Exclusive: How the (synced) Cookie Monster breached my encrypted VPN session

Panagiotis Papadopoulos
FORTH-ICS, Greece
panpap@ics.forth.gr

Nicolas Kourtellis
Telefonica Research, Spain
nicolas.kourtellis@telefonica.com

Evangelos P. Markatos
FORTH-ICS, Greece
markatos@ics.forth.gr

ABSTRACT

In recent years, and after the Snowden revelations, there has been a significant movement in the web from organizations, policymakers and individuals to enhance the privacy awareness among users. As a consequence, more and more publishers support TLS in their websites, and vendors provide privacy and anonymity tools, such as secure VPNs or Tor onions, to cover the need of users for privacy-preserving web browsing. But is *the sporadic appliance of such tools enough to provide privacy?*

In this paper, we describe two privacy-breaching threats against users accessing the Internet over a secure VPN. The breaches are made possible through Cookie Synchronization, nowadays widely used by third parties for advertisement and tracking purposes. The generated privacy leaks can be used by a snooping entity such as an ISP, to re-identify a user in the web and reveal their browsing history even when users are hidden behind a VPN. By probing the top 12K Alexa sites, we find that 1 out of 13 websites expose their users to these privacy leaks.

CCS CONCEPTS

• **Security and privacy → Pseudonymity, anonymity and untraceability**; **Web protocol security**; • **Information systems** → Online advertising;

KEYWORDS

Cookie Synchronization, TLS browsing session, User Tracking, Web Privacy, Anonymity

ACM Reference Format:
Panagiotis Papadopoulos, Nicolas Kourtellis, and Evangelos P. Markatos. 2018. Exclusive: How the (synced) Cookie Monster breached my encrypted VPN session. In *EuroSec'18: 11th European Workshop on Systems Security , April 23–26, 2018, Porto, Portugal.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3193111.3193117

1 INTRODUCTION

In recent years, the rise of targeted advertising and the high revenues it produces (2.7× more revenue and 2× more efficient than non-targeted [2]) has lured data brokers and trackers to collect increasingly more user data [17], which can process to produce audience segments and sell them to advertisers [18]. To make matters worse, last year, the USA House of Representatives voted to reverse FCC's regulations that prevented Internet Service Providers (ISPs) from selling users web-browsing data without their explicit consent [14]. Apart from the profit-oriented data collection, ad-related tracking techniques have been also exploited by agencies (as proven after the Snowden revelations [36]) to increase the effectiveness of their mass surveillance mechanisms.

This intensive data collection and use beyond the control of the end-users, in some cases has been characterized as pervasive or even intruding, thus, raising significant privacy concerns [22, 36]. These privacy concerns have made Internet users more privacy-aware [4], leading browser vendors to adopt more privacy preserving features in their products (e.g., anti-tracker solutions [6, 10], integrated VPN channels [28], etc.). Therefore, it is easy to see why a rapidly increasing number of users communicate with content providers (i.e., websites) through secure HTTPS connections [13]. The use of TLS authenticates the remote content provider and guarantees the integrity and confidentiality of any sensitive information that is transmitted from and to the web server. In this way, no third party located between the two ends can monitor the content the user browses and, thus, what their interests and preferences are.

Additionally, there are users who add another layer of protection by redirecting all of their traffic through a Virtual Private Network (VPN) connection (or Tor-like onion chains) [26, 32]. By using such secure tunneling, users can hide their actual IP address (and consequently their geolocation), thus preserving their anonymity. Thereby, users can prevent not only the ISP from monitoring what they browse, but also the web server itself from learning where the user is located both in the network, and geographically.

In literature, there have been several attacks recorded against both HTTPS and VPN-secured traffic. Regarding HTTPS, in [16] the authors present a classifier with which an eavesdropper can conduct traffic fingerprinting to estimate what the user browses based on specific traffic characteristics such as the size of the web page. In [5], authors present impersonation attacks against TLS by exploiting combinations of RSA and DH key exchange, session resumption, and renegotiation. In addition, there are specific vulnerabilities found in commercial VPN, such as DNS hijacking and IPv6 leakage presented in [33], or the WebRTC bug [34]. Although the above approaches depict the susceptibility of the protocols, they either provide results of specific accuracy, or they mostly rely on particular vulnerabilities of the deployed commercial VPN service.

