2016 IEEE Symposium on Security and Privacy

The Cracked Cookie Jar: HTTP Cookie Hijacking and the Exposure of Private Information

Suphannee Sivakorn*, Iasonas Polakis* and Angelos D. Keromytis
Department of Computer Science
Columbia University, New York, USA
{suphannee, polakis, angelos}@cs.columbia.edu
*Joint primary authors

***Abstract*—The widespread demand for online privacy, also fueled by widely-publicized demonstrations of session hijacking attacks against popular websites, has spearheaded the increasing deployment of HTTPS. However, many websites still avoid ubiquitous encryption due to performance or compatibility issues. The prevailing approach in these cases is to force critical functionality and sensitive data access over encrypted connections, while allowing more innocuous functionality to be accessed over HTTP. In practice, this approach is prone to flaws that can expose sensitive information or functionality to third parties.**

In this paper, we conduct an in-depth assessment of a diverse set of major websites and explore what functionality and information is exposed to attackers that have hijacked a user's HTTP cookies. We identify a recurring pattern across websites with partially deployed HTTPS; service personalization inadvertently results in the exposure of private information. The separation of functionality across multiple cookies with different scopes and inter-dependencies further complicates matters, as imprecise access control renders restricted account functionality accessible to non-session cookies. Our cookie hijacking study reveals a number of severe flaws; attackers can obtain the user's home and work address and visited websites from Google, Bing and Baidu expose the user's complete search history, and Yahoo allows attackers to extract the contact list and send emails from the user's account. Furthermore, e-commerce vendors such as Amazon and Ebay expose the user's purchase history (partial and full respectively), and almost every website exposes the user's name and email address. Ad networks like Doubleclick can also reveal pages the user has visited. To fully evaluate the practicality and extent of cookie hijacking, we explore multiple aspects of the online ecosystem, including mobile apps, browser security mechanisms, extensions and search bars. To estimate the extent of the threat, we run IRB-approved measurements on a subset of our university's public wireless network for 30 days, and detect over 282K accounts exposing the cookies required for our hijacking attacks. We also explore how users can protect themselves and find that, while mechanisms such as the EFF's HTTPS Everywhere extension can reduce the attack surface, HTTP cookies are still regularly exposed. The privacy implications of these attacks become even more alarming when considering how they can be used to deanonymize Tor users. Our measurements suggest that a significant portion of Tor users may currently be vulnerable to cookie hijacking.

I. Introduction

With an ever-increasing part of our everyday life revolving around the Internet and a large amount of personal data being uploaded to services, ensuring the privacy of our digital communications has become a critical and pressing matter. In the past few years, there has been much discussion regarding the necessity of securing web connections from prying eyes. The publicity garnered by the Firesheep extension [1], which demonstrated how easily attackers can hijack a user's session, was a catalyst in expediting migration of critical user activity to mandatory HTTPS connections in major services (e.g., transmitting user credentials during the log-in process).

Nonetheless, many major websites continue to serve content over unencrypted connections, which exposes the users' HTTP cookies to attackers monitoring their traffic. Not enforcing ubiquitous encrypted connections may be attributed to various reasons, ranging from potential increases to infrastructure costs and the loss of in-network functionality [2] to maintaining support for legacy clients. If access control policies correctly separated privileges of authenticated (e.g., session cookies) and non-authenticated cookies (e.g., persistent tracking cookies), stolen HTTP cookies would not allow attackers to obtain any personal user information. However, that is not the case in practice [3], and things become worse as services continue to sacrifice security over usability. Websites assign privileges to HTTP cookies to personalize functionality, as it improves user experience, but avoid requesting re-authentication unless absolutely necessary, as it impacts user engagement. While session hijacking has been extensively explored, limited attention has been given to the privacy risks of non-session cookies being hijacked; Castelluccia et al. [4] demonstrated how stolen HTTP cookies could allow attackers to reconstruct a user's Google search history.

A subset of the problem we explore has been highlighted in studies that measured the exposure of personal or personally identifiable information (PII) in unencrypted traffic [5]–[8]. However, those studies are limited by nature and do not capture the full extent of the privacy threat that users face due to unencrypted connections. First, modern websites are highly dynamic and information can be fetched in obfuscated form and constructed on the client-side at runtime. Second, websites may only serve private information over encrypted connections, while flawed access control separation renders that information accessible to HTTP cookies (we demonstrate this with Google Maps exposing a user's address in Google Search). Third, eavesdropping is limited to the user's actions for a specific time window, and certain pieces of information require specific actions to be exposed, which may not occur during the monitoring period. Fourth, we find that stolen HTTP

DOI 10.1109/SP.2016.49

cookies can also access account functionality, both *explicitly* (e.g., send an email from the user's account) and *implicitly* (e.g., receive personalized query results from a search engine).

In this paper, we explore the extent and severity of the unsafe practice followed by major services of partially adopting encrypted connections, and its ramifications for user privacy. We demonstrate how HTTP *cookie hijacking attacks* not only enable access to private and sensitive user information, but can also circumvent authentication requirements and gain access to protected account functionality. To our knowledge, this is the first in-depth study exploring the privacy implications of partial adoption of HTTPS. We audit 25 major services, selected from a variety of categories that include search engines and e-commerce sites. In each case, we analyze the use of HTTP cookies, the combination of cookies required to expose different types of information and functionality, and search for inconsistencies in how cookies are evaluated. This allows us to obtain a comprehensive understanding of the feasibility and impact of this class of attacks in practice. We uncover flaws in major websites that allow attackers to obtain a plethora of sensitive user information and also to access protected account functionality. As a precautionary measure, we conduct all experiments on our personal or test accounts.

We conduct an IRB-approved measurement study on a subset of our university's public wireless network, to understand the browsing behavior of users when connected to unprotected public networks. On average, we detect more than 8K unique accounts exposing their cookies for hijacking each day. *Our measurements have the sole purpose of estimating the number of users that are susceptible to hijacking attacks; we do not access any user accounts, collect any personal information, or attempt to deanonymize any users*.

Furthermore, we look at multiple practical aspects of cookie hijacking, and identify how each component of this intricate ecosystem can impact the attacks. We find that partial deployment of HSTS, a security mechanism which is gaining traction and supported by modern browsers, does not present an actual obstacle to cookie hijacking, as unencrypted connections to certain pages or subdomains of a service still expose the cookies. Furthermore, client-side mechanisms like the HTTPS Everywhere extension can reduce the attack surface, but can not protect users when websites do not *support* ubiquitous encryption. We also find that both Chrome and Firefox have a multitude of components that expose users' cookies. And while the apps we test are considerably more secure in Android than in iOS, both platforms have official apps with millions of users that use unencrypted connections.

Due to the practicality of these attacks and the pervasiveness of the vulnerable websites, we investigate how cookie hijacking can lead to the deanonymization of Tor users. In our IRB-approved study, we find that 75% of the outgoing connections from a new exit node are over HTTP. Based on the comparison to the respective measurements from our university's wireless network, we believe that a large number of Tor users may be exposed to HTTP cookie hijacking and susceptible to deanonymization.

Overall, our goal is twofold. First, to alert developers of the pitfalls of partially enforcing HTTPS while offering personalized functionality. Second, to inform users about the protection offered by popular security and privacy-enhancing systems and the caveats of not knowing the precise extent of their protection. The main contributions of this paper are:

- We conduct an in-depth study on the impact and gravity of HTTP cookie hijacking attacks against major services. Our findings demonstrate that a wide range of private information and protected account functionality is accessible. The diversity of these websites suggests that this is a widespread systemic risk of unencrypted connections, and not a topical threat against a specific class of sites.
- Our measurement study demonstrates the extent of the risk; we monitor part of our university's public wireless network over the course of one month, and identify over 282K user accounts that exposed the HTTP cookies required for the hijacking attacks.
- Our analysis on the collateral exposure of cookies shows that browser extensions, search bars, and mobile apps of major vendors expose millions of users to risk.
- We explore how HSTS can impact HTTP cookie hijacking. We demonstrate that partial deployment renders the mechanism ineffective, as a single unencrypted connection may be sufficient for an attacker to obtain the required cookies.
- We describe how major websites can be used as deanonymization vectors against users that rely on the Tor bundle for anonymity, and find that existing mechanisms cannot adequately protect users.
- We disclosed our findings to the services we audited and the Tor community, in an effort to assist them in protecting their users from this significant privacy threat.

The remainder of this paper is structured as follows: in Section II we offer background information, and motivation for our work through a network traffic study. In Section III we offer details on our analysis of cookie hijacking attacks against popular services, and explore the collateral exposure of user cookies by mobile apps and browser components in Section IV. We explore the deanonymization risk that Tor users face in Section V, and discuss general countermeasures against cookie hijacking in Section VI. We address the ethical aspects of our research in Section VII, discuss related work in Section VIII, and conclude in Section IX.

II. Background, Threat Model, and Motivation

In this section we provide a short description of the security mechanisms supported by browsers for protecting users' communications, an overview of our threat model, and motivation through a network traffic analysis study.

A. Browser security mechanisms

In recent years, browsers have included support for various security mechanisms that are designed to protect users from a range of attacks (e.g., [9], [10]). The one most relevant to our work is HSTS, as it can prevent HTTP cookie hijacking

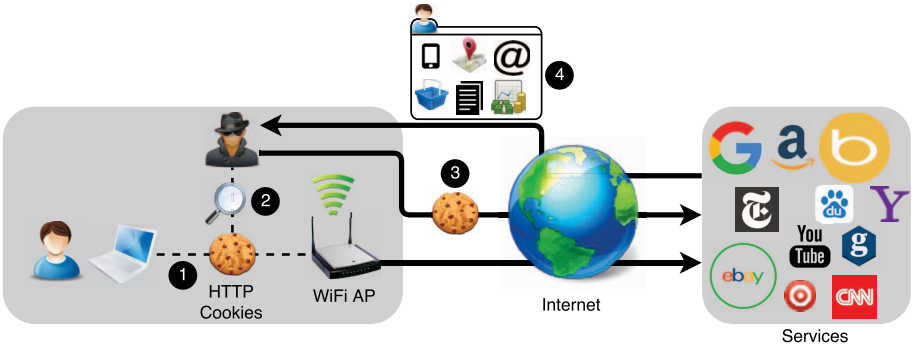


Fig. 1. Workflow of an HTTP cookie hijacking attack. After the victim's cookies are exposed on the unencrypted connection (1) and stolen (2), the attacker can append the stolen cookies when browsing the target websites (3) and gain access to the victim's personal information and account functionality (4).

attacks. However, we also mention certificate pinning, as it is employed in Chrome and Firefox through the HSTS preloading mechanism. We refer the reader to [11] for a more detailed description of HSTS and certificate pinning.

HSTS. The HTTP Strict Transport Security mechanism [12] allows websites to instruct browsers to only initiate communication over HTTPS. This is done through the `Strict-Transport-Security` HTTP header. HSTS is currently supported by all major browsers, and certain mobile browsers [13]. A noteworthy point of failure is during the user's initial request, before the HSTS header is received, which exposes the user to hijacking if sent over HTTP. As a precautionary measure, major browsers rely on a "preloaded list" which proactively instructs them to connect to domains over HTTPS. This protects users during the initial request to a website, and websites can request to be included in the list through an online form[1]. HSTS preloading is currently supported by Chrome, Firefox, Safari, and Internet Explorer [14].

Certificate pinning. Adversaries may create or obtain fraudulent certificates that allow them to impersonate websites as part of man-in-the-middle attacks [15]. To prevent that, websites can specify a (limited) set of hashes for certificates in the website's X.509 public key certificate chain. Browsers are allowed to establish a secure connection to the domain only if at least one of the predefined pinned keys matches one in the certificate chain presented. This was proposed as an extension to HSTS [16], and is currently supported by (at least) Firefox and Chrome. The recent HPKP specification [17] describes an HTTP response header field for pinning certificates.

B. Threat model

Depending on the attacker's ability and resources, a user's HTTP cookies can be hijacked through several techniques. To demonstrate the severity of the threat, we assume the role of a weak adversary and conduct experiments through passive eavesdropping. Nonetheless, we also investigate cookie characteristics that could be exploited by active adversaries for increasing the scale of the attacks.

[1]https://hstspreload.appspot.com/

HTTP cookie hijacking. The adversary monitors the traffic of a public wireless network, e.g., that of a university campus or coffee shop. Figure 1 presents the workflow of a cookie hijacking attack. The user connects to the wireless network to browse the web. The browser appends the user's HTTP cookies to the requests sent in cleartext over the unencrypted connection (1). The traffic is being monitored by the eavesdropper who extracts the user's HTTP cookies from the network trace (2), and connects to the vulnerable services using the stolen cookies (3). The services "identify" the user from the cookies and offer a personalized version of the website, thus, exposing the user's personal information and account functionality to the adversary (4).

