Tracking Users across the Web via TLS Session Resumption

Erik Sy
University of Hamburg

Christian Burkert
University of Hamburg

Hannes Federrath
University of Hamburg

Mathias Fischer
University of Hamburg

ABSTRACT

User tracking on the Internet can come in various forms, e.g., via cookies or by fingerprinting web browsers. A technique that got less attention so far is user tracking based on TLS and specifically based on the TLS session resumption mechanism. To the best of our knowledge, we are the first that investigate the applicability of TLS session resumption for user tracking. For that, we evaluated the configuration of 48 popular browsers and one million of the most popular websites. Moreover, we present a so-called prolongation attack, which allows extending the tracking period beyond the lifetime of the session resumption mechanism. To show that under the observed browser configurations tracking via TLS session resumptions is feasible, we also looked into DNS data to understand the longest consecutive tracking period for a user by a particular website. Our results indicate that with the standard setting of the session resumption lifetime in many current browsers, the average user can be tracked for up to eight days. With a session resumption lifetime of seven days, as recommended upper limit in the draft for TLS version 1.3, 65% of all users in our dataset can be tracked permanently.

CCS CONCEPTS

• **Security and privacy** → **Browser security**; **Privacy protections**;

KEYWORDS

Session IDs, Session Tickets, PSK Identity, Tracking Period, Browser Measurement

1 INTRODUCTION

User tracking via HTTP cookies and browser fingerprinting is a reality [2, 4, 15]. Tracking mechanisms are commonly used to observe conversions, namely whether an advertisement on website *A* leads to a desired user activity on website *B*. Herrmann et al. [5] revealed that temporary tracking mechanisms can be used to identify users based on their characteristic browsing patterns over longer time periods. They found that 85, 4% of users can be identified based

on their browsing behaviour if the temporary tracking mechanism lasts up to 24 hours. Also, the creation of long-term browsing profiles is possible, if a tracker can observe a large share of a user's browsing activity. Big players like Google and Facebook leverage the wide-spread use of their advertising networks and social plugins to track users across websites and gain detailed user profiles.

However, as users are increasingly aware of the privacy threat from tracking, they use privacy-friendly browsers, private browsing modes, and browser extensions to restrict tracking practices such as HTTP cookies. Browser fingerprinting got more difficult, as trackers can hardly distinguish the fingerprints of mobile browsers. They are often not as unique as their counterparts on desktop systems [4, 12]. Tracking based on IP addresses is restricted because of NAT that causes users to share public IP addresses and it cannot track devices across different networks. As a result, trackers have an increased interest in additional methods for regaining the visibility on the browsing habits of users. The result is a race of arms between trackers as well as privacy-aware users and browser vendors.

One novel tracking technique could be based on TLS session resumption, which allows abbreviating TLS handshakes by leveraging key material exchanged in an earlier TLS session. Thus, it introduces a possibility to link two TLS sessions. However, continuous user tracking via TLS session resumption is only possible as long as the browser is not restarted, because this clears the TLS cache. Especially mobile devices are *always on* and seldomly restarted. Finally, the feasibility of user tracking via TLS session resumption depends on the TLS configuration of both server and browser, as well as on the user's browsing behaviour. It is unknown so far whether this approach is feasible for user tracking in real-world scenarios.

To the best of our knowledge, we are the first to report on the applicability of TLS session resumption for user tracking. The main contributions of our paper are:

- We measure session resumption lifetimes of all Alexa Top Million websites and the same for 48 browsers. We assess the real-world configuration of these mechanisms and derive the possible duration of user tracking, respectively.
- We introduce the *prolongation attack* that allows to extend the tracking period beyond the session resumption lifetime.
- We analyse the impact of the prolongation attack on the resulting tracking periods and on the ratio of permanently trackable users, based on an additional DNS dataset to derive the users' browsing behaviour. Our results indicate that based on a session resumption lifetime of one day, as standard setting in many popular browsers, the average user can be tracked for up to 8 days. With a session resumption lifetime of seven days, which reflects the recommended upper limit by the draft on TLS version 1.3, even 65% of all users
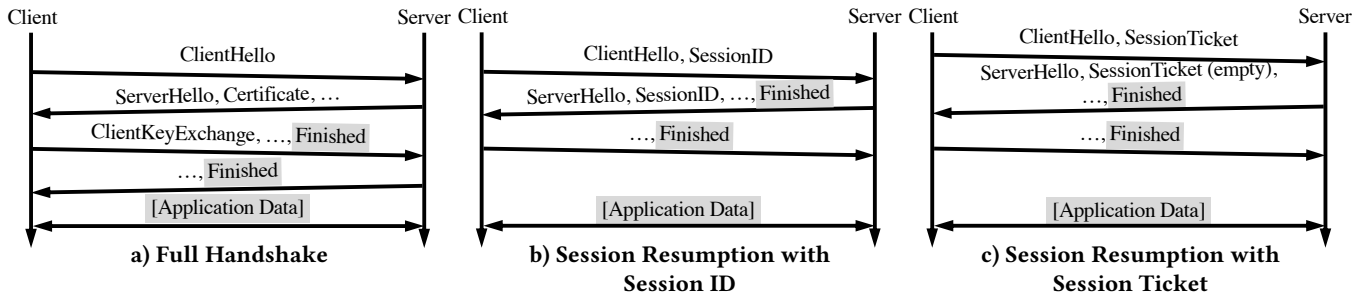
Figure 1: Handshakes in TLS 1.2 with and without session resumption, highlighting encrypted **data.**

in our dataset could be tracked permanently by at least one website in the Alexa dataset.
- We propose countermeasures that require modifying the TLS standard and the configuration of popular browsers to impede tracking based on TLS session resumption. Most effective is to disable TLS session resumption completely.

The remainder of this paper is structured as follows: Section 2 describes the background on TLS session resumption. Section 3 reviews the privacy problems arising from the usage of TLS and Section 4 explains the collection of data we used in this paper. Section 5 summarizes our major results on tracking via TLS session resumption. Countermeasures are summarized in Section 6 and related work is reviewed in Section 7. Section 8 concludes the paper.

2 BACKGROUND

In this section, we describe session identifiers (IDs) and session tickets as methods to resume a previous TLS connection for TLS up to version 1.2 [20]. Then, we regard session resumption via pre-shared keys (PSK), which is proposed in the draft of TLS 1.3 [19] and not compatible with previous resumption methods. Finally, we compare the presented mechanisms to each other.

2.1 Session ID Resumption

In this mechanism, the server assigns a random session ID during the initial handshake with the browser (client). Client and server store this session ID along with the session keys and connection states. To resume a session, the client sends the stored session ID with the first protocol message (ClientHello) to the server, as shown in Figure 1 b). If the server recognises the connection and is willing to resume the session, it replies with the same session ID to re-establish the respective session.

2.2 Session Ticket Resumption

This approach is defined in RFC 5077 [3] as an extension of the TLS protocol. In its initial ClientHello message, the client is required to express support for session ticket resumption, which will be acknowledged with an appropriate ServerHello response. After the key exchange between server and client, the server provides the client with an encrypted session ticket, which is transmitted outside of the TLS encrypted channel. This ticket contains the session keys and connection states, which are encrypted with the private *session ticket encryption key* (STEK) of the server. The client stores this session ticket along with the used session key and connection states.

Upon reconnection, the client includes the session ticket within the ClientHello message as shown in Figure 1 c). The server then decrypts the session ticket with the STEK and retrieves session key and connection state. If the server accepts the ticket, then the session can be resumed with an abbreviated handshake.

2.3 Session Resumption via Pre-Shared Keys

The draft of TLS 1.3 [19] replaces session IDs and session tickets with the concept of session resumption via pre-shared keys (PSK), which works as shown in Figure 2. After the initial handshake, the server sends a PSK identity to the client. The content of the PSK identity depends on the server and may contain a database lookup key or a self-encrypted and self-authenticated ticket. The client stores this PSK identity along with its own session keys.

In a subsequent handshake, the client provides this PSK identity within the ClientHello message to the server as seen in Figure 2 b) and 2 c). Depending on the content of the PSK identity, the server decrypts the ticket and uses the contained session keys and connection states to resume the session, or the server uses the contained lookup key to find the session keys and connection states in its own database.