In this paper, we prove that, even in the case of a perfectly secured VPN service, *the preservation of the users privacy is still not a given.* Specifically, we show that the so called Cookie Synchronization mechanism that different advertisers increasingly use [1, 27] to sync their cookies for the same users, can compromise the anonymity of the users under a broad range of circumstances. And this can

EuroSec'18, April 23–26, 2018, Porto, Portugal

ACM ISBN 978-1-4503-5652-7/18/04.
https://doi.org/10.1145/3193111.3193117

happen even when these users are connected to the Internet with a secure VPN. Specifically, in Cookie Synchronization, tokens securely stored in TLS cookies, which are able to uniquely identify a user (userID), are being shared with third parties allowing them to re-identify the user across different websites, even if the user hides their real IP address. When this Cookie Synchronization procedure takes place over plain HTTP, it is not only the syncing entities that learn the userID of the user, but also a snooping entity such as an internet service provider (ISP), which can follow the user's browsing habits just by keeping track of the synced cookies.

In particular, the contributions of this paper are the following:

- We present in full detail the above attack, and show how a last mile observer can snoop parts of a user's browsing history through Cookie Synchronization.
- We build a weblog analyzer able to parse HTTP(S) requests, detect Cookie Synchronizations and check for possible browsing history and ID spilling from secure TLS sessions.
- As a proof of concept, we perform active measurements through our secure VPN, by probing the top 12K Alexa sites, and explore the feasibility and spread of these attacks in the wild. By fetching the landing pages, we collect a dataset of $440K$ HTTP(S) requests, and by using it as input to our analyzer, we show that 1 out of 13 of the most popular websites worldwide expose their users to these two privacy-breaching attacks.

1.1 Threat model

In this paper, we assume a *curious* monitoring entity (e.g., an ISP) which is interested in collecting user data (such as location and browsing patterns or interests), that can afterwards sell to anyone interested (i.e, data management platforms, advertisers or data brokers [9]). Of course, this curious ISP does not want to jeopardize its own users-clients, therefore it infers user data only by passively monitoring the network traffic it routes. In the early years of HTTPS, which had a small portion of the Internet traffic, this passive monitoring was a trivial process. However, lately, more than half of the Internet [13] uses encryption and TLS to prevent prying eyes from accessing the contents of the secure communication.

In addition to the popular use of HTTPS from many websites, there are privacy-aware users who tunnel their HTTPS web traffic through secure VPN, in order to not only preserve the privacy of the content they browse, but also preserve their anonymity. In the rest of the paper, we assume a user who utilizes a secure VPN service that is perfectly protected by known vulnerabilities such as the IPv6/WebRTC leaking and DNS hijacking.

2 THE PRIVACY LEAK

In the modern web, user data matters; the more such data an online entity owns, the higher its overall marketing value. Evidently, there is an increasing growth of tracking entities [20] that collect many data for the web users such as interests, behavioral data, etc. However, for all these data to make sense and be monetizable, their aggregators must have a way to combine them. That is, a common identifier must exist, that could bind together the different user profiles belonging to the same user, thus, allowing the different entities to merge their datasets [1]. To provide this common identifier, the technology of Cookie Synchronization came to the rescue.

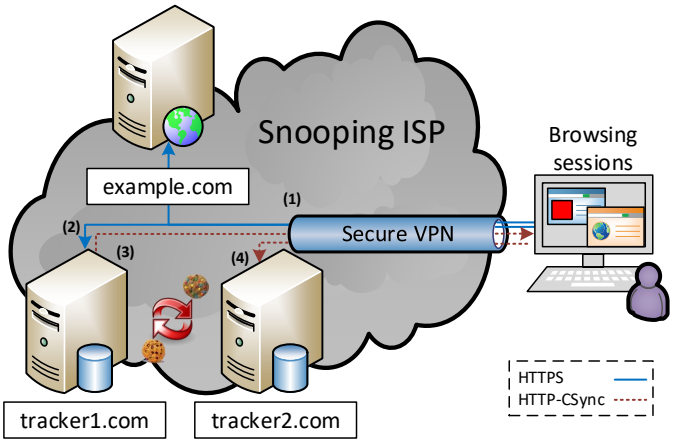

Figure 1: High level overview of the TLS session leak. A privacy-aware user (1) visits a webpage (*example.com*) over TLS and VPN. (2) It sends tracking information to tracker1.com, and receives its cookie over TLS. (3,4) It takes only a HTTP-based Cookie Synchronization (among tracker1.com and tracker2.com) in order to spill user unique identifiers and visited website. Then, a snooping ISP can re-identify the user just by monitoring the synced cookies, even if their real IP address is hidden.