Cookie availability. These attacks require the user to have previously logged into the service, for the required cookies to be available. Having closed the browser since the previous log in does not affect the attacks, as these cookies persist across browsing sessions.

Active adversary. Attackers can follow more active approaches, which increase the scale of the attack or remove the requirement of physical proximity to the victims, i.e., being within range of the same WiFi access point. This also enables more invasive attacks. For example, the attacker can inject content to force the user's browser to send requests to specific vulnerable websites and expose the user's cookies, even if the user does not explicitly visit those sites. This could be achieved by compromising the wireless access point or scanning for and compromising vulnerable routers [18]. Furthermore, if the HTTP cookies targeted by the attacker do not have the `HttpOnly` flag set [19], they can be obtained through other means, e.g., XSS attacks [20]. Users of major services can also be exposed to such attacks from affiliated ad networks [21].

State-level adversary. In the past few years there have been many revelations regarding mass user surveillance by intelligence agencies (e.g., the NSA [22]). Such entities could potentially deploy HTTP cookie hijacking attacks for obtaining access to users' personal information. Reports have disclosed that GCHQ and NSA have been collecting user cookies at a large scale as part of user-tracking programs [23], [24]. As we demonstrate in Section III, these collected cookies

TABLE I
STATISTICS OF OUTGOING CONNECTIONS FROM A SUBSET OF OUR CAMPUS' PUBLIC WIRELESS NETWORK FOR 30 DAYS.

Protocol	Connections	Requests	Vulnerable Requests*	Exposed Accounts
HTTP	685,500,365	1,398,044,178	29,908,099	282,459
HTTPS	772,562,024	–	–	–

*HTTP requests to domains that we have audited and found to be vulnerable.

could be used to amass a large amount of sensitive information that is exposed by major websites. Furthermore, in Section V we discuss how Tor users, who are known to be targeted by intelligence agencies [25], can be deanonymized through the hijacked HTTP cookies of major services.

C. Motivation - Network Traffic Study

The feasibility of cookie hijacking attacks by eavesdroppers is dependant on the browsing behavior of users when connected to public wireless networks. If users only visit websites with ubiquitous encryption or employ VPN tunneling solutions, HTTP cookie hijacking can be prevented. We conduct an exploratory study of the traffic passing through the public wireless network of our university's campus.

IRB. Before conducting any experiments, we submitted a request to our Institutional Review Board that clearly described our research goals, collection methodology, and the type of data to be collected. Once the request was approved, we worked closely with the Network Security team of our university's IT department for conducting the data collection and analysis in a secure and privacy-preserving manner.

Data collection. In order to collect the data, we setup a logging module on a network tap that received traffic from multiple wireless access points positioned across our campus. The RSPAN was filtered to only forward outgoing traffic destined to TCP ports 80 and 443, and had a throughput of 40-50 Mb/s, covering approximately 15% of the public wireless outgoing traffic. Our data collection lasted for 30 days. We used the number of TCP SYN packets to calculate the number of connections. When the connection is over HTTP or HTTPS, we capture the destination domain name through the HTTP host header and the TLS SNI extension respectively. For each HTTP request we log the destination domain, and the name of any HTTP cookies appended (e.g., SID). We also calculated a HMAC of the cookie's `value` (the random key was discarded after data collection). The cookie names allow us to verify that users are logged in and susceptible to cookie hijacking for each service, as we have explored the role of each cookie and also identified the subset required for the complete attack (described in Section III).

While we do not log the cookie value for privacy reasons, the keyed hash value allows us to distinguish the same user within a service to obtain a more accurate estimation of the number of exposed accounts. We must note that our approach has limitations, as the numbers we estimate may be higher than the actual numbers; a user's cookie value may have changed over the course of the monitoring period or the user may use multiple devices (e.g., laptop and smartphone). However, some services employ user-identifier cookies, which we leverage for differentiating users even if the other cookie values have changed. Furthermore, we cannot correlate the same user across services as we do not collect source IP addresses or other identifying information; thus, we refer to vulnerable *accounts*. Nonetheless, we consider this to be a small trade-off for preserving users' privacy, and consider our approximation accurate enough to highlight the extent of users being exposed when browsing popular services.

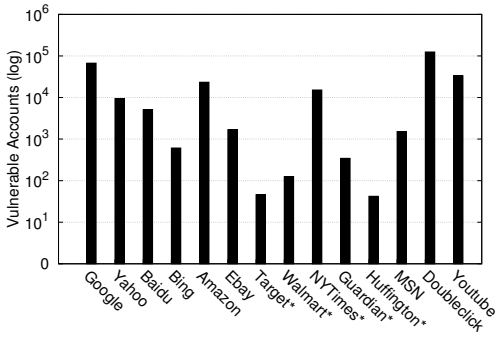


Fig. 2. Number of exposed accounts per service. Services marked with "*" have an explicit userID cookie (or field) that allows us to differentiate users.

Findings. Table I presents the aggregated numbers from the data collected during our study. During our monitoring, we observed more that 29 million requests towards the services that we have found to be vulnerable. This resulted in 282,459 accounts exposing the HTTP cookies required for carrying out the cookie hijacking attacks and gaining access to both their private information and account functionality. Figure 2 breaks the numbers down per service. Search engines tend to expose many logged in users, with 67,201 Google users being exposed during our experiment. Every category of services that we looked at has at least one very popular service that exposes over ten thousand users during the monitoring period. Ad networks also pose a significant risk, as they do not require users to login and ads are shown across a vast number of different websites, which results in Doubleclick exposing more than 124K users to privacy leakage.

III. REAL-WORLD PRIVACY LEAKAGE

In this section, we present our study on the ramifications of HTTP cookie hijacking attacks in real websites. We audit the top Alexa websites from a varied collection of categories using test accounts (or our personal when necessary), and find that HTTP cookie hijacking attacks affect the majority of popular websites we tested. Table II presents an overview of the services and our results. We provide details on the private information and account functionality we are able to access with stolen cookies for certain websites, and describe other classes of attacks that become feasible. Due to space constraints, certain services are described in Appendix A.

TABLE II
OVERVIEW OF THE AUDITED WEBSITES AND SERVICES, THE FEASIBILITY OF COOKIE HIJACKING ATTACKS, AND THE TYPE OF USER INFORMATION AND ACCOUNT FUNCTIONALITY THEY EXPOSE.

Service	HTTPS Adoption	Cookie Hijacking	XSS Cookie Hijacking	Information and Account Functionality Exposed
Google	partial	✓	✗	first and last name, username, email address, profile picture, home and work address, search optimization, click history of websites returned in search results
Baidu	partial	✓	✓	username, email address, profile picture, entire search history, address of any saved location
Bing	partial	✓	✓	first name, profile photo, view/edit search history (incl. images and videos), links clicked from search results, frequent search terms, saved locations, information in interest manager, edit interest manager
Yahoo	partial	✓	✓	username, full name, email address, view/edit search history, view/edit/post answers and questions in Yahoo Answers (anonymous or eponymous), view/edit finance portfolio, view subject and sender of latest incoming emails, extract contact list and send email as user
Youtube	partial	✓	✗	view and change (through pollution attacks) recommended videos and channels
Amazon	partial	✓	✓	view user credentials (username, email address or mobile number), view/edit profile picture, view recommended items, view user wish lists, view recently browsed items, view recently bought items, view/edit items in cart, view shipping name and city, view current balance, view user's review (even anonymous), send email of products or wishlist on behalf of user, obtain email addresses of previously emailed contacts
Ebay	partial	✓	✓	delivery name and address, view/edit items in cart, view/edit purchase history, view items for sale, view previous bids, view user's messages, view/edit watch list and wish lists
MSN	partial	✓	✓	first and last name, email address, profile picture
Walmart	partial	✓	✓	first name, email address, view/edit items in cart, view delivery postcode, write product review
Target	partial	✓	✓	first name, email address, view/edit items in cart, recently viewed items, view and modify wish list, send email about products or wish list
CNN	partial	✓	✓	view/edit profile (full name, postal address, email address, phone number, profile picture) view/edit linked Facebook account, write/delete article comments, recently viewed content on iReport
New York Times	partial	✓	✓	username, email address, view/edit basic profile (display name, location, personal website, bio, profile picture) username, email address, view/edit list of saved articles, share article via email on behalf of user
Huffington Post	partial	✓	partial	profile can be viewed and edited (login name, profile photo, email address, biography, postal code, location, subscriptions, fans, comments and followings). change account password, delete account
The Guardian	partial	✓	✓	username, view public section of profile (profile picture, bio, interests), user's comments, replies, tags and categories of viewed articles, post comments on articles as user
Doubleclick	partial	✓	✓	ads show content targeted to user's profile characteristics or recently viewed content
Skype	partial*	✗	✗	-
LinkedIn	partial*	✗	✗	-
Craigslist	partial*	✗	✗	-
Chase Bank	partial*	✗	✗	-
Bank of America	partial*	✗	✗	-
Facebook	full	✗	✗	N/A
Twitter	full	✗	✗	N/A
Google+	full	✗	✗	N/A
Live (Hotmail)	full	✗	✗	N/A
Gmail	full	✗	✗	N/A
Paypal	full	✗	✗	N/A

*While these services do not have ubiquitous HTTPS, no personalization is offered over HTTP pages.

Threat persistence. Invalidating session cookies when a user logs out is standard practice. High-value services do so even after a short time of user inactivity. We examined whether the services also invalidate the HTTP cookies required for our hijacking attacks. We found that even if the user explicitly logs out *after* the attacker has stolen the cookies, almost all cookies still retain access privileges and can carry out the attack. Thus, attackers can maintain access to the victim's personal information and account functionality until the cookies' set expiration date which can be after several months (Google cookies expire after 2 years). Ebay was the only service out of the vulnerable that invalidates the cookies after logging out. Those cookies do not instruct the browser to expire upon exiting, indicating that Ebay manages the cookies' validity on the server side. Below we also discuss the unusual behavior of Youtube for users that are not logged in.

A. Google

Typically, the adversary can steal the victim's HTTP cookie for Google by observing a connection to any page hosted on `google.com` for which encryption is not enforced.

Cookie hijacking. Google automatically redirects users connecting over HTTP to `google.com` to HTTPS, to protect their searches from eavesdropping. However, upon the initial request, before being redirected and enforcing encrypted communication, the browser will send the HTTP cookies. Furthermore, the user can also use the address bar for visiting Google services; e.g., the user can type "`www.google.com/maps`" to visit Google Maps. Under these usage scenarios the browser will again expose the user's HTTP cookies, and if an adversary is monitoring the traffic, she can hijack them. Redirecting instead of enforcing HTTPS is most likely a conscious decision for supporting legacy clients that do not run HTTPS (outdated User Agents are not redirected).

Browser behavior. The adversary must observe an unencrypted connection to `google.com`, which may not occur under all scenarios. However, a very typical scenario is for the victim to use the browser's address bar. Consequently, to understand the conditions under which the requirements will hold, we explore how popular browsers handle user input in the address bar, when trying to visit `google.com`. As shown

TABLE III
BROWSER BEHAVIOR FOR USER INPUT IN ADDRESS BAR.

Browser	Connect over HTTP
Desktop	
Chrome (v. 45)	✓
Firefox (v. 41)	✓
Safari (v. 8.0)	✓
Internet Explorer (v. 11)	✓
Opera (v. 32)	✓
Mobile	
Safari (iOS 9)	✓
Chrome (v.46, Android 5.1.1)	✗ (conditionally)

*user input: {google.com, www.google.com}

```
//(*.)google.com, iff using SSL, must use an acceptable
    certificate.
{ "name": "google.com", "include_subdomains": true,
    "pins": "google" },

//Now we force HTTPS for subtrees of google.com.
{ "name": "mail.google.com", "include_subdomains": true,
    "mode": "force-https", "pins": "google" },
```

Listing 1. Subset of rules in Chrome's HSTS-preload file.

in Table III, for straightforward user input, popular browsers will connect to google.com over HTTP. Due to the auto-complete feature of certain browsers (e.g., Firefox), even if the victim only types "google", the auto-complete mechanism will add ".com", and the browser will again connect over HTTP. Therefore, under common browsing patterns, the existing design will expose a user's cookie when visiting the main search engine. Interestingly, while the default iOS browser (Safari) exhibits the same behavior, Chrome on Android will connect to Google over HTTPS to securely prefetch page resources. However, if users turn this option off to improve performance[2], Android Chrome will also connect over HTTP.

