	48	0.2262	56	0.0553	37	0.1193	59	0.0694	27	0.0346
Technology/Internet	63	0.0751	76	0.1169	39	0.0353	101	0.1083	48	0.0677
News/Media	102	0.1133	94	0.1941	36	0.0264	162	0.3000	131	0.3256
Shopping	42	0.0480	76	0.0945	21	0.0129	179	0.4049	146	0.3825
Social Networking	61	0.0671	58	0.0590	28	0.0256	57	0.0562	31	0.0303
Sports/Recreation	98	0.2214	111	0.2555	42	0.0797	163	0.2694	138	0.2635

Table 4: The footprint size (FPS) and weighted footprint size (WFPS) for the domains and user models used in our experiments. Note that the WFPS is based on a Zipf distribution with parameter 1 over the top 500 sites of the user genre.

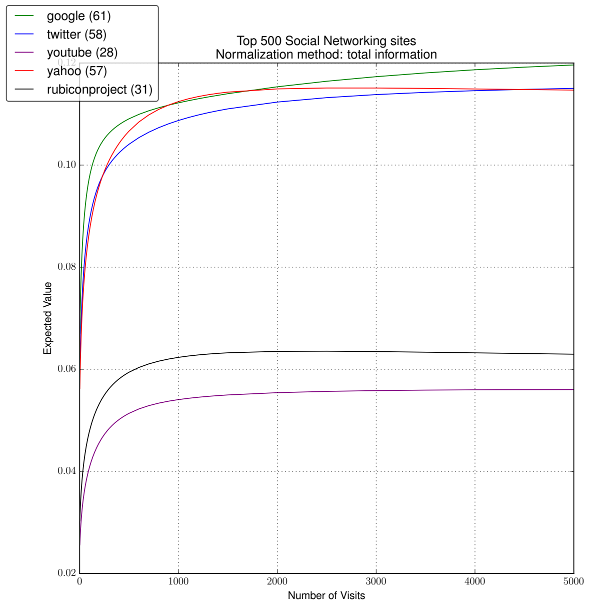

Figure 7: Expected user inter-domain information leakage for the Social Networking user, normalized by total information for the first 5,000 views.

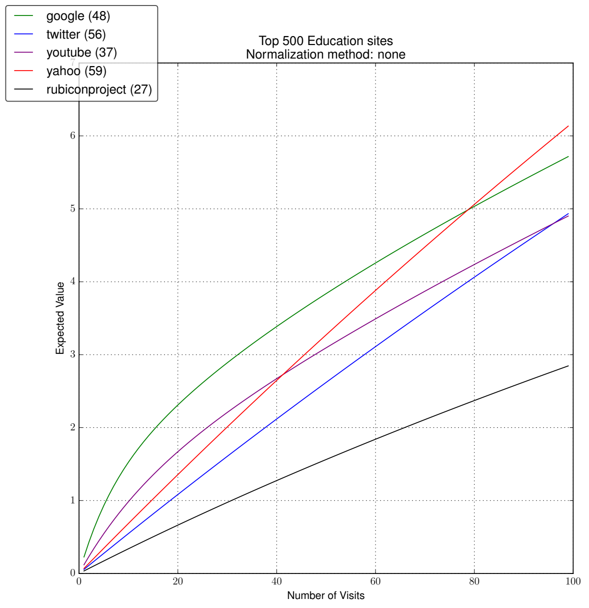

Figure 8: Expected (raw) user inter-domain information leakage for the Education user for the first 100 views.

mation collected by D, therefore, falls between that of D_{high} and D_{low} for all values of n.

7.3 Results

We ran experiments for each user, domain pair shown in Table 4 for $n = 1, \ldots, 5000$, using each normalization process. Due to space limitations, we discuss only a subset of domains for three users, but include statistics and plots for the others.

First, we note that as $n \to \infty$, U will visit all sites in $C^{(U)}$ at least once. Therefore, whatever domain has the largest footprint will obtain the most (raw) information.

Consider first the Social Networking User (see Figure 7), where values are normalized by total information. We note that for small values of n, Google obtains more information than any other domain. For the range from n is roughly 900 to roughly 1700, Yahoo obtains more information than Google, despite its smaller footprint. While Google and Twitter have similar footprint sizes (61, and 58 respectively), Google has a higher weighted footprint size (≈ 0.067 vs ≈ 0.059), meaning that Google reaches more of the traffic and roughly the same proportion of sites. However, weighted footprint size is a good proxy for information gathered only for a low number of visits (note that when $n = 1$, the weighted footprint size is exactly the expected number of sites visited). As n increases, the high-ranking sites in Google's footprint have likely already been visited, and thus Google gains information based only on the rest of its footprint. As n goes to infinity, Google obtains more information than Twitter due to its larger footprint.