2.1 What is Cookie Synchronization?

Cookie Synchronization, as first presented by Olejnik et al. in [27], is a technique used by third-party domains to bypass the same-origin policy [37] and match the different pseudonymous user IDs they have assigned to the same user. These user IDs are stored in cookies on the user-side, and the syncing procedure takes place by piggybacking the user ID in the URL of a request from the third party A included in the visited website W, to another collaborating party B. Party A may have its own iframe embedded in website W, or just a single 1×1 pixel image (web beacon [23]), which is enough to receive a GET image request and redirect the browser to $B's$ domain with its user ID piggybacked.

Cookie Synchronization provides the potential for more effective and persistent user tracking, allowing the tracking entities (i) to reach a previously inaccessible user (B can drop its own cookie), (ii) to synchronize their databases in the background [1] using the matched user IDs, and (iii) when used in conjunction with ever-cookie [35], to re-spawn deleted cookies and link a user's browsing histories from before and after a cookie erasure.

2.2 Spilling out of TLS

It is well known, that the basic principle of TLS channels is the confidentiality of the transmitted information, which guarantees the privacy and security of the communication. However, TLS does not guarantee these principles in a plug-and-play manner. There are several guidelines [11] warning about the harmful practice of mixing encrypted and non-encrypted content in TLS sessions. Yet, there is still a significant portion of publishers that fail to maintain adequately the security of their TLS channels [29].

In this paper, we demonstrate the massive privacy threat that HTTP-based Cookie Synchronization can impose to the user's TLS sessions. A possible scenario for this leak, as depicted in Figure 1, is the following: Let's assume a privacy-aware user, who conducts multiple browsing sessions from their PC, and uses a secure VPN to hide their true IP address. At some point, they visit *https://example.com*: a trustworthy reputable website, which requires a secure TLS channel establishment (Step 1). The website is ad-supported and thereby it includes ad- and analytic- related third parties that deliver effective personalized advertisements to the visitors. However, as we noted, *https://example.com* is a well reputable website and therefore it includes only TLS supporting third parties.

One of these third parties is *https://tracker1.com*. This third party securely sets a cookie with a user unique identifier (userID) $ID = user123$ on the user's side, in order to re-identify this user in case of a next visit to a site that this third party is similarly included (Step 2). Immediately after, it performs a Cookie Synchronization in order to share the set userID with a remote collaborating domain *http://tracker2.com* (Step 3). Technically, *tracker1.com* redirects an HTTPS request coming from the user's browser to *http://tracker2.com* (Step 4), while it loads the location URL with its userID:

3xx Redirect request Headers
Location: tracker2.com?syncID=**user123**&partner=tracker1
Referrer:{example.com}

Response Headers
Set-Cookie: {cookie_ID=**userABC**}

This Redirect request allows *tracker2.com* to (i) learn the syncedID (i.e. the userID of tracker1.com), and (ii) respond back with a Set-Cookie, thus setting its own cookie in the user's browser (Note: Prior to Cookie Synchronization, tracker2.com was not included in any way in *example.com* and thus it had no reach to the user).

A careful reader may have already detect the severe privacy leaks in this scenario:

(1) **Common userID leak:** The HTTP redirection exposes the TLS cookie to both the synced remote party and the snooping ISP. By allowing *tracker2.com* to set its own cookie on the user's browser in plaintext, the ISP is allowed to learn $user123 == userABC$. In this way, the ISP can re-identify the user in the web from now on, by leveraging the requests of *tracker2.com*. Specifically, whenever it captures HTTP requests to *tracker2.com* with cookie $ID = userABC$, it can identify who the user is, even if they use VPN or mixnets to hide their real IP address. Obviously, the more third party entities participate in the Cookie Synchronization the easier it is for the ISP to capture HTTP requests loaded with these synced cookies and, thus, re-identify the user.

(2) **Browsing history leak:** There are specific directives (Section 14.36 in RFC 2616 [19]) instructing web entities to either blanking or replacing with inaccurate data the referrer field of HTTP GET requests (referrer hiding), if it refers to a parent HTTPS page. This way, possible exposure of the visited website is fully prevented. Yet, there is an absence of similar directives for the case or HTTP Redirect requests as discussed by the web community. [1] A consequence of this aspect, is that unstripped referrer fields of redirections from HTTPS domains to HTTP domains harm the confidentiality of the secure session, by leaking the TLS-visited website to any monitoring body. Although, at first glance, this leak seems generic and not directly connected with Cookie Synchronization redirections, a careful reader may have observed that in our above scenario, the exposed synced userID of the Cookie Synchronization is the actual identifier that makes this referrer leak persistent. Specifically, Cookie Synchronization amplifies the traditional referrer leak by allowing the snooping ISP to bind the leaked visited website with a persistent ID, which will identify the user even after an IP address or Tor circuit change.