HSTS preloading. As described in Section II, major browsers employ pre-loading lists for HSTS. As can be seen in Listing 1, the preloaded HSTS policy for Chrome does not actually force the browser to connect to google.com over HTTPS. It does however employ certificate pinning; it requires an acceptable certificate if the browser is *already* connecting over HTTPS. This is applied to all local country-based variations of Google's search engine, and the main page itself. On the other hand, critical Google subdomains support HSTS preloading and are explicitly forced to connect over HTTPS. As a result, users that visit the Google search engine through the address bar, will most likely connect over an unencrypted channel, and their cookies will be exposed.

Information leakage. If the adversary simply visits google.com using the stolen cookie, no sensitive information will be accessible as the browser is redirected to HTTPS. However, if the adversary "forces" the browser to visit Google over HTTP, sensitive information can be accessed. During our auditing we have identified the following.

[2]https://support.google.com/chrome/answer/1385029

Personal information. Due to the cookie, Google considers the victim logged-in, resulting in personal information being leaked. As can be seen in Figure 3(a), we gain access to the user's name and surname, Gmail address, and profile picture.

Location. Google Maps allows users to set their Home and Work addresses, for easily obtaining directions to/from other destinations. While Google Maps requires HTTPS, which prevents us from acquiring any information, if the adversary connects to google.com over HTTP and searches for "home" or "work", the search results will contain a widget of Google Maps revealing the respective address. An example can be seen in Figure 3(b). Accessibility to location information can expose the user to physical threats [26], [27].

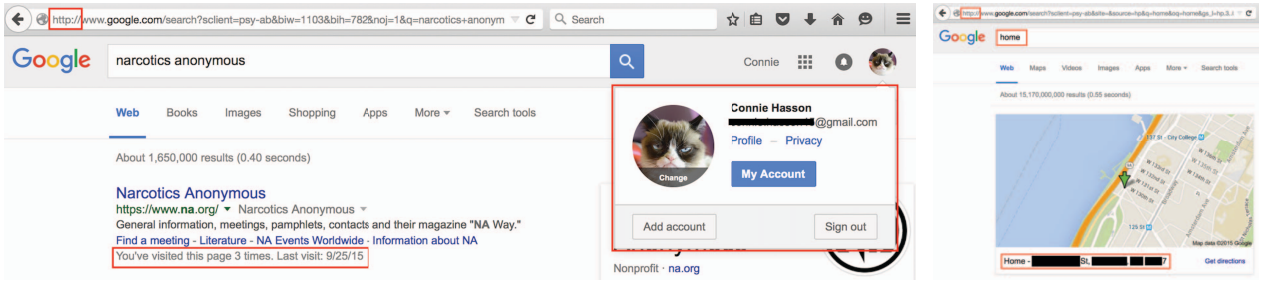
Browsing history. Using the stolen cookie, the adversary can start issuing Google searches for various terms of interest. If the search results contain links that the user has previously visited through the search engine, Google will reveal how many times the page has been visited and the date of the last visit. Users can opt-out of historical information being included in their search results, however, this option is enabled by default. If enabled, the adversary can search for a variety of terms and infer sensitive data about the user. Figure 3(a) shows an example scenario where the adversary obtains such information. Depending on the attacker's goal, she could employ a precompiled dictionary of sensitive keywords for finding sensitive web activity, or a dictionary of the most popular Google search terms for recovering parts of the user's web visiting history. While previous work demonstrated that unencrypted sessions could enable attackers to reconstruct a user's Google search history [4], this is the first, to our knowledge, attack that discovers webpages *visited* by the user through Google.

Exploiting search optimization. Google search may return results that have been personalized for the user, either by inserting specific entries, or changing the rank of specific results. Previous work has demonstrated a methodology for measuring personalization in Google search results [28]. By adapting this technique, the adversary can extract entries from the search results that have been returned based on characteristics of the victim's profile.

Shopping. Using the HTTP google.com cookie when visiting Google's shopping page, which runs mainly over HTTP, will reveal the user's first and last name, Gmail handle, Google profile. It also allows viewing and editing the user's shortlist (items under consideration).

Pollution attacks. If the attacker issues search queries using the stolen cookies, the search terms are treated as if originating from the user and added to the search history. This allows the adversary to affect the victim's contextual and persistent search personalization through pollution attacks [29].

Youtube exhibits a strange behavior that we did not come across in other services. If the victim is logged in, the stolen cookie does not reveal any information. However, if the victim is not logged in, the cookie that is exposed gives access to the user's recommended channels and videos, which can be changed through pollution attacks. Furthermore, information



(a) Profile and History (b) Location

Fig. 3. Private information obtainable from user's Google account through HTTP cookie hijacking.

about the user's music interests can be used to infer private attributes [30].

B. Bing

According to a recent report [31], Bing handles approximately 20.4% of the desktop searches originating from the U.S. Bing is also the default search engine for Siri, iPhone's voice-driven assistant, as well as all Microsoft-based products. When auditing Bing we found that, by default, all connections are served over HTTP, i.e., all searches are sent in clear-text. Users have to explicitly type `https` in the browser's address bar to be protected from eavesdropping.

Personal information. Bing will expose the user's first name and profile photo. The profile photo can be used to obtain more information of the user through face recognition and publicly available data in other websites [32].

Location. If the victim has saved any locations on Bing Maps they are also exposed. Apart from the work or home addresses, this may include other locations the user has visited in the past (e.g., bars, health clinics).

Interest Manager. This recently introduced feature, allows users to select interests from a variety of topics. Based on the category, this can reveal private information including financial assets and political inclination.

Search and browsing history. Once the adversary steals the cookie, she can retrieve the user's search history, including those in the images and videos categories. Apart from a widget displaying the users most recent and most frequent search queries, the search history page also reveals the page that the user visited from each search.

Pollution attacks. The attacker can also issue search queries for conducting a pollution attack and, subsequently, delete those entries for stealthiness. This will remove any trails of the attack, and prevent the victim from detecting it.

C. Yahoo

Depending on the type of browser, and whether it is being run for the first time, visiting `yahoo.com` through the address bar will either connect to HTTP and then redirect to HTTPS, or maintain the unencrypted connection. However, links in the main Yahoo page are all redirected through `http://hsrd.yahoo.com`. Even if the user explicitly

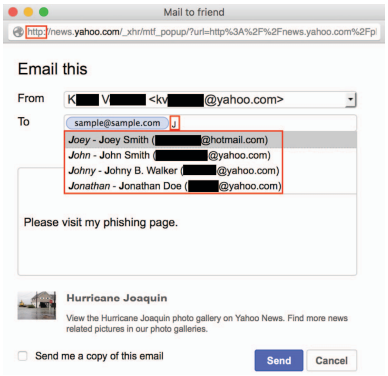


Fig. 4. Extracting contact list and sending email from the victim's account in Yahoo.

connects to Yahoo over HTTPS, if any link in the homepage is clicked, it will connect to that subdomain over an unencrypted connection. Therefore, regardless of how the victim connects, we have identified three HTTP cookies (Y, F, T) that are exposed to eavesdroppers. We first describe the information and functionality that attackers can access, and then how we perform a cookie forging attack to remove the requirements for the user to browse specific subdomains while being monitored.

Personal information. The Y and T cookies set for `yahoo.com` allow the attacker to obtain the user's first name. The full last name and email address can also be obtained, as we explain below.

Yahoo Mail. To facilitate sharing posts with friends, articles in Yahoo contain an "Email to friends" button, which presents a popup window in which the adversary can add an arbitrary message, as shown in Figure 4. Furthermore, the `Sender` field has auto-complete functionality, which allows us to obtain the victim's complete contact list. These features combined can be leveraged for deploying effective phishing or spam campaigns. The contacts' emails can be used for acquiring information about those users from other services and deploying personalized spam campaigns [33]. The widget also contains the user's full name and email address. Extracting the contacts requires all three cookies set for the main domain, while sending the email requires them for the `news.yahoo.com` or

the `finance.yahoo.com` subdomain depending on which section the article is located in.

If the user hovers over or clicks on the mail notification button, the attacker can also access the incoming mail widget, which reveals the `Sender` and partial `Subject` (up to 21 characters) of the 8 most recent incoming emails. This is due to a cookie being attributed an "authenticated" status. This lasts approximately one hour, after which it cannot access the widget. If at any point the user accesses the notification button again, the hijacked cookie is re-authorized.

Yahoo Search. Having acquired the main domain and search subdomain Y and T cookies, the adversary can gain access to the victim's complete search history. Apart from viewing the searched terms, these cookies allow editing the history and removing previous searches. However, Yahoo explicitly states that even if past searches are deleted, user search data is still logged. This enables *stealthy* pollution attacks; after issuing search queries for influencing the personalization profile of the user, the adversary can then delete all issued searches and remove traces of the attack.

Yahoo Answers. One of the many services offered by Yahoo, is a popular "question and answer" site, where users can ask any type of question, and other members of the community can provide answers (albeit sometimes with questionable quality [34]). Users posting questions or answers, may choose to remain anonymous for a given question, especially if the topic is considered sensitive [35]. Upon auditing Yahoo, we found that the victim's HTTP cookie allows partial control over the account; the adversary is able to ask or answer questions (either eponymously or anonymously), and also to view and edit previous questions and answers posted by the victim. Thus, the adversary can effectively "deanonymize" posts and obtain potentially sensitive information about the victim, which was posted under the assumption of anonymity. The adversary can also post comments as the victim in the comment section of news articles. This requires the Y, T cookies for the `yahoo.com` domain and the `answers.yahoo.com` subdomain.

Yahoo Finance. Another service offered by Yahoo is related to financial news and functionality, and also offers tools for users to manage their personal finances. This includes creating portfolios with their stock information etc. The Y and T cookies for the main domain and finance subdomain allow the attacker to view and edit the victim's portfolio. If the victim visits the finance page, the corresponding cookies are exposed.

Cookie forging. Different cookie combinations provide access to specific user information or account functionality and, depending on the subdomain on which the information is hosted, the respective cookies for those domains are required. However, we can use a cookie acquired from one (sub)domain to craft the same cookie for a different subdomain and gain access to the specific information or functionality. For example, if the user only visits the main Yahoo homepage during the monitoring period, the attacker will obtain the Y, F, T cookies for `yahoo.com`. The attacker can then "forge" those cookies for the `search.yahoo.com` subdomain using the corresponding `value` attributes of the hijacked cookies and subsequently gain access to the user's search history.

D. Baidu

Baidu is the leading search engine in the Chinese language and among the top 5 sites according to Alexa. To create an account in Baidu, the user is required to register either with a Chinese mobile phone number or just provide an email address. The majority of pages in Baidu are served over an unencrypted connection. As with the other search engines we tested, the HTTP cookie can expose significant private information to attackers. Apart from the profile picture and username, the user's email address is also revealed. Furthermore, the user's entire search history can be retrieved, and pollution attacks are feasible. Finally, Baidu Maps allows users to save locations, similar to Bing Maps, and all saved locations can be obtained through the hijacked HTTP cookie.

E. E-commerce Websites

Amazon. The homepage follows the common approach of redirecting to HTTPS if connected to over HTTP. However, product pages are served over HTTP and, as a result, users' cookies will be exposed during their browsing sessions.

Personal Information. The adversary can obtain the information used by the victim for logging in; this includes the victim's username, email address and/or cell phone number. Furthermore, when proceeding to checkout items in the cart, Amazon also reveals the user's full name and city (used for shipping). Viewing and changing the user's profile picture is also permitted. Amazon also allows users to post their reviews under a pseudonym, which is not connected to the user's name. However, the adversary can view the user's reviews (which may include sensitive items), thus, breaking the pseudonymous nature of those reviews. Previous work has demonstrated the privacy risks of recommender systems and experiments in Amazon indicated that sensitive purchases can be inferred from the user's review history [36].

Account History. The user's HTTP cookie is sufficient for accessing private information regarding previous actions. Specifically, the adversary can obtain information regarding recently viewed items, and recommendations that are based on the user's browsing and purchase history. The wish-lists where the user has added items of interest are also accessible. Furthermore, the adversary can obtain information regarding previously purchased items either through the recommendation page or through product pages (which depict date of purchase). In an extensive study on privacy-related aspects of online purchasing behavior [37], users rated the creation of a detailed profile from their purchase history and other personal information as one of the most troubling scenarios.