2.4 Comparison of Session Resumption Mechanisms

Session resumption via PSK introduces several improvements regarding tracking. In contrast to session tickets and session IDs, the server sends the PSK identity after an initial handshake through the encrypted TLS channel. Furthermore, the server can issue multiple PSK identities at once, thus each resumption attempt uses a different PSK identity. These improvements by session resumption via PSK, protect against a correlation of single user sessions by a passive network-based observer.

RFC 5246 [20] recommends a lifetime of session IDs of less than 24 hours, the draft of TLS 1.3 [19] extends this duration to seven days. No maximum lifetime is specified for session tickets in the RFC 5077 [3]. However, servers may provide a hint to the client about their individually supported maximum lifetime. Thus, for session tickets the lifetime may exceed 24 hours.

The full handshake of TLS 1.3 (see Figure 2 a) requires one round trip less to complete in comparison to TLS 1.2 (see Figure 1 a). Thus, regarding performance, the methods of session IDs, session tickets, and 1-round-trip time (RTT) session resumption via PSK require the same number of round trips as the full handshake of TLS 1.3,
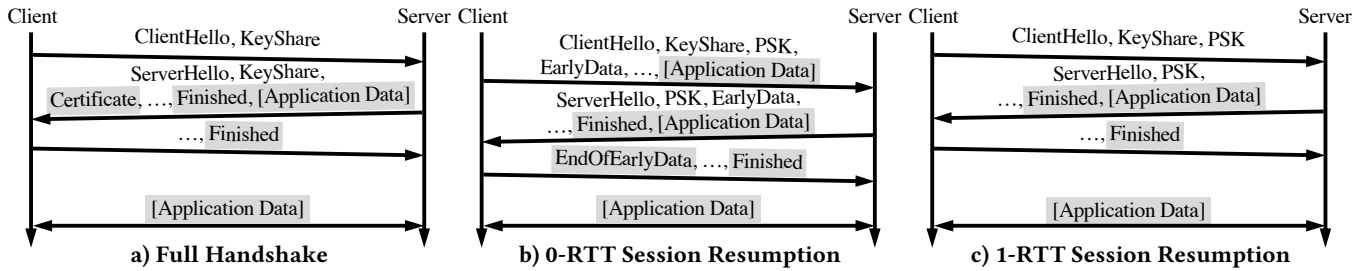
a) Full Handshake   **b) 0-RTT Session Resumption**   **c) 1-RTT Session Resumption**

Figure 2: Handshakes in TLS 1.3 with and without session resumption, highlighting `encrypted` data.

while 0-RTT session resumption via PSK can save one additional round trip.

Another distinction of the discussed resumption mechanisms is their ability to provide forward secrecy. Forward secrecy describes the property of secure communication protocols, that a compromised long-term cryptographic key does not lead to a compromise of the confidentiality of past sessions. TLS 1.2 and TLS 1.3 generally support forward secrecy by using Diffie-Hellman key establishment to negotiate a temporary, symmetric session key.

From a security perspective, TLS 1.3 improves previous TLS protocol versions by supporting forward secrecy for 1-RTT session resumption (see Figure 2 c) through Diffie-Hellman key establishment, however it is not mandatory [19]. In the case of 0-RTT resumptions, forward secrecy cannot be realised for the first application data transmitted by the client. Thus, session resumption in TLS 1.2 and partially in TLS 1.3 reduces the communication security compared to a fresh TLS session.

Another drawback of 0-RTT resumption is that servers need to implement countermeasures against replay attacks, which TLS itself guards against for other resumption mechanisms. Thus, in TLS 1.3, session resumptions with a reduced number of round trips can only be realised at the cost of reduced security guarantees.

Table 1 provides a brief overview of the differences between the TLS session resumption mechanisms.

3 PRIVACY PROBLEMS WITH TLS SESSION RESUMPTION

In this section, we describe the impact of session resumption lifetimes on users' privacy. Subsequently, we review the consequences of an unrestricted use of session resumption mechanisms with third-party online trackers.

3.1 Lifetime of Session Resumption Mechanisms

Always on and *always with* are characteristics of mobile devices such as smartphones and tablets that provide a ubiquitous access to the Internet and account for about half of all web browsing activities [23]. A web browser along with its TLS cache can remain active for multiple days in the background of mobile operating systems. Thus, very long session resumption lifetimes of several days or weeks are technically feasible.