For example, suppose Google places a cookie on the top-ranked page for U. After $n = 10$ visits, it is very likely that the user has visited that page. Therefore, from that point on, the contribution of the page to Google's weighted footprint size no longer applies to the same degree in terms of information gathered. It is for this reason that we see Yahoo surpass Google in terms of information gathered after, in this case, roughly 900 visits.

We see similar behavior in the Education User (see Figure 8). As before, Yahoo surpasses Google in terms of total information after a low value of n (around 80, in this case). Note that Google's WFPS for the Education User, however, is far greater than that of Yahoo or Twitter (roughly 0.23 vs 0.06 and 0.07 respectively), despite their similar foot print sizes (48 vs 56 and 59 respectively). Thus, much as before, we see that for low values of n, Google obtains the most information, but as n approaches infinity, Yahoo (with the largest footprint) will eventually obtain the most information.

This phenomenon is further illustrated by Youtube. Even with its small footprint (FPS: 37), Youtube has a relatively large WFPS (roughly 0.12). For $n \leq 40$, we see that Youtube obtains more information leaked by the Education user than any of the other domains, save Google.

We now turn to the News/Media user. Consider Figure 9, where the raw (unnormalized) information values are shown for $n = 1$,

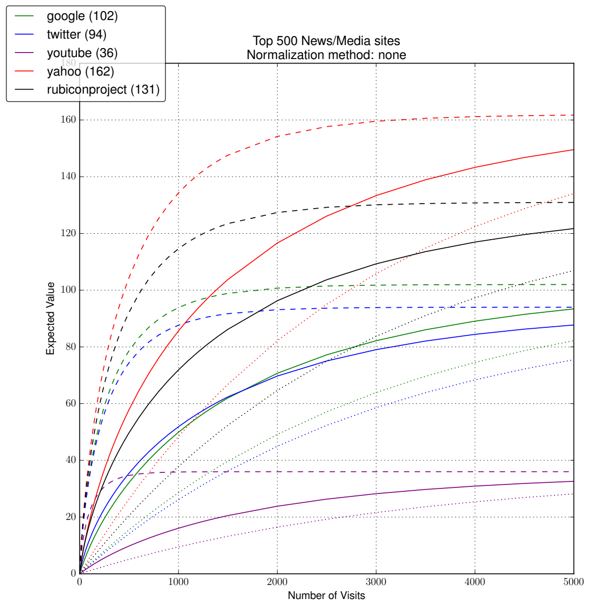

Figure 9: Expected (raw) user inter-domain information leakage for the News/Media user for the first 5,000 views. Dashed lines show D_{high}, dotted lines show D_{low}.

$\dots, 5000$. Yahoo and Rubiconproject have the largest footprint sizes, and we see after roughly $n = 1000$, Yahoo obtains more information than Twitter ever could (as denoted by the blue dashed line). After a few hundred visits, all domains obtain more information than Youtube could, due to Youtube's extremely small footprint (FPS: 36).

Consider Google and Twitter for the News/Media User, see Figure 9. While their footprint sizes are similar (102 and 94 respectively), we again see WFPS serves as a good proxy for low values of n. Namely, Twitter (WFPS: 0.19) gathers more information than Google (WFPS: .11) for $n < 1500$. In addition, we observe that Twitter yields a curve closer to its D_{high} (the dashed line in blue), than Google (the dashed line in green).

Note that as $n \to \infty$, all domains approach their corresponding D_{high}. However, Yahoo, Twitter, and Rubiconproject are all closer to their corresponding D_{high} values for large n than Google and especially Youtube. This indicates that, for up to some value of n, these domains are closer to optimal in their placement of third party cookies than Google or Youtube (given their fixed footprint sizes).

We conclude that while the weighted footprint size is a good proxy for information obtained for low values of n, it is not enough as n increases. By accounting for the full footprint, our framework is able to quantify the information leakage for an arbitrary user distribution. We also note that user behavior in regards to site genre preference plays a large roll in how quickly information is gathered, and by whom.

8. RELATED WORK

Standards for cookies are specified in a series of RFC's [1, 19, 20]. Among other things, these RFCs specify the attributes and syntax of cookies. These serve as a guide for our analysis of specific characteristics of cookies in our data sets.