Shopping Cart. The user's cart is also accessible, and the adversary can see the items currently in the user's cart. Additionally, the cart can be modified, and existing items can be removed, and other items can be added.

Vendor-assisted spam. We also found that the cookie exposes functionality that can be leveraged for deploying spam

campaigns to promote specific items that are presented as "endorsed" by the victim. The widget has an auto-complete feature that reveals the contacts that the user has emailed in the past. The attacker can either send emails about a specific item or a wish-list, and can add text in the email's body. URLs can be included; while the email is sent as simple text, email providers such as Gmail render it as a clickable link. Since the emails are actually sent by Amazon (`no-reply@amazon.com`), they are most likely to pass any spam detection heuristics. Furthermore, the `From` field, contains the victim's username, further strengthening the personalized nature of the spam/phishing email.

Extortion scams. Previous work has revealed how scammers extorted money from users through *One Click Fraud* scams by threatening to reveal "embarrassing" details about the users' online activities [38]. In a similar vein, the attacker can employ two different scam scenarios. In the first case, if the attacker identifies potentially embarrassing item(s) in the user's viewing or purchase history, she can send an email to the user disclosing knowledge about the item(s), and other personal information obtained about the user, and request money to not share that information with the user's contacts (even if no contact information has been collected). In the second scenario, the attacker can send an email blackmailing the user to pay money, otherwise she will send an email to the victim's friends and family with information about his cart that is full of embarrassing items. Subsequently, the attacker will add such items to the user's cart or wishlist, and send the corresponding email through Amazon to the victim's own email address as proof of her capabilities.

Walmart. Apart from the information exposed in the website, the cookie's `value` attribute contains 34 fields of information about the user and his account (see Appendix A).

F. News Media

Information acquired from media outlets can reveal characteristics and traits of the user (e.g., political inclination), and demographic information [39]. We audited the websites of several of the most popular print or broadcast news organizations (see Appendix A).

G. Indirect Information Exposure - Ad Networks

We explore the impact of hijacking ad network cookies. Online ads account for a significant portion of website real estate, and their ubiquitous nature has been discussed extensively in the context of user tracking (e.g., [40], [41]). Here we focus on Doubleclick, which is owned by Google, as it is the most prevalent advertising network with a presence on 80% of the websites that provide advertisements [42]. As opposed to most of the previous services where the user had to explicitly visit the website[3], the cookies of an ad network can be exposed by visiting any of a large number of websites that display ads from the respective network. While the symbiotic nature of service providers and data aggregators is complicated, the ads presented while browsing with stolen user cookies from ad networks can be used to infer sensitive information.

An interesting aspect of hijacking ad-network cookies is that they result in *side-channel information leakage*. We describe two scenarios which leak different types of information.

Attack scenario 1. Consider a scenario where user `U` has an account on the social networking service `X`, and has disclosed various pieces of personal information in the profile. Let us also consider that `U` is knowledgeable and has correctly set privacy settings to restrict public visibility of that information, and `X` has gone to great lengths to protect users from information leakage and also enforces ubiquitous encryption across the website, including connections to third parties (e.g., when fetching ads). However, website `X` offers personalized advertising and ad network `A` has obtained personal information of `U` by setting different selection criteria over time and identifying `U` across websites through an HTTP cookie. Now lets consider that while being monitored by the attacker, `U` browses a completely unrelated website `Y` which serves ads by ad network `A` and does not enforce encrypted connections. Even though `U` does not have an account on `Y`, the browser sends the HTTP cookie for `A`, which can be used to identify `U` and return an ad that is tailored to match information disclosed by `U` in the original website `X`. The attacker can hijack the exposed HTTP cookie for `A`, and receive ads tailored for `U`. Based on the content of these ads, the attacker can infer personal information of `U`.

Attack scenario 2. User `U` is browsing through an e-commerce site `E`, which uses the ad network `A` to advertise its products in other websites. `U` searches for items that belong to a specific category `C`, and after the site returns a list of relevant products, `U` clicks on a link and views the page of product `P`. A short time later, the attacker visits an unrelated website that is known to show various ads, and appends `U`'s stolen HTTP cookie for the ad network `A`. The attacker is then presented with several ads relevant to `U`'s browsing history. Some are more generic and expose information about `U`'s gender, while others explicitly refer to category `C` and even depict the specific item `P`.

Information leakage. We conducted a small number of manual experiments for identifying cases of personal information being leaked by Doubleclick. Previous work has shown that ads presented to users may be personalized based on the user's profile characteristics [43], associated to sensitive topics [44], [45] (e.g., substance abuse, health issues, sexual inclination), and that advertisers can even obtain private user information not explicitly provided by the service [46].

Here we describe one of our experiments for scenario 2. We browsed maternity clothes on a popular e-commerce website, and visited the page of a few returned products. We, then browsed other sites from a different machine connected to a different subnet, and appended the Doubleclick HTTP cookie from the previous browsing session. We were presented with ads from the e-commerce website advertising women's clothing. Several ads even advertised a specific maternity product whose page we had visited (see screenshots in Appendix A).

[3]We found a popular e-commerce homepage that issues a Google search request over HTTP, exposing the user's cookies.

TABLE IV
COOKIE EXPOSURE BY POPULAR BROWSER EXTENSIONS AND APPS.

Name	Type	Browser	#	Cookie leaked
Google Maps	app	Chrome	N/A	✓
Google Search	app	Chrome	N/A	✓
Google News	app	Chrome	1.0M	✓
Amazon Assistant	extension	Chrome	1.1M	✓
Bing Rewards	extension	Chrome	74K	✓
eBay for Chrome	extension	Chrome	325K	✓
Google Dictionary	extension	Chrome	2.7M	✓
Google Hangouts	extension	Chrome	6.4M	✗
Google Image Search	extension	Chrome	1.0M	✗
Google Mail Checker	extension	Chrome	4.2M	✗
Google Translate	extension	Chrome	5.5M	✗
Yahoo Mail Notification	extension	Chrome	1.2M	✗
Amazon	default search bar	Firefox	N/A	✓
Bing	default search bar	Firefox	N/A	✗
Ebay	default search bar	Firefox	N/A	✓
Google	default search bar	Firefox	N/A	✗
Yahoo	default search bar	Firefox	N/A	✗
Amazon 1Button	extension	Firefox	157K	✓
Bing Search	extension (unofficial)	Firefox	28K	✓
eBay Sidebar	extension	Firefox	36K	✓
Google Image Search	extension	Firefox	48K	✓
Google Translator	extension (unofficial)	Firefox	794K	✓
Yahoo Toolbar	extension	Firefox	31K	✓

TABLE V
COOKIE EXPOSURE BY OFFICIAL MOBILE APPS.

Application	Platform	Version	#	Cookie leaked
Amazon	iOS	5.3.2	N/A	✗
Amazon	iOS	5.2.1	N/A	✓
Amazon	Android	28.10.15	10-50M	✗
Bing Search	iOS	5.7	N/A	✓
Bing Search	Android	5.5.25151078	1-5M	✓
Spotlight (Bing)	iOS	iOS9.1	N/A	conditionally
Siri (Bing)	iOS	iOS9.1	N/A	✗
Ebay	iOS	4.1.0	N/A	conditionally
Ebay	Android	4.1.0.22	100-500M	conditionally
Google	iOS	9.0	N/A	✗
Google	Android	5.4.28.19	1B+	✗
Gmail	iOS	4.1	N/A	✗
Gmail	Android	5.6.103338659	1-5B	✗
Google Search Bar	Android	5.4.28.19	N/A	✗
Yahoo Mail	iOS	4.0.0	N/A	conditionally
Yahoo Mail	Android	4.9.2	100-500M	✗
Yahoo News	iOS	6.3.0	N/A	✓
Yahoo News	Android	18.10.15	10-50M	✗
Yahoo Search	iOS	4.0.2	N/A	✗
Yahoo Search	Android	4.0.2	1-5M	✗
Yahoo Sports	iOS	5.7.4	N/A	✓
Yahoo Sports	Android	5.6.3	5-10M	✗

Depending on the time lapsed between the user browsing the e-commerce site and the attacker browsing with hijacked cookies, there is a decrease in the frequency of ads that contain the viewed product. However, we found that even after several hours we received ads that continued to promote the exact product, and women's clothing ads even after several days.

IV. COLLATERAL COOKIE EXPOSURE

In this section we explore other means by which a user's HTTP cookies may be exposed.

A. Browser Components

According to a manifest file analysis of over 30K Chrome extensions [47], a higher number of extensions requested permission for connecting to Google over HTTP compared to HTTPS. The same was true for wildcarded (`http://*/*`) permission requests. This indicates that a considerable number of extensions may be weakening security by connecting over unencrypted connections to websites that also support encrypted connections. To that end, we explore whether browser components expose users to cookie hijacking attacks.

We analyze a selection of the most popular browser components, for Chrome and Firefox, that have been released by major vendors we have audited. Our aim is not to conduct an exhaustive evaluation, but to obtain an understanding of the implementation practices for browser components and assert whether they also suffer from a limited use of encryption. While we experiment with a relatively small number of components, we consider any discovered exposure indicative of general practices, as official extensions from major vendors are likely to adhere to certain quality standards. As Google has discontinued the development of extensions for Firefox, we cannot do a direct cross-browser comparison for most of its components.

Table IV lists the web components we have evaluated, their reported number of downloads if available, and if they leak the cookies required for our hijacking attacks. Our experiments yield a number of surprising findings. The 3 Chrome apps released by Google we tested expose the HTTP cookies, while their extensions present mixed results with 4 out of 9 leaking the cookie. As one of those is Google Dictionary, with over 2.7 million downloads, a significant number of Chrome users is vulnerable to considerable risk. Every Firefox extension we tested, along with two of the default search bars, actually expose the required HTTP cookies over unencrypted connections. Interestingly, Google's Search by Image extension is secure for Chrome but not for Firefox. As there is no official Bing app for Firefox, we test the most popular one, and we also audit a popular unofficial Google translator extension with over 794K users, both of which turn out to be vulnerable. Overall, these findings highlight the privacy threats that millions of users face due to browser components.

B. Mobile Devices

Mobile devices have become ubiquitous, and account for a large part of the time users spend online. Due to the quota restrictions in mobile data plans, users frequently connect to public WiFi access points. According to Cisco [48], an estimated 45% of mobile traffic is "offloaded" to WiFi connections. While this is not restricted to public WiFi networks, it is indicative of user behavior, with a recent survey reporting that 72% of the participants connect to public WiFi [49]. To explore the feasibility of our HTTP cookie hijacking attacks against users on mobile devices, we audited the official iOS and Android apps for the most popular services that we found to expose private information and account functionality.

The overview of our results is shown in Table V. It is noteworthy that Bing differentiates mobile cookies and, as a

TABLE VI
HTTPONLY ATTRIBUTE OF THE COOKIES REQUIRED FOR HIJACKING ACCOUNTS, AND THE FEASIBILITY OF CONDUCTING THE ATTACKS WITH THE COOKIES THAT ARE OBTAINABLE REMOTELY.

Site	HttpOnly	non—HttpOnly	XSS Hijacking
Amazon	—	x-main	✓
Bing	—	_U, WLS	✓
Baidu	—	BDUSS	✓
CNN	—	CNNid, authid	✓
Doubleclick	—	id	✓
Ebay	—	cid, nonsession	✓
Google	HSID	SID	✗
Guardian	—	GU_U	✓
HuffingtonPost	huffpost_s	huffpost_user huffpost_user_id last_login_username	✗
MSN	MSNRPSAuth	—	✗
New York Times	—	NYT-S	✗
Target	—	WC_PERSISTENT guestDisplayName UserLocation	✓
Walmart	—	customer, CID	✓
Yahoo	F	T, Y	partial
Youtube	VISITOR_INFO1_LIVE	—	✗

result, hijacked mobile cookies expose the search and click history that has been conducted only over the mobile device; the remaining personal information presented in Section III-B is still obtainable. Spotlight, the system-wide search feature of iOS, is also powered by Bing. When the user issues a search query, Spotlight connects over HTTPS to Apple servers. However, the search results contain a "Show more in Bing" button and, if clicked, will open the browser showing the search results and leak the user's HTTP Bing cookie. For Siri, the voice-guided assistant, the Bing results are opened in the browser over HTTPS, preventing cookie hijacking. Once again Yahoo follows poor security practices as 3 out of 4 iOS apps leak the user's cookies. As expected both versions of Gmail protect the cookies, while iOS Amazon apps prior to version 5.3.2 expose the cookie. Furthermore, both Amazon iOS apps contain cookies that reveal information about the user's device and mobile carrier (details in Appendix A). For both platforms, the Ebay app will expose the cookies under certain conditions. First, Ebay sellers are allowed to customize their item pages and often add links to other items they are selling; if the seller has added an HTTP Ebay link to those items, the cookie will be exposed if a link is clicked by the user. Empirically we found that that these HTTP links are common. The other scenario is if the user clicks on the "Customer Support" menu.