Furthermore, the attempt of a client to resume a session by transmitting an identifier to the server leaks the identifier to the server regardless of whether the session is resumed or rejected (see
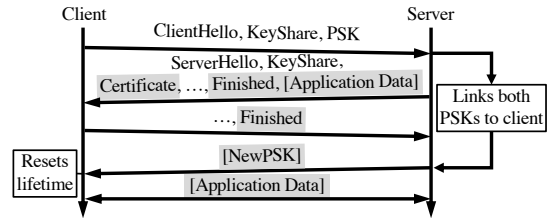

Figure 3: Prolongation attack, where the client attempts a TLS 1.3 1-RTT resumption and the server falls back to a full handshake. The server can link both PSKs to the same user, while the user's resumption lifetime is prolonged with the new PSK.

Figure 3). Thus, the identifier leakage is sufficient to correlate the initial and the newly established session to the same entity.

To further extend the capability of online tracking, a website might issue a new identifier (session ID, session ticket or PSK identity) on each revisit, and thus track a user indefinitely as long as the time between two visits does not exceed the session resumption lifetime of the user's browser. We refer to this server behaviour as *prolongation attack*. The TLS standard does not define client behaviour in a way to prevent this attack. While Figure 3 refers to TLS 1.3 1-RTT, similar attacks apply to 0-RTT and previous TLS versions.

3.2 Third-Party Tracking via Session Resumption

Third-party tracking refers to a practice, where a party, other than the targeted website, can track a user's visit. It is a widespread phenomenon on the Internet with an average of 17.7 third-party trackers per website across the Alexa Top 500 categories [2]. Google with its various hostnames is present on nearly 80% of the Alexa Top Million Sites [2, 9] and thus can gain deep insights on users' browsing behaviour.

As for tracking via session resumption, third-party trackers can recognise users based on the identifier, which they present to resume a previous TLS session. Thus, the tracker can link multiple observed visits of the user across sites, where the tracker is included as a third-party. However, to distinguish the various first-party sites a user visited, the tracker requires an identifier such as a HTTP referrer or a custom URL per first-party.

Table 1: Comparison of TLS session resumption mechanisms

	Session ID	Session Tickets	0-RTT via PSK	1-RTT via PSK
Server stores its own secret TLS state	yes	no	optional	optional
Number of RTT compared to full handshake	-1 RTT	-1 RTT	-1 RTT	identical
Initial handshake contains unencrypted identifier	yes	yes	no	no
Identifier reuse for multiple connections	yes	yes	should not	should not
Forward secrecy	no	no	no	optional
Uses one key for sessions of multiple users	no	yes	optional	optional
Recommended limit of the resumption lifetime	24h [20]	>24h [3]	7 days [19]	7 days [19]

4 DATA COLLECTION

In this section, we describe our various data sources which we use to determine the feasibility of online tracking based on TLS session resumption. For our empirical analysis, we accumulated the TLS configuration of popular online services and browsers. Furthermore, we investigate web browsing patterns of users based on a DNS traffic data set.

4.1 Alexa Top Million Data Set

To get an estimate on the usage of TLS session resumption in the web and to conduct a qualitative analysis of the used session resumption mechanisms we measured the HTTPS behaviour of the Alexa Top Million Sites [9]. Over a period of 31-days in March and April of 2018, we connected daily to each and every site in the Alexa Top Million on TCP port 443 using the OpenSSL Toolkit [18] and recorded the handshake behaviour.

To probe the configured session resumption lifetime for session IDs and session tickets used by the Alexa Top Hundred Thousand, we revisited each website periodically in intervals of five minutes or less, each time presenting the identical session ID or session ticket, respectively.

To measure which sites share their session ID cache or STEK with other online services, we saved the TLS connection state of a session with one site and tried to pairwise reconnect this state to the other websites in the sample. Subsequently, we created groups of sites that allow mutual session resumption. This pairwise evaluation causes quadratic cost, therefore we reduced our sample to the Alexa Top Thousand Sites.