While we are aware of no prior large-scale empirical study of web cookies, the study by Tappenden *et al.* [31] is similar to ours in terms of the methodology the authors employ to crawl and collect cookies. However, the goal of their work was to highlight the need for better testing of cookies within the web development framework.

There have been a large number of studies on web user privacy and the potential for information leakage via web cookies (*e.g.,* [6, 30, 32]). The study by Mayer and Mitchell provides an insightful survey of privacy issues related to 3rd party tracking and includes discussion of measurement methods that can be used to gather cookies [22]. The work of Roesner *et al.*[26] examines the prevalence and details of 3rd party tracking using a small set of web crawls similar to ours, however the cookies are not their primary research focus.

Our study extends prior work on information leakage. The work by Borders and Prakash in [2] describes an algorithm that quantifies information leakage in HTTP traffic. However, their method is focused on page content rather than cookies. The studies by Krishnamurthy *et al.* on privacy and information leakage are of particular interest [14, 18, 21]. Several of those studies consider information leakage in online social networks [16, 17]. In [15], the authors use a small scale crawl to establish "privacy footprints", which among other things, consider 1st and 3rd party cookie placements. We consider a much larger data set and we take an entirely different approach by developing a mathematical model of privacy leakage.

9. SUMMARY AND CONCLUSIONS

In this paper, we present results of a large-scale, empirical study of web cookies. We use data collected with our specialized web crawler, which gathered 1st and 3rd party cookies from the Alexa top 100K web sites. In total we obtained over three million cookies from two crawls spanning 18 months. Our analysis shows a wide variety of features in the cookies themselves and illuminates the alarming prevalence of powerful, insecure cookies.

We also examine setting and placement characteristics. We find that some websites **set over 300 cookies** and a very small number of 3rd parties have huge placement footprints across the web. Using these findings as a basis, we develop a model to investigate user information leakage. We demonstrate how our framework illuminates the rich interactions between user behaviors and cookie placing behavior, in terms of gathering/leaking information with a case study of synthetic users and real-world domains. Our model shows that the effect of a domain's footprint on information gathered differs dramatically with user, both in terms of preferences (regarding site genres), and frequency of web browsing. We find that even when keeping user interests fixed, the relative amount of information gathered by real-world domains with similar or equal footprint sizes differs based on browsing frequency. We posit that these results and our mathematical framework help to clarify and enhance conversations about user privacy in the web.

In ongoing work we are continuing our data gathering efforts by expanding to a larger number of web sites and by reexamining cookie placements by both 1st and 3rd parties over time. We are enhancing our information leakage framework by considering more detailed models for user browsing behavior *e.g.,* what might be captured in state-transitions. We are also adapting our findings on information leakage to new methods for assessing and ensuring user privacy.

Acknowledgements

This material is based upon work supported by DHS grant BAA 11-01 and AFRL grant FA8750- 12-2-0328. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of DHS or AFRL.

10. REFERENCES

[1] A. Barth. RFC 6265: HTTP State Management System, April 2011.

[2] K. Borders and A. Prakash. Towards Quantification of Network-based Information Leaks via HTTP. In *In Proceedings of the Third USENIX Workshop on Hot Topics in Security (HotSEC)*, San Jose, CA, May 2008.

[3] The Cookie Collective. How We Classify Cookies, 2013. http://cookiepedia.co.uk/classify-cookies.

[4] US Federal Trade Commission. Protecting Consumer Privacy in an Era of Rapid Change: A Proposed Framework for Businesses and Policymakers, December 2010.

[5] Italo Dacosta, Saurabh Chakradeo, Mustaque Ahamad, and Patrick Traynor. One-time Cookies: Preventing Session Hijacking Attacks with Stateless Authentication Tokens. *ACM Transactions on Internet Technololgy*, 12(1):1:1–1:24, July 2012.

[6] Steven Englehardt, Dillon Reisman, Christian Eubank, Peter Zimmerman, Jonathan Mayer, Arvind Narayanan, and Edward W Felten. Cookies that give you away: The surveillance implications of web tracking. In *Proceedings of the 24th International Conference on World Wide Web*, pages 289–299. International World Wide Web Conferences Steering Committee, 2015.

[7] Zachary Evans and Hossain Shahriar. Web session security: Attack and defense techniques. *Case Studies in Secure Computing: Achievements and Trends*, page 389, 2014.

[8] Kevin Fu, Emil Sit, Kendra Smith, and Nick Feamster. Dos and Don'Ts of Client Authentication on the Web. In *Proceedings of the 10th Conference on USENIX Security Symposium - Volume 10*, SSYM'01, pages 19–19, Berkeley, CA, USA, 2001. USENIX Association.