C. Active Attacks

Remote hijacking. We analyze the cookies of each service in depth, and identify what information the attacker can obtain remotely, e.g., by stealing the user's cookies through an XSS attack. While the `HttpOnly` attribute can prevent attackers from remotely obtaining cookies through browser scripts, our findings reveal limited adoption, indicating that the situation has not improved in recent years [19].

As can be seen in Table VI, websites with multiple cookies never set the attribute for all. Most websites set the attribute for some cookies, but allow other cookies to be accessed by scripts. Furthermore, Amazon and Target have the `HttpOnly` attribute set to `false` for all their cookies. Surprisingly, 66.6% of the websites that are vulnerable to our cookie hijacking attacks, also expose users to remote cookie hijacking. For Yahoo, while we cannot access all the information and account functionality described previously, several instances remain possible (e.g., search history, username etc.) Even though the attack cannot be done for Huffington Post as the `huffpost_s` cookie has the flag set, the remaining cookies still expose the user's username and email address.

V. DEANONYMIZATION RISK FOR TOR USERS

In this section, we investigate if more privacy-conscious users are protected against our presented cookie hijacking attacks. Specifically, we explore how users employing the Tor bundle (Tor Browser with pre-installed extensions) can be deanonymized by adversaries. In this case, we consider a variation of the threat model from the previous sections; the adversary monitors Tor exit nodes instead of public wireless access points. We do not consider content-injection or active attacks, such as SSL stripping [10] for weakening protection.

A. Experimental Analysis

We repeat our experiments from Section III on a subset of the audited websites. To understand the protection that privacy-conscious users can obtain, we experiment with three different client-side setups. In the first case, we simulate a user that uses Firefox and connects over the Tor network [50] for increased protection. The second user is more well-informed and has installed the HTTPS Everywhere browser extension for better protection. The final case is of a user that has selected the default configuration of the Tor bundle, which includes the Tor Browser (a modified Firefox) and other extensions (including HTTPS Everywhere).

HTTPS Everywhere. This browser extension [51] is the result of collaboration between EFF and the Tor Project. The extension contains per domain rule-sets for a large number of domains[4], which instruct the re-writing of links within pages to force encrypted connections. However, websites may contain pages or subdomains whose functionality breaks over HTTPS. For those cases, each website's rule-set will contain exceptions for identifying links pointing to problematic pages, which are not overwritten and are connected to over HTTP. The rule-sets are created and maintained by the community, which requires a significant amount of manual effort, and can result in incomplete rules. Certain sites (e.g. doubleclick.net, ebay.com) are turned off by default, as their functionality breaks if turned

[4]https://www.eff.org/https-everywhere/atlas/

TABLE VII
EXAMPLES OF URLS AND SUBDOMAINS OF POPULAR SERVICES THAT EXPOSE TOR USERS' COOKIES FOR DIFFERENT BROWSER CONFIGURATIONS.

Browser Configuration	Google		Bing		Yahoo		Amazon	Ebay
	domain	subdomains	domain	subdomains	domain	subdomains	domain	domain
Firefox	✓	✓	✓	✓	✓	✓	✓	✓
Firefox + HTTPS Everywhere	`error404` page `google.com/service*`	`translate.google.com` `picasa.google.com`	✗	`m2.cn.bing.com` `blogs.bing.com`	✗	✓	✓	✓
Tor Bundle	`error404` page `google.com/service*`	`translate.google.com` `picasa.google.com`	✗	`m2.cn.bing.com` `blogs.bing.com`	✗	✓	✓	—

*`service`: {mail, maps, drive, docs, talk, ...}

TABLE VIII
ACCOUNTS FROM OUR PUBLIC WIRELESS TRACE (SECTION II-C) THAT REMAIN EXPOSED EVEN WITH HTTPS EVERYWHERE INSTALLED.

Services	Exposed Accounts	Reduction
Google	31,729	53.12%
Yahoo	5,320	43.55%
Baidu	4,858	4.63%
Bing	378	38.03%
Amazon	22,040	5.68%
Ebay	1,685	0%
Target	46	0%
Walmart	97	23.62%
NYTimes	15,190	0%
Guardian	343	0.29%
Huffington	42	0%
MSN	927	39.25%
Doubleclick	124,352	0%
Youtube	264	99.21%
Total	207,271	26.62%

on. Therefore, user accounts are likely to be exposed even with this extension in place, since a single HTTP request is enough.

Table VII contains the results of our experiments. In the first case where the user browses through Firefox and only employs Tor, the user remains vulnerable to the full extent of the attacks described in Section III (denoted by ✓) . This is expected as Tor is not designed to prevent this class of attacks. In the second and third cases where HTTPS Everywhere is also installed, we discover a varying degree of effectiveness.

For Google the attack surface is significantly reduced, as users visiting the main domain through the address bar are protected. As this is a common usage scenario (if not the most common), a significant number of users may be protected in practice. However, the extension's rule-set does not cover several cases, such as when the user visits one of Google's services through the address bar (e.g., by typing `google.com/maps`), or when receiving Google's `Error 404` page. For Bing the attack surface is also significantly reduced, but users can still be exposed, e.g., by a subdomain that hosts the search engine but does not work over HTTPS. For cases such as Amazon and Yahoo, the protection offered by the extension is ineffective against our attacks, as browsing the website will expose the required cookies. In Amazon any product page will reveal the required cookie, while in Yahoo we always receive the cookies required from the links on the homepage redirecting through `hsrd.yahoo.com`. While for Ebay our attacks remain effective when we use Firefox, we could not complete the experiment with the Tor browser as any login attempts simply redirect to the login page without any error message (probably due to incompatibility with an extension). For the cases where the attack is still feasible, Table VII does not present an exhaustive list of vulnerable points, but an indicative selection of those we have experimented with. In practice, any URL that is handled by the exceptions in each website's rule-set can potentially expose the HTTP cookies.

Quantifying impact. To simulate the potential impact of HTTPS Everywhere, we use the network trace collected from our campus' public WiFi, and calculate the number of accounts that would remain exposed due to URLs not handled by HTTPS Everywhere rule-sets (version 5.1.0). We found that over 77.57% of all the collected HTTP traffic would remain over HTTP even if HTTPS Everywhere was installed in every users' browser. Due to those connections, 207,271 accounts remain exposed to our cookie hijacking attacks. Table VIII breaks down the numbers per targeted service. The largest impact is seen in Youtube where less than 1% of the users remain exposed while Ebay, Doubleclick and numerous news sites are not impacted at all. Surprisingly, even though Google's main page is protected, over 46% of the users remain exposed when visiting a Google service. For the remaining search engines, the impact has a varying degree, with over 95% of the Baidu users remaining susceptible to cookie hijacking.

While the Tor bundle offers significant protection against a variety of attacks, its effectiveness in mitigating cookie hijacking attacks varies greatly depending on each website's implementation. Even with all protection mechanisms enabled, users still face the risk of deanonymization when visiting popular sites. Therefore, the threat they face greatly depends on their browsing behavior, which we try to evaluate next.

B. Evaluating Potential Risk

We want to explore whether privacy-conscious users actually visit these major websites over the Tor network, or if they avoid them due to the lack of ubiquitous encryption.

Ethics. Again, we obtained IRB approval for our experiments. However, due to our ethical considerations for the Tor users (as they are not members of our university nor connecting to our public wireless network), we do not replicate the data collection we followed in our experiment

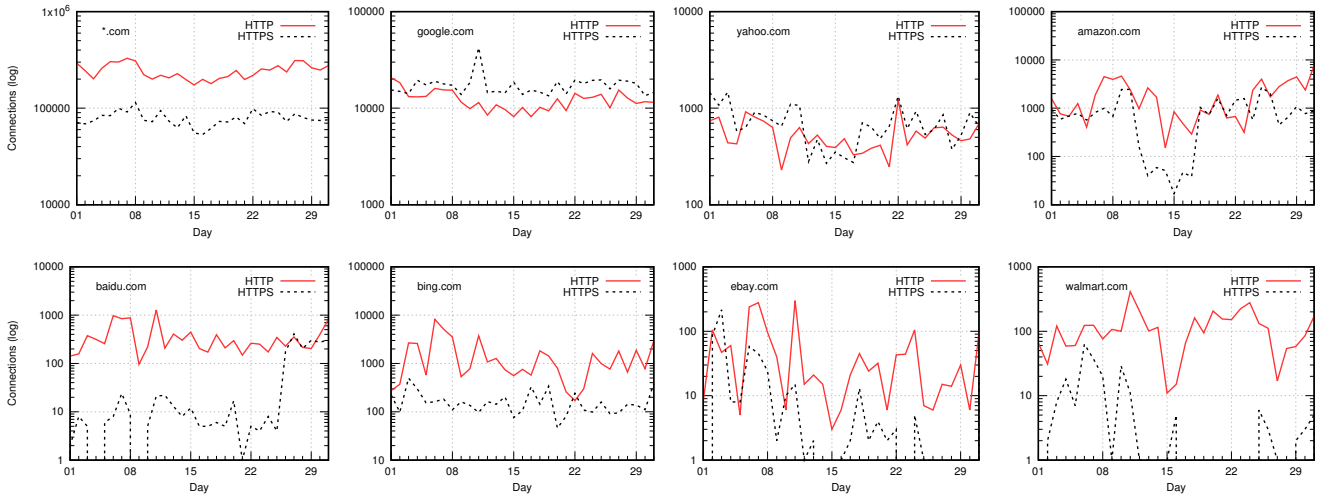


Fig. 5. Number of encrypted and unencrypted connections per day, as seen from a freshly-deployed Tor exit node.

from Section II-C. We opt for a coarse-grained non-invasive measurement and only count the total connections towards the websites we audited in Section III, using the port number to differentiate between HTTP and HTTPS. *We do not log other information, inspect any part of the content, or attempt to deanonymize any users.* Furthermore, *all data was deleted* after calculating the number of connections. Since we do not look at the name of the cookies sent in the HTTP connections, we cannot accurately estimate the number of users that are susceptible to cookie hijacking attacks. Our goal is to obtain a rough approximation of the number and respective ratio of encrypted and unencrypted connections to these popular websites. Based on the measurements from our university's wireless trace, we can deduce the potential extent of the deanonymization risk that Tor users face. We consider this an acceptable risk-benefit tradeoff, as the bulk statistics we collect do not endanger users in any way, and we can inform the Tor community of a potentially significant threat they might already be facing. This will allow them to seek countermeasures for protecting their users.

Tor exit node. The number of outgoing connections were measured over 1 month, on a fresh exit node with a default reduced exit policy[5] and bandwidth limited to 300 KB/s.

Measurements. Figure 5 presents the number of total connections and broken down for some services. The number of connections over HTTP account for 75.4% of all the connections we saw, with an average of 10,152 HTTP and 3,300 HTTPS connections per hour. While non-HTTP traffic may be contained within the total connections, we do not distinguish it as that would require a more invasive approach. For most of the services, the unencrypted connections completely dominate the outgoing traffic to the respective domains. On the other hand, for Google we observe an average of 508 HTTP connections per hour as opposed to 705 HTTPS connections. Similarly we logged 23 unencrypted connections to Yahoo per hour and 36 encrypted connections. We do not consider the Doubleclick side channel leakage attack for Tor, as the double key session cookies employed by the Tor browser affect third party cookies and their ability to track users across domains.

Susceptible population. We see that there is a significant amount of HTTP traffic exiting Tor and connecting to popular websites that expose a vast collection of private user information. While the ratio of unencrypted connections is even higher than that of our university's network, possibly fewer users will be logged in when using Tor. More experienced users may be aware of the shortcomings of this mechanism and avoid the pages and subdomains that are not protected when connecting over untrusted connections. Nonetheless, we expect that many users will exhibit normal browsing patterns, thus, exposing their cookies to attackers. Furthermore, even though we can not know how many of the users are indeed logged in and susceptible to cookie hijacking (that would require looking at the cookie names), for some websites observing encrypted connections is an almost definitive sign that we are also observing HTTP traffic of logged in users; due to functionality breaking and the corresponding exceptions in the HTTPS Everywhere rule-sets, HTTPS traffic for Amazon and Baidu signifies account-related functionality that requires users to be logged in (e.g., Amazon checkout) and is accompanied by HTTP traffic (Amazon products pages). Thus, we believe that a considerable number of Tor users may be facing the risk of deanonymization through hijacked cookies.