[1] https://stackoverflow.com/questions/2158283/will-a-302-redirect-maintain-the-referer-string/5441932#5441932

3 MEASURING ID-SPILLING IN THE WILD

To assess the feasibility of the leak we described earlier, we collected and analyzed a dataset from the most popular websites worldwide during December 2017. By using Selenium, we deployed a headless Chrome browser and crawled the landing pages of the top 12K Alexa websites, through the secure VPN of one of our institutions (FORTH). In this process, we fetch each website once, and after each probe, we erase the state of our browser. The overall volume of our dataset includes 440K HTTP(S) requests.

3.1 Cookie Syncing Detection Mechanism

To detect the Cookie Synchronization processes, first of all we extract all cookies set on the browser-side. Then, following techniques of previous works [27, 30] in the area, we create a collection of heuristics aiming to extract all IDs shared among the entities which could possibly constitute a userID.

First, we remove all cookies without expiration date (i.e., session cookies) and extract in a list, all of their stored IDs (i.e. cookieIDs) that could uniquely identify the user. Then, we parse the dataset with the HTTP requests to detect ID-looking strings carried (i) as parameters in every HTTP request's URL, or (ii) in the referrer URL. Such detected strings are a superset of the possible and valid cookie sync IDs. Each of such detected ID-looking strings is stored in a hashtable, along with the URL's domain (receiver of the ID). Thus, in case we have already seen the same ID in the past in a different URL, we consider the two requests as a ID sharing.

To ensure that we capture, and remove, cases of different domains owned by the same provider (e.g., doubleclick and googlesyndicate), we use several external sources like DNS whois, blacklists etc. By filtering out domains of the same provider, we can discriminate between intentional ID leaking and unequivocally legitimate cases of internal ID-sharing, thus, avoiding false positives. Finally, in order to verify if the detected shared IDs constitute unique user identifying IDs, we look for each of them in our list of cookie IDs. If there is a match, we consider this HTTP request as a Cookie Synchronization request, and the loaded ID as a synced userID.

3.2 Data analysis

We examine our dataset by building an analyzer which implements our Cookie Synchronization detection technique. We detect 89479 HTTP/HTTPS syncing requests that appear in 3878 websites (32% of the overall crawled websites) and synchronize with 733 different third-party domains, a number of 17171 unique identifiers (UserIDs).

Type	Amount
- Websites crawled	12000
- HTTP(S) requests	440000
- Websites with CSync	3878
- Total CSync redirections	89479
- Total unique synced IDs	17171
- Total unique 3rd party CSync domains	773
- SSL Syncing companies	475/733 (64.2%)
- non-SSL Syncing companies	258/733 (35.2%)
- TLS websites	8398/12000
- TLS websites with CSync	2317/8398
- TLS CSync redirections	58831
- Unique synced IDs in TLS websites	9045
- **Non-TLS syncings in TLS websites**	2879
- **Unique UserIDs leaked**	609/9045
- **Leaking TLS-protected websites**	174/2317

Table 1. Summary of results

From these 733 third parties, 35.2% does not support HTTPS. Table 1 summarizes our findings in this dataset.

We note the following caveats in our data collection process: First, the top Alexa list includes domains from supporting web services, such as CDNs, DDoS protection, or comment hosting services, etc. These are sites the user indirectly visits, while visiting the website they are actually interested in. Thus, it is highly unlikely that a real user in their everyday browsing behavior will visit such web services directly as 1st party websites. Second, due to the automated nature of our data collection, we had a few cases where the fetched domains denied to serve our crawler due to their anti-bot policy. Third, there are cases where third parties may avoid to perform ad-auctions [31] and Cookie Synchronizations whenever they detect requests from a headless browser, as the ad-ecosystem is not interested in serving targeted ads to bots. Considering all the aforementioned, our findings in this dataset are a lower bound of the problem. In fact, we believe that the portion of affected websites a user may face this privacy leakage while browsing the web, can be higher than reported here.