[9] John Giannandrea and Lou Montulli. Persistent Client State: HTTP Cookies, October 1994.

[10] Arthur Goldberg, Robert Buff, and Andrew Schmitt. A comparison of HTTP and HTTPS performance. *Computer Measurement Group, CMG98*, 1998.

[11] Ghostery Inc. Ghostery, 2014.

[12] JISC Legal Information. EU Cookie Directive - Directive 2009/136/EC, April 2010.

[13] Martin Johns. SessionSafe: Implementing XSS Immune Session Handling. In Dieter Gollmann, Jan Meier, and Andrei Sabelfeld, editors, *Computer Security - ESORICS 2006*, volume 4189 of *Lecture Notes in Computer Science*, pages 444–460. Springer Berlin Heidelberg, 2006.

[14] B. Krishnamurthy, D. Malandrino, and C. Wills. Measuring Privacy Loss and the Impact of Privacy Protection in Web Browsing. In *In Proceedings of the Symposium on Usable Privacy and Security*, Pittsburgh, PA, July 2007.

[15] B. Krishnamurthy and C. Wills. Generating a Privacy Footprint on the Internet. In *In Proceedings of the ACM Internet Measurement Conference*, Rio de Janerio, Brazil, October 2006.

[16] B. Krishnamurthy and C. Wills. Characterizing Privacy in Online Social Networks. In *In Proceedings of the ACM SIGCOMM Workshop on Online Social Networks*, Seattle, WA, August 2008.

[17] B. Krishnamurthy and C. Wills. Privacy Leakage in Mobile Online Social Networks . In *In Proceedings of the USENIX Workshop on Online Social Networks*, Boston, MA, June 2010.

[18] Balachander Krishnamurthy and Craig Wills. Privacy Diffusion on the Web: A Longitudinal Perspective. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 541–550, New York, NY, USA, 2009. ACM.

[19] D. Kristol and L. Montulli. RFC 2109: HTTP State Management System, February 1997.

[20] D. Kristol and L. Montulli. RFC 2965: HTTP State Management System, October 2000.

[21] D. Malandrino, L. Serra, A. Petta, V. Scarano, R. Spinelli, and B. Krishnamurthy. Privacy Awareness about Information Leakage: Who knows what about me? In *In Proceedings of the Workshop on Privacy in the Electronic Society*, Berlin, Germany, November 2013.

[22] J. Mayer and J. Mitchell. Third-Party Web Tracking: Policy and Technology. In *In Proceedings of the IEEE Symposium on Security and Privacy*, San Francisco, CA, May 2012.

[23] Jonathan R. Mayer and John C. Mitchell. Third-Party Web Tracking: Policy and Technology. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, SP '12, pages 413–427, Washington, DC, USA, 2012. IEEE Computer Society.

[24] Mozilla. Betterprivacy, May 2014.

[25] Lukasz Olejnik, Tran Minh-Dung, and Claude Castelluccia. Selling Off Privacy at Auction.

[26] Franziska Roesner, Tadayoshi Kohno, and David Wetherall. Detecting and Defending Against Third-party Tracking on the Web. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 12–12, Berkeley, CA, USA, 2012. USENIX Association.

[27] Philippe De Ryck, Lieven Desmet, Frank Piessens, and Martin Johns. Attacks on the user's session. In *Primer on Client-Side Web Security*, SpringerBriefs in Computer Science, pages 69–82. Springer International Publishing, 2014.

[28] J. Schwartz. Giving the Web a Memory Cost Its Users Privacy, September 2001.

[29] Société Des Produits Nestlè. Original Nestlé® Toll House® Chocolate Chip Cookies, 2014.

[30] Ashkan Soltani, Shannon Canty, Quentin Mayo, Lauren Thomas, and Chris Jay Hoofnagle. Flash Cookies and Privacy. 2009.

[31] Andrew F. Tappenden and James Miller. Cookies: A Deployment Study and the Testing Implications. *ACM Transactions on the Web*, 3(3):9:1–9:49, July 2009.

[32] Rodica Tirtea. *Bittersweet cookies some security and privacy considerations*. Heraklion, 2011.

[33] Chuan Yue, Mengjun Xie, and Haining Wang. An Automatic HTTP Cookie Management System. *Computer Networks*, 54(13):2182–2198, September 2010.

[34] Yuchen Zhou and David Evans. Why aren't http-only cookies more widely deployed. *Proceedings of 4th Web*, 2, 2010.