User bias. As this is a newly deployed exit node, the population of users connecting to it may be biased towards inexperienced users, as more privacy-conscious ones may avoid exiting from such nodes. Thus, our observed ratio of encrypted connections or the websites users connect to, may present differences to other exit nodes. Nonetheless, adversaries could already own exit nodes with long uptimes, or be able to monitor the outgoing traffic from legitimate exit nodes, which is a common adversarial model for Tor-related research [50]. Thus, we believe this to be a credible and severe threat to Tor users that want to maintain their anonymity while browsing (popular) websites.

[5]https://trac.torproject.org/projects/tor/wiki/doc/ReducedExitPolicy

VI. Countermeasures and Discussion

Our work focuses on highlighting the privacy ramifications of HTTP cookie hijacking attacks, and we have demonstrated the gravity and pervasiveness of sensitive information being exposed by high-profile services. We discuss potential causes for the current vulnerable state of major websites, and how existing security mechanisms fare in practice. While the defenses for preventing these attacks are known and, seemingly, straightforward, our experiments demonstrate that even the most popular and resourceful websites succumb to design and implementation flaws.

Partial encryption and personalization. Due to the complexity in implementing large-scale web services, and also defining precise access privileges for multiple (inter-dependent) cookies for different subdomains, web developers are prone to errors such as the incomplete separation of access control for unauthenticated cookies. In turn, this allows passive eavesdroppers that hijack HTTP cookies to obtain sensitive information. While we tested certain websites where partial encryption did not result in privacy leakage, none of those services offered a personalized version of the service to HTTP cookies. This indicates the conflicting nature of offering personalization while aiming to maintain ease-of-use by not requiring re-authentication. As such, we argue that any service that supports user accounts and personalizes the experience, should enforce ubiquitous encryption, which would mitigate the privacy threats we have explored.

Cookie Flags. By setting the `Secure` cookie flag to `True`, websites can ensure the confidentiality of a cookie by instructing the browser to only send over encrypted connections. However, while this can prevent passive eavesdroppers from acquiring the cookies, it is known that active attackers can overwrite secure cookies from an insecure channel [52]. Therefore, the `Secure` flag as a stand-alone measure cannot fully protect users. It should be used in combination with fully deployed HSTS and support for ubiquitous encryption. Furthermore, the `HttpOnly` flag should be set to prevent remote cookie hijacking.

Security mechanisms. We have evaluated the impact of browser-supported security mechanisms on the feasibility of our attacks. Here we discuss our findings about the protection these mechanisms offer and their shortcomings.

HSTS. Recent work presented an extensive study on HSTS and discussed the pitfalls of deploying it in practice, reporting that many developers fail to implement it correctly [11]. In our work, we focus on the fact that even if implemented correctly, partial deployment nullifies the protection it offers and users remain exposed. This is particularly apparent when the main landing page of a website does not enforce HSTS. Even if users are subsequently redirected to HTTPS (as is the case with Google), the HTTP cookies are exposed during the initial connection. As it is common for users to directly visit popular websites by typing their name in the address bar, which is facilitated by auto-completion functionality, this practice can expose a large number of users to cookie hijacking attacks. Taking into consideration our findings regarding the amount of personal information and account functionality that unauthenticated cookies can access, this is a significant privacy risk that users face. The need for full HSTS deployment has also been argued for by others [11], [52].

HTTPS Everywhere. Through our experimentation, we found that this browser extension improves user security by minimizing the attack surface, and can prevent risks due to partial (or non-existent) deployment of HSTS. However, it is crucial to note that, even with this extension in place, users are not entirely protected. As site functionality can break if the server does not support HTTPS for a specific subdomain or page, HTTPS Everywhere relies on rule-sets that contain exceptions for these cases. As such, while certain websites might be significantly covered, other cases still contain a considerable number of unprotected pages. If users click on the extension's notification icon, a menu shows information regarding the current page and if content has been fetched over non-encrypted connections. However, users are notoriously good at ignoring warnings, and their design can significantly affect user actions [53]. The menu contains an option to block such connections. While this can break the browsing experience, it may be a prudent choice for users that consider their privacy of paramount importance. This could apply to users that rely on systems such as Tor for maintaining their anonymity, who can be deanonymized as we discuss in Section V. Nonetheless, this is not the default option, and if the user visits such a website before enabling the option, the HTTP cookies will be exposed. Thus, enabling this option by default and allowing users to opt-out is a safer approach.

VPN. End users should also consider the use of VPN technology when connecting to untrusted public wireless networks, as it reduces the threat of the user's traffic being sniffed [54].

VII. Ethics and disclosure

To ensure the ethical nature of our research, we provided a detailed description of our data collection and analysis process to Columbia University's IRB, and obtained approval for both our experiments with the public wireless network and the Tor network. Furthermore, all captured data was destroyed after the end of our evaluation measurements.

Disclosing attacks against popular services raises ethical issues as, one might argue, adversaries may have previously lacked the know-how to conduct these attacks. However the practicality of cookie hijacking suggests that such attacks could soon happen in the wild (if not happening already). To that end, we have already contacted all the audited websites to disclose our findings in detail. We have also contacted Tor developers to inform them of the deanonymization threat users face. We believe that by shedding light on this significant privacy threat, we can incentivize services to streamline support for ubiquitous encryption. Furthermore, we must alert users of the privacy risks they face when connecting to public wireless networks or browsing through Tor, and educate them on the extent of protection offered by existing mechanisms.

VIII. Related Work

Hijacking and other cookie-related issues. Zheng et al. [52] recently presented an empirical study on the feasibility of cookie injection attacks. While cookies have the `Secure` flag that can prevent browsers from sending them over unencrypted connections, there is no provision to ensure that such cookies are also set only over HTTPS connections. As a result, during an HTTP connection to a domain, a man-in-the-middle attacker can *inject* cookies that will be appended in future HTTPS connections to that specific domain. In their real-world assessment of this attack, the authors explored how cookie injection could enable attacks such as account and (sub-)session hijacking, and cart manipulation in e-commerce sites. They also identify how browser-specific handling of cookies can enable attacks. Bortz et al. [55] had previously described cookie injection attacks, and proposed origin cookies for achieving session integrity in web applications.

Wang et al. [56] identified flaws in popular ID providers of single-sign-on services that allowed attackers to log into services as the users. Karlof et al. [57] introduced pharming attacks that relied on DNS hijacking and allowed attackers to hijack user sessions. Lekies et al. [58] leveraged the exemption of remote scripts included through the HTML `script` tag from the Same-Origin policy for leaking personal information and, in some cases, hijacking sessions. Barth et al. [59] introduced the *login CSRF* attack where the user is logged into a legitimate service as the attacker, which can result in the exposure of sensitive user information.

Numerous approaches have been proposed for preventing session hijacking [60]–[62]. Jackson and Barth presented ForceHTTPS [63], a browser extension for enforcing HTTPS connections. This was reformed and standardized as HSTS [12]. Kranch and Bonneau [11] performed an extensive study on the adoption of HSTS and certificate pinning in the wild. They reported a lack of understanding by web developers on the proper use of these mechanisms, as they often use them in illogical or invalid ways. Selvi [64] demonstrated scenarios where an attacker could bypass HSTS protection. Bhargavan et al. [65] also showed how the HSTS header could be partially truncated, resulting in the expiration of the HSTS entry within seconds. Singh et al. [3] studied the incoherencies of web access control and showed that user actions and resources could be improperly exposed to web applications.

Mayer and Mitchel [40] discussed the policy and technology issues of third-party web tracking. Roesner et al. [41] studied the behavior of web trackers and found an extensive set of trackers. They also explored the impact of browser mechanisms, such as cookie blocking and DoNotTrack, and found that preventing web-tracking from popular social network widgets also broke their functionality. Sivakorn et al. [66], demonstrated how HTTP cookies could be used for influencing Google's advanced risk analysis system and bypassing reCAPTCHA challenges. Attackers could use hijacked cookies in a similar fashion, which can bypass even more stringent safeguards that require extensive browsing history.

Privacy leakage. Krishnamurthy and Willis explored online social networks and the leakage of users' personally identifiable information (PII) in HTTP headers, and described how third-party servers can exploit the PII leakage to link it with user actions across different domains [5]. In follow-up work, the authors focused on the privacy leakage in social networks that leveraged the GPS capabilities of mobile devices to offer location-based functionality [6], and explored how pieces of information collected from different social networks could be combined for further compromising user privacy. Englehardt et al. [7] explored the feasibility of conducting mass surveillance by monitoring unencrypted traffic and inspecting third-party tracking cookies. They also identified cases of PII being exposed in unencrypted traffic, which can be leveraged for improving the clustering of user traffic and linking different requests to the same user. While their work focuses on a different attack scenario, their results also highlight the threat of unencrypted connections. Liu et al. [8] developed a novel method for detecting PII being transmitted in network traffic, without prior knowledge of the fields and form of the information transmitted by each service. Due to the very small fraction of fields that actually contain PII, the authors argue that looking for fields with values that are unique to a user results in very high false positives and false negatives. Thus, mass surveillance attacks will have to employ more advanced techniques. The evaluation of the proposed approach on a large-scale trace presented a false positive rate of 13.6%.

These approaches, however, have a limited viewpoint and can only detect information sent in clear text during the monitoring period. There exist multiple common scenarios where exposed personal information will not be detected: (i) websites are highly dynamic and content may be fetched in an obfuscated form and constructed at runtime on the client side, (ii) sensitive content may always be fetched over encrypted connections, even though HTTP cookies may (erroneously) have sufficient access privileges, (iii) certain pieces of information are only exposed after specific user actions, which may not occur during the monitoring period. Furthermore, cookie hijacking attacks can also access protected account functionality in certain cases due to imprecise access control. Overall, our goal is to explore the prevalence and criticality of private information and account functionality being accessible to HTTP cookies, and understanding how varying components of the complicated ecosystem (from browser security mechanisms to mobile apps) affect the attack surface and feasibility of hijacking. Furthermore, as the authors state [7], using the Tor bundle likely defeats their attack scenario. On the other hand, we demonstrate that while the Tor bundle reduces the attack surface, cookie hijacking remains feasible.

Risks of personalization. The personal information leakage we identify in our attacks is a direct result of websites offering a personalized experience to users. Castelluccia et al. [4] highlighted the problem of privacy leakage that can occur when personalized functionality is accessible to HTTP cookies. The authors demonstrated how adversaries could reconstruct a user's Google search history by exploiting the personalized

suggestions. Korolova presented novel attacks that use targeted ads for obtaining private user information [46]. Toch et al. [67] analyzed the privacy risks that emerge from popular approaches to personalizing services.

Encrypted connections. The privacy threats we study are also the result of websites not enforcing encryption across all pages and subdomains. Previous work has shown the risks of supporting mixed-content websites, where pages accessed over HTTPS also include content fetched over HTTP [68]. While security mechanisms or browser extensions reduce the attack surface, they do not entirely mitigate these attacks. A significant step towards improving user privacy, is the deployment of ubiquitous encryption. Naylor et al. [2] discussed the "cost" of a wide deployment of HTTPS and analyzed aspects such as infrastructure costs, latency, data usage, and energy consumption. However, even when the connection is encrypted, previous work has demonstrated the feasibility of a wide range of attacks at both application and cryptographic level that can subvert the protection [10], [69]–[72]. Fahl et al. [73], [74] explored such attacks in the mobile domain.

Deanonymizing Tor users. Huber et al. [75] discussed how Tor users could be deanonymized by PII being leaked in HTTP traffic. Chakravarty et al. [76] proposed the use of decoy traffic with fake credentials for detecting adversaries monitoring traffic from Tor exit nodes. While their prototype focused on IMAP and SMTP servers, their technique could be extended to also leverage decoy accounts in major websites. If the attacker doesn't change the account in a visible way, this technique will only detect attacks if the service offers information about previous logins (e.g., as Gmail does). Winter et al. [77] deployed their tool HoneyConnector for a period of 4 months, and identified 27 Tor exit nodes that monitored outgoing traffic and used stolen decoy credentials.