We conducted all scans from our university campus and followed best practices of active scanning [1].

4.2 Browser Measurements

To assess the browser behaviour in regard to session resumption we used a sample of 48 browsers for mobile and desktop platforms as shown in Table 2. This sample includes the most popular browsers with respect to their global market share [14, 16, 22], which were publicly accessible for either iOS, Android, and/or desktop operating systems. Besides the most popular browsers, we included Tor Browser, Orbot, Brave, Cliqz, JonDoBrowser, and Ghostery Privacy Browser as explicitly privacy-friendly browsers to our sample.

To gather the configured maximum resumption lifetimes of each browser and for each different resumption mechanisms, we used a test website with a custom JavaScript probe. We attempted to resume a session after varying intervals of up to 24 hours since the initial handshake. On each connection attempt, the server checks and records if the initially established session resumption identifier is transmitted. We tried only one session resumption per initial handshake to avoid potential side effects like a prolongation of the browser's resumption lifetime.

To test if browsers enable third-parties to link user activities on different websites as described in Section 3.2, we used a testbed as illustrated in Figure 4. The browser consecutively retrieves the websites A and B, which include the same third-party T. We observe, whether a session resumption with T is possible from the context of different first parties A and B.
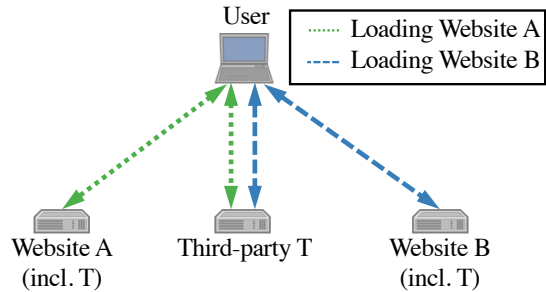
Figure 4: Testbed to measure browser behaviour in regard of third-party tracking.

4.3 DNS Data Set

To estimate the impact of the presented prolongation attack (see Section 3.1) on the tracking period of a user, we evaluate real-world browsing patterns. Furthermore, these browsing patterns allow us to approximate the ratio of resumed revisits for a given session resumption lifetime.

We obtained the data set used in [5, 6, 11], which contains the pseudonymized DNS traffic logs of 3862 students over a period of two months between April 30, 2010 and June 29, 2010. The data set originates from the student housing network at the University of Regensburg.

Our sample of DNS logs contains only the fraction of Internet traffic that originates from the students' unique and static IP address accessible within their room. Note that this sample might not cover the whole traffic of all users and thus our conclusions might only describe a lower boundary in regard to user tracking via TLS session resumption. A descriptive statistic of the used DNS data set can be found in [5].
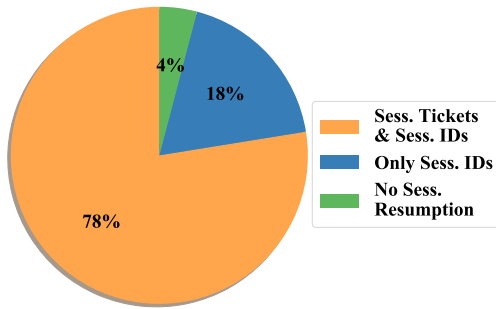
Figure 5: Supported session resumption mechanisms for TLS-enabled sites in Alexa Top Million on 15th April 2018.

We restricted our evaluations to a pseudonymized DNS data set to address ethical concerns. Besides a pseudonym for the source IP address, we also pseudonymized target hostnames on a per user basis. This approach prevents background knowledge attacks that use knowledge about the browsing history of another user within this data set.

Consequently, each record in our data set consists of a pseudonym for the source IP address, the query time, a pseudonym for the target hostname, and query type. For our analysis we considered only DNS queries of regular name resolution for IPv4 addresses as well as IPv6 addresses.

5 EVALUATION

To evaluate the feasibility of tracking based on TLS session resumption, we analyse in this section the configuration of servers and browsers and to which extent they restrict the session resumption mechanisms. Afterwards, we investigate real-world browsing patterns and check whether the technical restrictions of browsers are suitable to protect users' privacy against online tracking services.