Cookie Synchronization vs. Websites: Given that in this paper, we investigate leaks from TLS-protected websites, we extract a subset of these websites which included 8398 distinct domains. From these domains, 2317 (27.5%) have at least one Cookie Synchronization. In total, the Cookie Synchronizations we detected in these TLS-protected websites is 58831 and 9045 unique userIDs, as reported in Table 1. In Figure 2, we plot the distribution of the portion of TLS-based synchronizations per website for TLS and non-TLS websites. As we see, the vast majority (92%) of the Cookie Synchronizations of TLS supported websites happen over TLS. However, we see a quite respectable 7.6% of synchronizations initiated over plain HTTP, risking the confidentiality of the entire browsing session.

In Table 2, we present a real example of the leaking cases we detected. In this example, our crawler visited over TLS the page *https://enfemenino.com*. In this website, 2 Cookie Synchronizations are performed, where *https://taboola.com* advertiser shares with *http://tapad.com* and *http://rlcdn.com* the ID it assigned to the user.

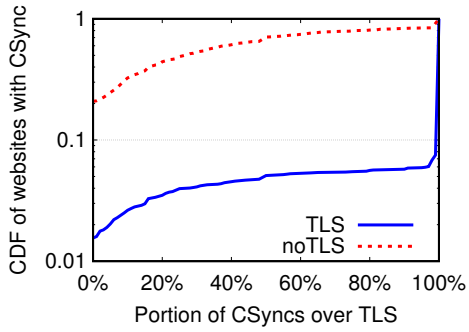

Figure 2: Distribution of the portion of TLS-based synchronizations per website for both TLS and non-TLS websites. As we see, the median non-TLS website has around 27% of its Cookie Synchronizations over TLS. On the other hand, most of the TLS-protected websites have 92% of their included synchronizations over TLS.

This way, the two tracking entities sync their set-cookies with the one of *https://taboola.com*. However, by doing that over plain HTTP, the visited website gets leaked through the referrer field to the monitoring ISP. In addition, this ISP from now on can skip the anonymity provided by the VPN and re-identify the user in the web just by monitoring the redirections and the HTTP cookies of *http://tapad.com* and *http://rlcdn.com*, even if the user's IP address is frequently changed.

TLS browsing session leak: To examine further the TLS websites that use non-TLS synchronizations, we plot the distribution of the plain-HTTP synchronizations per TLS website. As we see in Figure 3, some of these websites include synchronizations of userIDs over plain HTTP with up to 100 different third parties. In fact, 1 in 13 TLS-supported websites perform Cookie Synchronization via plain HTTP. This means that a snooping ISP has 100 times more chances to find, in the future, a HTTP request to one of these synced third parties and re-identify the user.

Role	Domain
- Visited website	https://enfemenino.com
- Cookie setter - SetCookie	https://taboola.com **ade87e60-5336-4dd9-9a2a-763e85516f6d-tuct150ff6a**
- Cookie syncer	http://idsync.rlcdn.com/382399.gif?partner_uid=**ade87e60-5336-4dd9-9a2a-763e85516f6d-tuct150ff6a**&redirect=1 referrer: **enfemenino.com** Get-cookie: {V2wkBRjV8XjspGYsUgWqL4jEl4=}
- Cookie syncer	http://pixel.tapad.com/idsync/ex/receive?partner_id=2227&partner_user_id=**ade87e60-5336-4dd9-9a2a-763e85516f6d-tuct150ff6a** referrer: **enfemenino.com** Get-cookie: {c57b29d1-f8e2-11e7-ac1b-0242ac110005}

Table 2. Example of leak in our dataset. Our crawler visited over TLS the *https://enfemenino.com*. Two Cookie Synchronizations appear, where *https://taboola.com* advertiser shares with *tapad.com* and *rlcdn.com* the ID it assigned to the user. By doing that over plain HTTP the visited website is leaked through the referrer field to the monitoring ISP.

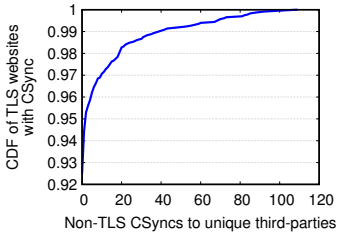

Figure 3: Distribution of non-TLS synchronizations per TLS website. Few websites (1 in 13) include quite a lot (up to 100!) of plain-HTTP CSync redirections.

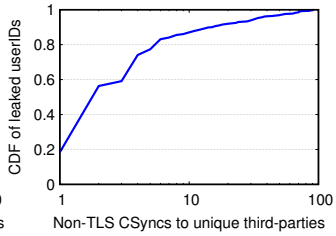

Figure 4: Distribution of the non-TLS synchronizations per leaked userID. There is a 10% of IDs that gets synced with more than 17 third parties.