IX. Conclusion

In this paper we presented our extensive in-depth study on the privacy threats that users face when attackers steal their HTTP cookies. We audited a wide range of major services and found that cookie hijacking attacks are not limited to a specific type of websites, but pose a widespread threat to any website that does not enforce ubiquitous encryption. Our study revealed numerous instances of major services exposing private information and protected account functionality to non-authenticated cookies. This threat is not restricted to websites, as users' cookies are also exposed by official browser extensions, search bars and mobile apps. To obtain a better understanding of the risk posed by passive eavesdroppers in practice, we conducted an IRB-approved measurement study and detected that a large portion of the outgoing traffic in public wireless networks remains unencrypted, thus, exposing a significant amount of users to cookie hijacking attacks. We also evaluated the protection offered by popular browser-supported security mechanisms, and found that they can reduce the attack surface but can not protect users if websites do not *support* ubiquitous encryption. The practicality and pervasiveness of these attacks, also renders them a significant threat to Tor users, as they can be deanonymized by adversaries monitoring the outgoing traffic of exit nodes.

X. Acknowledgements

We would like to thank the anonymous reviewers for their feedback. We would also like to thank the CUIT team of Joel Rosenblatt and the CRF team of Bach-Thuoc (Daisy) Nguyen at Columbia University, for their technical support throughout this project. Finally we would like to thank Georgios Kontaxis, Vasileios P. Kemerlis and Steven Bellovin for informative discussions and feedback. This work was supported by the NSF under grant CNS-13-18415. Author Suphannee Sivakorn is also partially supported by the Ministry of Science and Technology of the Royal Thai Government. Any opinions, findings, conclusions, or recommendations expressed herein are those of the authors, and do not necessarily reflect those of the US Government or the NSF.

References

[1] E. Butler, "Firesheep," 2010, http://codebutler.com/firesheep.

[2] D. Naylor, A. Finamore, I. Leontiadis, Y. Grunenberger, M. Mellia, M. Munafò, K. Papagiannaki, and P. Steenkiste, "The Cost of the "S" in HTTPS," in *Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '14. ACM, 2014, pp. 133–140.

[3] K. Singh, A. Moshchuk, H. J. Wang, and W. Lee, "On the Incoherencies in Web Browser Access Control Policies," in *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, 2010.

[4] C. Castelluccia, E. De Cristofaro, and D. Perito, "Private Information Disclosure from Web Searches," in *Privacy Enhancing Technologies*, ser. PETS '10, 2010.

[5] B. Krishnamurthy and C. E. Wills, "On the leakage of personally identifiable information via online social networks," in *Proceedings of the 2nd ACM workshop on Online social networks*, ser. WOSN '09, 2009.

[6] B. Krishnamurthy and C. Wills, "Privacy Leakage in Mobile Online Social Networks," in *Proceedings of the 3rd Workshop on Online Social Networks*, ser. WOSN '10, 2010.

[7] 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*, ser. WWW '15, 2015.

[8] Y. Liu, H. H. Song, I. Bermudez, A. Mislove, M. Baldi, and A. Tongaonkar, "Identifying Personal Information in Internet Traffic," in *Proceedings of the 3rd ACM Conference on Online Social Networks*, ser. COSN '15, 2015.

[9] B. Möller, T. Duong, and K. Kotowicz. (2014, Oct.) This POODLE bites: exploiting the SSL 3.0 fallback. https://googleonlinesecurity.blogspot.com/2014/10/this-poodle-bites-exploiting-ssl-30.html.

[10] M. Marlinspike, "New Tricks For Defeating SSL In Practice," *BlackHat DC*, Feb. 2009.

[11] M. Kranch and J. Bonneau, "Upgrading HTTPS in Mid-Air: An Empirical Study of Strict Transport Security and Key Pinning," in *Proceedings of the Network and Distributed System Security Symposium*, ser. NDSS '15, 2015.

[12] J. Hodges, C. Jackson, and A. Barth, "HTTP Strict Transport Security," RFC 6797, 2012.

[13] Can I use. HSTS Browser Support. http://caniuse.com/#feat=strictransportsecurity.

[14] L. Garron. HSTS Preload. https://hstspreload.appspot.com/.

[15] M. Stevens, A. Sotirov, J. Appelbaum, A. Lenstra, D. Molnar, D. A. Osvik, and B. De Weger, "Short chosen-prefix collisions for MD5 and the creation of a rogue CA certificate," in *Advances in Cryptology-CRYPTO 2009*, 2009, pp. 55–69.

[16] C. Palmer and C. Evans, "Certificate Pinning Extension for HSTS," RFC DRAFT, 2011.

[17] C. Palmer, C. Evans, and R. Sleevi, "Certificate Pinning Extension for HSTS," RFC 7469, 2015.

[18] N. Heninger, Z. Durumeric, E. Wustrow, and J. A. Halderman, "Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices," in *Proceedings of the 21st USENIX Security Symposium*, Aug. 2012.

[19] Y. Zhou and D. Evans, "Why Arent HTTP-only Cookies More Widely Deployed?" in *Proceedings of the Web 2.0 Security and Privacy 2010 workshop*, ser. W2SP '10, 2010.

[20] S. Fogie, J. Grossman, R. Hansen, A. Rager, and P. D. Petkov, *XSS Attacks: Cross Site Scripting Exploits and Defense*. Syngress, 2011.

[21] Randy Westergren. (2016) Widespread XSS Vulnerabilities in Ad Network Code Affecting Top Tier Publishers. http://randywestergren.com/widespread-xss-vulnerabilities-ad-network-code-affecting-top-tier-publishers-retailers.

[22] N. Perlroth, J. Larson, and S. Shane. (2013, Sep.) The New York Times - N.S.A. Able to Foil Basic Safeguards of Privacy on Web. http://www.nytimes.com/2013/09/06/us/nsa-foils-much-internet-encryption.html.

[23] R. Gallagher. (2015, Sep.) The Intercept - From Radio to Porn, British Spies Track Web Users Online Identities. https://theintercept.com/2015/09/25/gchq-radio-porn-spies-track-web-users-online-identities/.

[24] A. Soltani, A. Peterson, and B. Gellman. (2013, Dec.) The Washington Post - NSA uses Google cookies to pinpoint targets for hacking. https://www.washingtonpost.com/news/the-switch/wp/2013/12/10/nsa-uses-google-cookies-to-pinpoint-targets-for-hacking/.

[25] BBC News. (2014, Jul.) NSA 'targets' Tor web servers and users. http://www.bbc.com/news/technology-28162273.

[26] R. Gross and A. Acquisti, "Information Revelation and Privacy in Online Social Networks," in *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*, ser. WPES '05, 2005.

[27] I. Polakis, G. Argyros, T. Petsios, S. Sivakorn, and A. D. Keromytis, "Where's wally? precise user discovery attacks in location proximity services," in *CCS '15*, 2015, pp. 817–828.

[28] A. Hannak, P. Sapiezynski, A. Molavi Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove, and C. Wilson, "Measuring Personalization of Web Search," in *Proceedings of the 22nd International Conference on World Wide Web*, ser. WWW '13, 2013.

[29] X. Xing, W. Meng, D. Doozan, A. C. Snoeren, N. Feamster, and W. Lee, "Take This Personally: Pollution Attacks on Personalized Services," in *Proceedings of the 22nd USENIX Security Symposium*, 2013.

[30] A. Chaabane, G. Acs, and M. A. Kaafar, "You Are What You Like! Information Leakage Through Users Interests," in *Proceedings of the Network and Distributed System Security Symposium*, ser. NDSS '12, 2012.

[31] comScore. (2015, Aug.) July 2015 U.S. Desktop Search Engine Rankings. http://www.comscore.com/Insights/Market-Rankings/comScore-Releases-July-2015-U.S.-Desktop-Search-Engine-Rankings?

[32] A. Acquisti, R. Gross, and F. Stutzman, "Faces of facebook: Privacy in the age of augmented reality," *BlackHat*, 2011.

[33] I. Polakis, G. Kontaxis, S. Antonatos, E. Gessiou, T. Petsas, and E. P. Markatos, "Using Social Networks to Harvest Email Addresses," in *Proceedings of the 9th Annual ACM Workshop on Privacy in the Electronic Society*, ser. WPES '10, 2010.

[34] F. M. Harper, D. Raban, S. Rafaeli, and J. A. Konstan, "Predictors of Answer Quality in Online Q&Amp;A Sites," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '08, 2008.

[35] D. Pelleg, E. Yom-Tov, and Y. Maarek, "Can You Believe an Anonymous Contributor? On Truthfulness in Yahoo! Answers," in *SOCIALCOM-PASSAT '12*, 2012.

[36] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov, "You Might Also LIke: Privacy Risks of Collaborative Filtering," in *Proceedings of the 2011 IEEE Symposium on Security and Privacy*, 2011.

[37] J. Y. Tsai, S. Egelman, L. Cranor, and A. Acquisti, "The Effect of Online Privacy Information on Purchasing Behavior: An Experimental Study," *Info. Sys. Research*, vol. 22, no. 2, 2011.

[38] N. Christin, S. S. Yanagihara, and K. Kamataki, "Dissecting One Click Frauds," in *Proceedings of the 17th ACM Conference on Computer and Communications Security*, 2010.

[39] U. Chareca, "Inferring user demographics from reading habits," Master's thesis, Linköping University, 2014.

[40] J. R. Mayer and J. C. Mitchell, "Third-Party Web Tracking: Policy and Technology," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, 2012.

[41] 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*, ser. NSDI '12, 2012.

[42] P. Gill, V. Erramilli, A. Chaintreau, B. Krishnamurthy, K. Papagiannaki, and P. Rodriguez, "Follow the Money: Understanding Economics of Online Aggregation and Advertising," in *Proceedings of the 2013 Conference on Internet Measurement Conference*, ser. IMC '13, 2013.

[43] P. Barford, I. Canadi, D. Krushevskaja, Q. Ma, and S. Muthukrishnan, "Adscape: Harvesting and Analyzing Online Display Ads," in *Proceedings of the 23rd International Conference on World Wide Web*, ser. WWW '14, 2014.

[44] A. Datta, M. C. Tschantz, and A. Datta, "Automated Experiments on Ad Privacy Settings: ATale of Opacity, Choice, and Discrimination," *Proceedings on Privacy Enhancing Technologies*, vol. 2015, no. 1, 2015.

[45] M. Lécuyer, G. Ducoffe, F. Lan, A. Papancea, T. Petsios, R. Spahn, A. Chaintreau, and R. Geambasu, "XRay: Enhancing the Web's Transparency with Differential Correlation," in *Proceedings of the 23rd USENIX Security Symposium*, 2014.

[46] A. Korolova, "Privacy violations using microtargeted ads: A case study," in *Proceedings of the 2010 IEEE International Conference on Data Mining Workshops*, ser. ICDMW '10, 2010.

[47] A. Kapravelos, C. Grier, N. Chachra, C. Kruegel, G. Vigna, and V. Paxson, "Hulk: Eliciting Malicious Behavior in Browser Extensions," in *Proceedings of the 23rd USENIX Security Symposium*, 2014.

[48] Cisco. (2015, May) Visual Networking Index, Global Traffic Forecast. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html.

[49] PurpleWiFi. (2014, Jun.) Our latest survey: how do people use WiFi in public places? http://www.purplewifi.net/latest-survey-people-use-wifi-public-places/.

[50] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-generation Onion Router," in *Proceedings of the 13th USENIX Security Symposium*, ser. SSYM '04, 2004.

[51] EFF. HTTPS Everywhere. https://www.eff.org/https-Everywhere.

[52] X. Zheng, J. Jiang, J. Liang, H. Duan, S. Chen, T. Wan, and N. Weaver, "Cookies Lack Integrity: Real-World Implications," in *Proceedings of the 24th USENIX Security Symposium*, 2015.

[53] A. P. Felt, A. Ainslie, R. W. Reeder, S. Consolvo, S. Thyagaraja, A. Bettes, H. Harris, and J. Grimes, "Improving SSL Warnings: Comprehension and Adherence," in *Proceedings of the Conference on Human Factors and Computing Systems*, 2015.

[54] B. Potter, "Wireless Hotspots: Petri Dish of Wireless Security," *Commun. ACM*, vol. 49, no. 6, Jun. 2006.

[55] A. Bortz, A. Barth, and A. Czeskis, "Origin cookies: Session integrity for web applications," in *Proceedings of the Web 2.0 Security and Privacy 2011 workshop*, ser. W2SP '11, 2011.

[56] R. Wang, S. Chen, and X. Wang, "Signing Me onto Your Accounts through Facebook and Google: a Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, 2012.

[57] C. Karlof, U. Shankar, J. D. Tygar, and D. Wagner, "Dynamic Pharming Attacks and Locked Same-origin Policies for Web Browsers," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, 2007.