5.1 Evaluation of Server Configurations

The feasibility of TLS session resumption as a tracking mechanism depends considerably on the configuration of the server. In this section, we investigate the adaptation of TLS session resumption mechanisms, real-world configurations for the session resumption lifetime and security-related configuration issues.

5.1.1 Adoption of TLS Session Resumption Mechanisms. Figure 5 shows the support of TLS-enabled Alexa Top Million Sites for session resumption based on IDs or tickets. This measurement from the 15th April 2018 does not include session resumption via PSK because TLS 1.3 was still a draft at that point. In total, we found 691 280 sites among the Alexa Top Million that supported TLS. 95.9% of these sites do either support session IDs or both IDs and tickets. The remaining 28 236 sites do not support TLS session resumption by, for example, providing an empty string as an ID which does not allow to resume a session.

With 536 088 websites, 77.6% of all TLS-enabled Alexa Top Million Sites do also support session tickets. Note that, if a client-server pair supports both session resumption mechanisms, then TLS session tickets will be the preferred resumption mechanism [3].

5.1.2 Lifetime of Session Resumption Mechanisms. The server can include a *lifetime hint* along with the session ticket or PSK
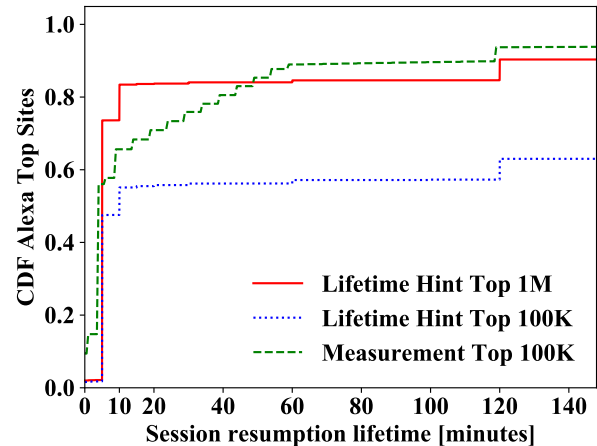

Figure 6: Cumulative distribution of Alexa Top Sites over short hinted and measured lifetimes of session tickets.

identity. If the browser respects lifetime hints, it will only try to resume previous sessions within this hinted lifetime.

The lifetime hints of TLS session tickets for the Alexa Top Million set (solid, red line) and Alexa Top Hundred Thousand set (dotted, blue line) are shown as cumulative distribution in Figure 6. This plot is normalised to the total number of obtained tickets, which were 535 306 and 56 407 for the respective sets on the 24th March 2018. With 46% and 71%, a lifetime hint of five minutes is the most popular configuration within the Alexa Top Hundred Thousand and Alexa Top Million respectively.

In both sets, around 2% of all sites use a lifetime hint of zero seconds. We interpret this configuration as erroneous because this effectively prevents session resumption. Instead, server operators should deactivate the session ticket extension to save resources for both server and client if they choose not to support session tickets.

We observe, that more than 80% of TLS session ticket enabled sites within the Alexa Top Million chose lifetime hints of less than or equal to ten minutes. However, around 10% of the remaining sites use lifetime hints larger than 24 hours. Google and Facebook as market leaders in behavioural advertising show particularly large hinted session resumption lifetimes. Facebook's lifetime hint of 48 hours is above the 99.99% percentile of all session ticket hints that we collected in our scans. Google and various of its domains are configured to a ticket lifetime of 28 hours, which is above the 97, 13% percentile in the Alexa Top Million Sites (see Figure 7).