Next, we examine the referrer fields of the syncing redirections from TLS websites, and check if the domain in the referrer field matches with the visited Alexa website. This check enables us to filter-out cases where the Cookie Synchronization was triggered by an embedded iframe of the website. In this case, the referrer field links to the iframe's domain.

We find 174 cases of websites, where referrer fields from HTTP Cookie Synchronizations leak the visited webpage along with full URL parameters. Besides that, through the same synchronizations, 610 unique userID were exposed to a monitoring ISP. Using this user information, the ISP can re-identify the user in the web and through the referrer-leaked webpages it can reconstruct the browsing history of the user. In Figure 4 we plot the distribution of the leaking synchronizations for each of these 610 userIDs. As we see, the median userID gets leaked by synchronizations with 2 different third parties. However, there is a 10% of user IDs that gets synced, and thus leaked, with more than 17 different third parties.

4 RELATED WORK

There are several studies presenting attacks against contemporary web privacy and security enhancing tools. Gonzalez et al. in [16] exploit the Server Name Indication (SN) extension of the TLS protocol to extract the domain the user visits, and then they built a classifier with which an eavesdropper can fingerprint the web traffic of a user to estimate the exact web pages they visit. To achieve that, they leverage specific traffic characteristics such as the transmitted volume of downloaded bytes. The accuracy of their method is particularly affected by the visited webpage's dynamic content and caching. Perta et al. in [33] conduct a study to investigate the privacy-related vulnerabilities of commercial VPN services. Their findings include vulnerabilities such as DNS hijacking and IPv6 leakage, through which the user's real IP address is leaked from the secure VPN, thus hindering their anonymity.

In our paper, we presented how a snooping ISP can exploit the Cookie Synchronization technique of the advertisers to track its customers. One of the first academic works that discussed the Cookie Synchronization mechanism is the paper of Olejnik et al. [27], which studies programmatic auctions from a privacy perspective, and presents Cookie Synchronization (that they call Cookie Matching) as an integral part of communication between the participating entities. In our study, we adopt their detection mechanism to detect Cookie Synchronizations.

Additionally, Acar et al. in [1] conduct a Cookie Synchronization privacy analysis by studying a small dataset of 3000 crawled sites. The authors study Cookie Synchronization in conjunction with respawning cookies and how, together, they affect the reconstruction of the user's browsing history by the trackers. They highlight the inadequacy of current anti-tracking policies. Specifically, enabling Do Not Track in their crawling browser only reduced the domains involved in synchronization by 2.9% and the number of synced IDs by 2.6%. In a recent census [12], authors measure Cookie Synchronization and its adoption in a set of 100K crawled sites, before highlighting the need of further investigation given its increased privacy implications. Their results show that 157 of top 200 (78%) third parties synchronize cookies with at least one other party.

Papadopoulos et al. in [30], used Cookie Synchronization as a metric to measure the privacy cost that personalized advertising imposes to the users. They analyzed a large dataset of mobile users and their results show that users receive a significant amount of Cookie Synchronization (3.4 synchronizations per ad-impression). Furthermore, a handful of third parties can learn up to 10% of the total unique userIDs of the median user across an entire year.

5 CONCLUSION

In recent years, the adoption of TLS has surpassed 50% of the websites worldwide. In addition, several browsing vendors provide more sophisticated anonymity preserving tools like Tor onions and VPN tunneling for privacy aware users. Yet, in this paper, we prove that these users are still far from being completely protected against eavesdropping entities and ISPs.

Specifically, we show that the tracking-related mechanism of Cookie Synchronization, can break the secure TLS session and (i) spill user unique identifiers (userIDs) along with (ii) the full URL that the user has visited over TLS. This way, a snooping ISP can re-identify the user in the web, even if they use VPN or Tor circuits, as well as reconstruct their browsing history. To assess the feasibility of this threat in the real world, we crawled the top 12K Alexa websites and after extracting the performed Cookie Synchronizations, we found that 1 out of 13 TLS-protected websites expose the privacy of their users.

Countermeasures: Nowadays, a lot of users deploy ad-blockers [25] to remove the annoying or resource consuming [30] ads. Indeed, ad-blockers can eliminate the privacy leak we present here by killing all third party domains. However, they would also kill the funding model of contemporary web, making content providers block ad-blocking visitors [21]. In addition, according to Englehardt et al. [12], less than half of the third parties (46%) in top websites use HTTPS, thus, impeding the overall adoption of HTTPS and generating lots of cases of mixed content in TLS supporting websites.