[58] S. Lekies, B. Stock, M. Wentzel, and M. Johns, "The Unexpected Dangers of Dynamic JavaScript," in *Proceedings of the 24th USENIX Security Symposium*, 2015.

[59] A. Barth, C. Jackson, and J. C. Mitchell, "Robust Defenses for Cross-Site Request Forgery," in *Proceedings of the 15th ACM Conference on Computer and Communications Security*, 2008.

[60] N. Nikiforakis, W. Meert, Y. Younan, M. Johns, and W. Joosen, "SessionShield: Lightweight Protection against Session Hijacking," in *Engineering Secure Software and Systems*, ser. ESSoS '11, 2011.

[61] P. De Ryck, L. Desmet, F. Piessens, and W. Joosen, "SecSess: Keeping Your Session Tucked Away in Your Browser," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, ser. SAC '15, 2015.

[62] M. Johns, "SessionSafe: Implementing XSS Immune Session Handling," in *Proceedings of the 11th European conference on Research in Computer Security*, ser. ESORICS' 06, 2006.

[63] C. Jackson and A. Barth, "ForceHTTPS: Protecting high-security web sites from network attacks," in *Proceedings of the 17th International World Wide Web Conference*, ser. WWW '08, 2008.

[64] J. Selvi, "Bypassing HTTP Strict Transport Security," *BlackHat-EU*, 2014.
[65] K. Bhargavan, A. Delignat-Lavaud, C. Fournet, , A. Pironti, and P.-Y. Strub, "Triple Handshakes and Cookie Cutters: Breaking and Fixing Authentication over TLS," in *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, 2014.
[66] S. Sivakorn, I. Polakis, and A. D. Keromytis, "I am robot: (deep) learning to break semantic image captchas," in *IEEE European Symposium on Security and Privacy (EuroS&P) 2016*.
[67] E. Toch, Y. Wang, and L. Cranor, "Personalization and privacy: a survey of privacy risks and remedies in personalization-based systems," *User Modeling and User-Adapted Interaction*, vol. 22, no. 1-2, 2012.
[68] P. Chen, N. Nikiforakis, C. Huygens, and L. Desmet, "A dangerous mix: Large-scale analysis of mixed-content websites," in *Proceedings of the 16th Information Security Conference*, 2013.
[69] J. Clark and P. C. van Oorschot, "SoK: SSL and HTTPS: Revisiting Past Challenges and Evaluating Certificate Trust Model Enhancements," in *Proceedings of the 2013 IEEE Symposium on Security and Privacy*.
[70] S. Chen, Z. Mao, Y.-M. Wang, and M. Zhang, "Pretty-bad-proxy: An overlooked adversary in browsers' https deployments," in *Proceedings of the 2009 IEEE Symposium on Security and Privacy*, 2009.
[71] Z. Durumeric, J. Kasten, D. Adrian, J. A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, and V. Paxson, "The Matter of Heartbleed," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, ser. IMC '14, 2014, pp. 475–488.
[72] D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. Valenta, B. VanderSloot, E. Wustrow, S. Zanella-Béguelin, and P. Zimmermann, "Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice," in *Proceedings of the 22nd ACM Conference on Computer and Communications Security*, 2015.
[73] S. Fahl, M. Harbach, T. Muders, L. Baumgärtner, B. Freisleben, and M. Smith, "Why Eve and Mallory Love Android: An Analysis of Android SSL (in)Security," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, 2012.
[74] S. Fahl, M. Harbach, H. Perl, M. Koetter, and M. Smith, "Rethinking SSL Development in an Appified World," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, 2013.
[75] M. Huber, M. Mulazzani, and E. Weippl, "Tor http usage and information leakage," in *Communications and Multimedia Security*, 2010.
[76] S. Chakravarty, G. Portokalidis, M. Polychronakis, and A. D. Keromytis, "Detecting Traffic Snooping in Tor Using Decoys," in *Recent Advances in Intrusion Detection*, 2011.
[77] P. Winter, R. Köwer, M. Mulazzani, M. Huber, S. Schrittwieser, S. Lindskog, and E. Weippl, "Spoiled Onions: Exposing Malicious Tor Exit Relays," in *Privacy Enhancing Technologies Symposium*, 2014.

APPENDIX

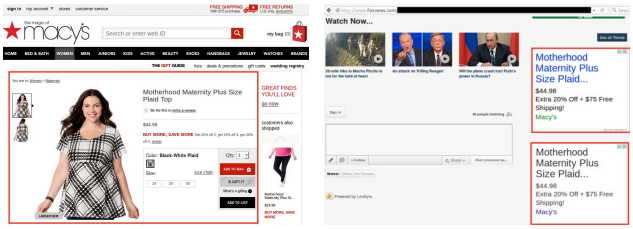
A. Information Leakage

Here we provide some details or information on certain services that were omitted from Section III.

Doubleclick. Figure 6 contains screenshots of our experiment that demonstrates how ad networks can reveal parts of a user's browsing history.

CNN. Almost the entire website runs over HTTP, including the login page, which can be exploited by active adversaries to modify or inject content. The credentials, however, are sent over HTTPS, preventing eavesdroppers from hijacking the user's session. Nonetheless, the HTTP cookie allows the attacker to view and edit the user's profile, which includes first and last name, postal address, email and phone number, profile picture and link to the user's Facebook account. Furthermore, the attacker can write or delete article comments, and also obtain the recently viewed or created reports on iReport, CNN's citizen journalism portal[6].

[6]http://ireport.cnn.com/



(a) Browsed page (b) Attacker receives ads exposing browsed page.

Fig. 6. Side-channel leak of user's browsing history by the Doubleclick ad network.

New York Times. The HTTP cookie allows the adversary to obtain or change the user's profile photo, name and last name, a link pointing to a personal homepage, and a short personal description (bio). The adversary can also obtain and edit the list of articles that the user has saved.

The Guardian. Stolen HTTP cookies provide access to the user's public profile sections, which includes a profile picture and username, a short bio, the user's interests, and previous comments on articles. The adversary can also post comments as the user.

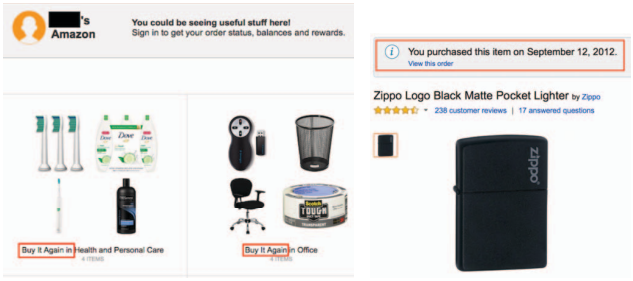
Huffington Post. Similar to CNN, almost the entire website runs overs unencrypted connections, and the HTTP cookie allows read and edit access to the user's profile, article subscriptions, comments, fans and followings. The profile includes the user's login name, profile photo, email address, biography, postal code, city and state. The attacker can also change the user's password, or delete the account.

```
{"User-Agent": "Mozilla/5.0 (iPhone; CPU iPhone OS
   9_0_2 like Mac OS X) AppleWebKit/601.1.46
   (KHTML, like Gecko) Mobile/13A452",
"cookie_name": "amzn-app-ctxt",
"cookie_value": "1.4%20
   {"os":"iOS"
   "ov":"9.0.2"
   "an":"Amazon"
   "av":"5.3.0"
   "dm":{"w":"640" "h":"960" "ld":"2.000000"}
   "uiv":5 "nal":1
   "cp":8xxxxx
   "xv":"1.11"
   "di":{"ca":"AT&T"
        "ct":"Wifi"
        "mf":"Apple"
        "pr":"iPhone"
        "md":"iPhone"
        "v":"4S"
        "dti":"A287xxxxxxxxx"
   }}"
}
```

Listing 2. User information disclosed in the value attribute of Walmart's HTTP customer cookie (values have been changed for privacy).

Amazon. the adversary can obtain information regarding previously purchased items either through the recommendation page (Figure 7(a)) or through product pages (Figure 7(b)). The iOS versions of the Amazon app also exposes information about the user's mobile device, as shown in Listing 2.

Ebay. Apart from the login and checkout pages, the remaining Ebay website runs over HTTP. As a result, the stolen



(a) Recommendations Page (b) Product Page

Fig. 7. Obtaining information about previously purchased items from user's Amazon account.

HTTP cookie gives the adversary access to both personal information and account functionality.

Personal information. The site always reveals the user's first name. Also, depending on what the victim uses for logging in (username or email address) is also exposed. By forging a cookie with the same value but a different scope (domain and path), we are also able to obtain the user's delivery address. The HTTP cookies can also access the user's messages, which are normally served over HTTPS.

History. The cookie provides access to the functionality that exposes the victim's purchase history, and also allows us to view and edit the items in the victim's watch and wish-lists. We can also see which items have been bought or bid upon in the past, and all the items being sold by the. victim.

Cart. Similarly to the other e-commerce websites we tested, the HTTP cookie enables access to the cart,. for viewing items already in it, or adding/removing items.

Walmart. If the adversary appends the stolen cookies when connecting, the website will reveal the user's first name, postcode, and also allow editing of the cart. However, upon inspection, we found that the `customer` HTTP cookie actually contains 34 fields of information about the user within its `value` attribute. Apart from the subset that can be seen in Listing 3, which includes the user's first and last name and email address, the cookie also contains ID information that points to the user's reviews and comments, and a tracking ID for third parties.

Target. As with most e-commerce sites, the stolen cookie reveals the user's first name, email address, and the ability to view and edit the cart, and the user's wish-list. Furthermore, it also reveals items recently viewed by the user.

Vendor-assisted attacks. The cookie exposes functionality that can be leveraged for deploying spam, similarly to Amazon. The attacker can either add items in the cart and send an email about those items (sent by `orders@service.target.com`), or create and send a wish-list (sent by `noreply@service.target.com`). In both cases, the emails explicitly contain the user's full name (thus, making the last name obtainable to the attacker). While the attacker cannot include any text, which would facilitate deploying spam or phishing campaigns, one could promote arbitrary items. The attacker can also deploy the two aforementioned extortion scams.

```
{"domain": ".walmart.com",
"name": "customer",
"path": "/",
"secure": false,
"httpOnly": false,
"value": "%7B%22firstName%22%3A%22JANE%22%2C%22
   lastName%22%3A%22DOE%22%2C%22
   emailAddress%22%3A%22janedoe%40example.com%22%2C%22
   isMigrated%22%3Atrue%2C%22
   omsCustomerId%22%3A%22xxxxxxx9%22%2C%22
   ReviewUser%22%3A%7B%22isValid%22%3Atrue%2C%22
   AdditionalFieldsOrder%22%xxxxxxxx%2C%22
   Avatar%22%3A%7B%7D%2C%22
   UserNickname%22%3A%22xxxxxxxxx%22%2C%22
   Photos%22%3A%5B%5D%2C%22
   ContextDataValues%22%xxxxxxxx%2C%22
   Videos%22%3A%5B%5D%2C%22
   ContextDataValuesOrder%22%3A%xxxxxxxxxC%22
   SubmissionId%22%3Axxxxx%2C%22
   ContributorRank%22%3A%22xxxx%22%2C%22
   StoryIds%22%xxxxxxxC%22
   AnswerIds%22%xxxxxx%2C%22
   QuestionIds%22xxxxxD%2C%22
   BadgesOrder%22xxxxxx%2C%22
   Badges%22%xxxxxxxxx2C%22
   Location%22%3Axxxxx%2C%22
   SecondaryRatingsOrder%22xxxxxx%2C%22
   ProductRecommendationIds%22%3A%xxxxxxxxx2C%22
   AdditionalFields%22%3A%7B%7D%2C%22
   SubmissionTime%22%3A%2220xx-xx-xxxxxxxxxxxx%22%2C%22
   ModerationStatus%22%3A%22APPROVED%22%2C%22
   ReviewIds%22%3xxxxxxx%2C%22
   ThirdPartyIds%22%xxxxxxxxxC%22
   Id%22%3A%22ff79xxxxxxxxxxxx509dc5%22%2C%22
   CommentIds%22%3A%5B%5D%2C%22
   SecondaryRatings%22%3Axxxxx%2C%22
   LastModeratedTime%22%3A%2220xxxxxxxxxxxxxx%22%2C%22
   reviewStatus%22%3A%7B%22
   hasReview%22%3Axxxxx%7D%7D%7D"
}
```

Listing 3. User information disclosed in the value attribute of Walmart's HTTP customer cookie (values have been changed for privacy).