To validate these results about lifetime hints, we measured the maximum session resumption lifetimes for session IDs and session tickets. We limited our sample size to the Alexa Top Hundred Thousand Sites for practicality reasons. Figure 6 and 8 show these results as green, dashed line for tickets and IDs, respectively. We find, that around 10% of the sites do not allow a resumption with session tickets immediately after the initial visit, while, for session IDs these are 23% of the websites. We observe in both figures, that approximately 40% of the sites support maximum resumption lifetimes above five minutes. More than 20% of websites within the Alexa Top Hundred Thousand that support session IDs allow resumptions after more than two hours.
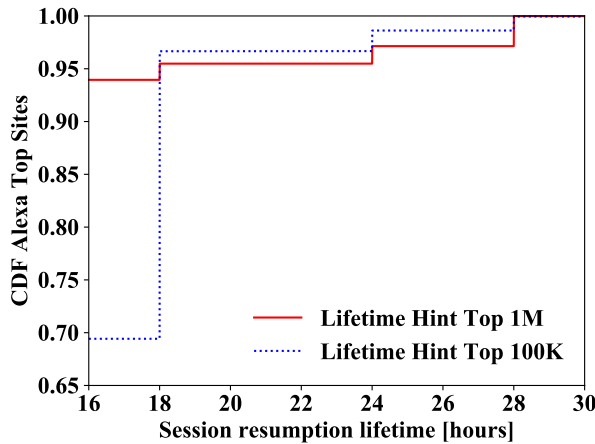
Figure 7: Cumulative distribution of Alexa Top Sites over long hinted lifetimes of session resumption tickets.
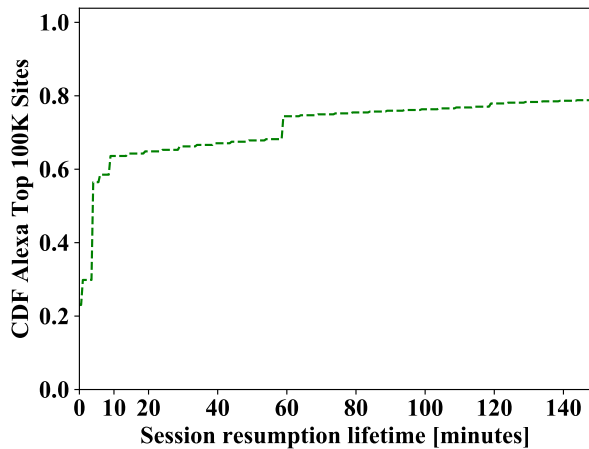

Figure 8: Cumulative distribution over the measured session resumption lifetime with session IDs.

As can be seen in Figure 6, our measurement of the maximum resumption lifetime deviates within a range of 10 to 55 minutes from the blue, dotted plot of the corresponding lifetime hints. So far, we are not able to explain this zig-zag shape of the green plot.

5.1.3 Security Issues of TLS Server Configurations. In this section, we empirically measure TLS state sharing, where multiple sites share their cryptographic secrets for user sessions. Furthermore, we evaluate the lifetime of *Session Ticket Encryption Keys* (STEK). These two measurements allow us to assess the vulnerable period and the number of vulnerable websites in case of a compromise of a site's STEK.

A passive, network-based observer in possession of a compromised STEK can decrypt the initial and the resumed sessions in TLS 1.2 due to the missing forward secrecy for session resumption (see Section 2.4). This issue becomes more severe as STEKs are shared across multiple hostnames, which increases the number of potentially affected sessions as well as the attack surface.
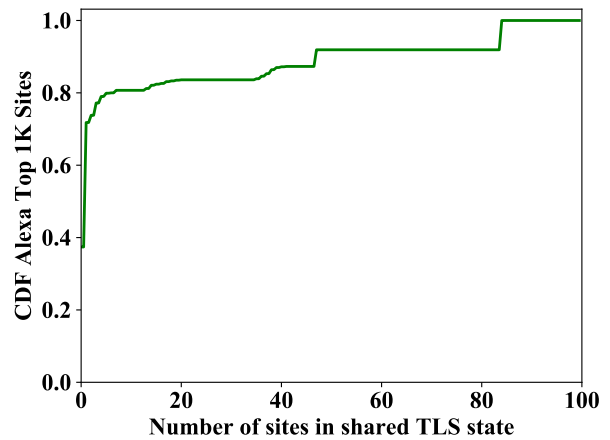

Figure 9: Cumulative distribution over the size of shared TLS states among sites within the Alexa Top Thousand.

We measured the shared TLS state by attempting to resume sessions of website A, where A is within Alexa Top Thousand, on every other site of the Alexa Top Thousand.

Figure 9 shows that 72% of the Alexa Top Thousand do either not support session resumption or do not share their TLS state with any other site of the Alexa Top Thousand. However, around 16% of websites share their TLS state with at least 25 other sites of the Alexa Top Thousand. The largest shared TLS state within the Alexa Top Thousand counts 84 sites and belongs to Google.