In effect, the most important countermeasure against this leak is to increase the adoption of HTTPS (in both web and mobile apps [29]), as major Non-Profit Organizations and Internet stakeholders promote [15]. This way, every single connection the user establishes with the remote servers to fetch a component in a mashup, will be secured. Of course, supporting HTTPS does not come cheaply. Despite of its huge advantages, one may argue that

there are specific latency and maintenance overheads, such as key and certificate maintenance, trust revocation handling, etc. [7, 8, 24]. Therefore, tracking entities may lack the incentives to deploy and deal with such overheads.

For this reason, we believe that it is the browser vendors that must (i) force the use of TLS by everyone and forbid *any* susceptible use of mixed content, and (ii) during request marshaling, strip *any* information (the referrer field in our case) may link together different type of traffic (HTTPS and HTTP). In fact, some privacy-sensitive browsers have already started providing such alternatives [6, 10]. By applying the above, not only the privacy of the users will be preserved, but the content providers will be fortified against visitor data and revenue loss [3].

Future Work: We plan to investigate further the characteristics of the TLS protected websites that are more prone to expose the privacy of the users. Specifically, by crawling a larger dataset of websites, we will conduct a deeper analysis of the content category and top-level domains of these websites. In addition, we plan to investigate to what extend the phenomenon appears in websites with lower popularity, considering that these websites may draw more 'sloppy' trackers that do not care about supporting TLS, and explore if there is an association with the popularity of such trackers.

Acknowledgments

The research leading to these results has received funding from the General Secretariat for Research and Technology (GSRT) of Greece, the Hellenic Foundation for Research and Innovation (HFRI) and European Union's Marie Sklodowska-Curie grant agreement No 690972. The paper reflects only the authors' view and the Agency and the Commission are not responsible for any use that may be made of the information it contains.

REFERENCES

[1] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. The web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, 2014.

[2] Howard Beales. The value of behavioral targeting. https://www.networkadvertising.org/pdfs/Beales_NAI_Study.pdf, 2010.

[3] Ross Benes. We get audience data at virtually no cost: Confessions of a programmatic ad buyer. https://digiday.com/marketing/get-audience-data-virtually-no-cost-confessions-programmatic-ad-buyer/, 2018.

[4] Josh Bernoff. Turns out consumers really do care about the data you're collecting. http://adage.com/article/digitalnext/turns-consumers-care-data-collecting/232331/, 2012.

[5] Karthikeyan Bhargavan, Antoine Delignat Lavaud, Cédric Fournet, Alfredo Pironti, and Pierre Yves Strub. Triple handshakes and cookie cutters: Breaking and fixing authentication over tls. In *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, SP '14, pages 98–113, Washington, DC, USA, 2014. IEEE Computer Society.

[6] Brave Software Inc. . Brave: A browser with your interests at heart. https://brave.com/, 2018.

[7] Antonios A Chariton, Eirini Degkleri, Panagiotis Papadopoulos, Panagiotis Ilia, and Evangelos P Markatos. DCSP: Performant certificate revocation a dns-based approach. In *Proceedings of the 9th European Workshop on System Security*, 2016.

[8] Antonios A Chariton, Eirini Degkleri, Panagiotis Papadopoulos, Panagiotis Ilia, and Evangelos P Markatos. CCSP: A compressed certificate status protocol. In *INFOCOM 2017-IEEE Conference on Computer Communications*, 2017.

[9] Privacy Rights Clearinghouse. Data brokers and your privacy. https://www.privacyrights.org/content/data-brokers-and-your-privacy, 2016.

[10] Cliqz GmbH. Cliqz: The no-compromise browser. https://cliqz.com/en/, 2018.

[11] Jo el van Bergen. Mixed content weakens https. https://developers.google.com/web/fundamentals/security/prevent-mixed-content/what-is-mixed-content, 2017.

[12] Steven Englehardt and Arvind Narayanan. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1388–1401. ACM, 2016.

[13] Klint Finley. Half the Web Is Now Encrypted. That Makes Everyone Safer. https://www.wired.com/2017/01/half-web-now-encrypted-makes-everyone-safer/, 2017.

[14] Klint Finley. The senate prepares to send internet privacy down a black hole. https://www.wired.com/2017/03/senate-prepares-send-internet-privacy-black-hole/, 2017.

[15] Electronic Frontier Foundation. Https everywhere. https://www.eff.org/https-everywhere, 2018.

[16] Roberto Gonzalez, Claudio Soriente, and Nikolaos Laoutaris. User profiling in the time of https. In *Proceedings of the 2016 ACM on Internet Measurement Conference*, pages 373–379. ACM, 2016.

[17] Annabelle Green. Customer data collection increased to improve customer experience, research finds. https://business-reporter.co.uk/2016/07/20/customer-data-collection-increased-improve-customer-experience-research-finds/, 2016.

[18] Rainey Reitman Kurt Opsahl. The disconcerting details: How facebook teams up with data brokers to show you targeted ads. https://www.eff.org/deeplinks/2013/04/disconcerting-details-how-facebook-teams-data-brokers-show-you-targeted-ads, 2013.

[19] Paul J Leach, Tim Berners-Lee, Jeffrey C Mogul, Larry Masinter, Roy T Fielding, and James Gettys. Encoding sensitive information in URI's. https://tools.ietf.org/html/rfc2616#section-15.1.3, 1999.

[20] Adam Lerner, Anna Kornfeld Simpson, Tadayoshi Kohno, and Franziska Roesner. Internet jones and the raiders of the lost trackers: An archaeological study of web tracking from 1996 to 2016. In *25th USENIX Security Symposium (USENIX Security 16)*, Austin, TX, 2016. USENIX Association.

[21] Brian Morrissey. Forbes starts blocking ad-block users. http://digiday.com/publishers/forbes-ad-blocking/, 2015.

[22] Paul Mozur. Internet Users in China Expect to Be Tracked. Now, They Want Privacy. https://www.nytimes.com/2018/01/04/business/china-alibaba-privacy.html, 2018.

[23] Brian H Murray and James J Cowart. Web bugs: A study of the presence and growth rate of web bugs on the internet. *Cyveillance Inc*, 2001.

[24] David Naylor, Alessandro Finamore, Ilias Leontiadis, Yan Grunenberger, Marco Mellia, Maurizio Munafò, Konstantina Papagiannaki, and Peter Steenkiste. The cost of the "s" in https. In *Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies*, CoNEXT '14, pages 133–140, New York, NY, USA, 2014. ACM.

[25] Netimperative. Ad-blocking soars 10% in just 3 months. http://www.netimperative.com/2016/01/ad-blocking-soars-10-in-just-3-months/.

[26] Lily Hay Newman. If you want a vpn to protect your privacy, start here. https://www.wired.com/2017/03/want-use-vpn-protect-privacy-start/, 2017.

[27] Lukasz Olejnik, Minh-Dung Tran, and Claude Castelluccia. Selling off user privacy at auction. In *21st Annual NDSS'14*, 2014.

[28] Opera Software. Free VPN in Opera browser. Surf the web with enhanced privacy. http://www.opera.com/computer/features/free-vpn, 2018.

[29] Elias P. Papadopoulos, Michalis Diamantaris, Panagiotis Papadopoulos, Thanasis Petsas, Sotiris Ioannidis, and Evangelos P. Markatos. The long-standing privacy debate: Mobile websites vs mobile apps. In *Proceedings of the 26th International Conference on World Wide Web*, WWW'17, 2017.

[30] Panagiotis Papadopoulos, Nicolas Kourtellis, and Evangelos P. Markatos. The cost of digital advertisement: Comparing user and advertiser views, 2018.

[31] Panagiotis Papadopoulos, Nicolas Kourtellis, Pablo Rodriguez Rodriguez, and Nikolaos Laoutaris. If you are not paying for it, you are the product: How much do advertisers pay to reach you? In *Proceedings of the 2017 Internet Measurement Conference*, IMC '17, 2017.

[32] Ian Paul. How and why you should use a VPN any time you hop on the internet. https://www.techhive.com/article/3158192/privacy/howand-whyyou-should-use-a-vpn-any-time-you-hop-on-the-internet.html, 2017.

[33] Vasile C Perta, Marco V Barbera, Gareth Tyson, Hamed Haddadi, and Alessandro Mei. A glance through the VPN looking glass: IPv6 leakage and DNS hijacking in commercial VPN clients. *Proceedings on Privacy Enhancing Technologies*, 2015(1):77–91, 2015.

[34] Daniel Roesler. STUN IP Address requests for WebRTC. https://github.com/diafygi/webrtc-ips, 2015.

[35] samy.pl. evercookie - virtually irrevocable persistent cookies, 2014.

[36] Ashkan Soltani, Andrea Peterson, and Barton Gellman. 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/, 2013.

[37] World Wide Web Consortium (W3C). Same origin policy. https://www.w3.org/Security/wiki/Same_Origin_Policy, 2010.