RFC 5077 [3] recommends a scheme for the construction of TLS session tickets, which includes an identifier for the respective STEK. With daily scans of the Alex Top Million Sites for 31-days, we obtained a sample of over 16 million session tickets. In the next step, we assigned each site their corresponding set of STEK identifiers.

Figure 10 shows, that 73% of the TLS session ticket enabled Alexa Top Million Sites to change their STEK within three days. However, 10% of the websites with support for session tickets did not change their STEK within two weeks. Consequently, a compromised STEK from such a website allows a network-based observer to decrypt all sessions that use session tickets with this site for at least two weeks.

Note that TLS implementations might deviate from the recommended construction scheme for session tickets [3]. Our setup expects session tickets to follow the recommended structure, deviating websites are therefore falsely classified as exchanging STEKs with every new ticket. Thus, Figure 10 shows only a lower boundary of websites that change their STEK less frequently than every 24 hours.

## 5.2 Evaluation of Browser Configurations

In this section, we investigate the default configuration of popular web browsers to determine the feasibility of user tracking via TLS session resumption. First, we report on the session resumption lifetime of browsers for session IDs and tickets. Then, we evaluate the capability of third-parties to track users across different first-party websites.

Table 2: TLS session resumption configuration of browsers

Plt.	Browser	Lifetime for Sess. ID	Lifetime for Sess. Ticket	Third-party tracking
Desktop	360 Sec. Bro. v9.1	600 min	540 min	blocked
	Amigo v61.0	30 min	30 min	viable
	Brave	30 min	30 min	viable
	Chrome v66.0	60 min	30 min	viable
	Cliqz	1 day	10 min	viable
	Coc Coc v68.4	30 min	30 min	viable
	Comodo Dra. v63	30 min	30 min	viable
	Firefox v59.0	1 day	1 day	viable
	Internet Expl. v11	600 min	600 min	viable
	JonDoBrowser	-	-	blocked
	Konqueror v5.0	30 min	30 min	blocked
	K-Meleon v75.1	1 day	1 day	viable
	Lunascape v6.15	600 min	540 min	viable
	Maxthon v5.2	30 min	30 min	viable
	Microsoft Edge v41	600 min	600 min	blocked
	Opera v52	30 min	30 min	viable
	Pale Moon v27	1 day	10 min	viable
	QQ Browser v10	30 min	30 min	viable
	QupZilla v2.2.6	30 min	30 min	viable
	Safari v11.1	1 day	1 day	viable
	SeaMonkey v2.49	1 day	1 day	viable
	Sleipnir v6.2	30 min	30 min	blocked
	Sogou Expl. v8	30 min	30 min	viable
	SRWare Iron v65	30 min	30 min	viable
	Tor Browser	-	-	blocked
	UC Browser v7.0	30 min	30 min	viable
	Vivaldi v1.14	30 min	30 min	viable
Android	Amigo v1.10.187	60 min	30 min	viable
	Android Bro. v7.1.2	30 min	30 min	viable
	Brave v1.0.42	60 min	30 min	viable
	Cheetah Bro. v5.22	60 min	30 min	viable
	Chrome v61.0	30 min	60 min	viable
	Cliqz v1.6.2	60 min	30 min	viable
	Firefox v56.0	1 day	20 min	viable
	Ghostery Priv. v1.3	30 min	30 min	viable
	Maxthon v4.5.10	30 min	30 min	viable
	Opera Mini v30.0	15 sec	15 sec	viable
	Orbot v16.0.0	-	-	blocked
	QQ Bro. v1.2.0	1 day	1 day	viable
	Samsung Intern. v6	50 min	50 min	viable
	Sleipnir v 3.5.7	60 min	30 min	viable
	SRWare Iron v61.0	60 min	30 min	viable
	UC Browser v12.0	1 day	30 min	viable
	Yandex Bro. v18.1	50 min	30 min	viable
iOS 11	Chrome v62.0	120 min	-	viable
	Firefox v9.2	120 min	-	viable
	Opera v16.0	120 min	-	viable
	Safari v11.0	120 min	-	viable
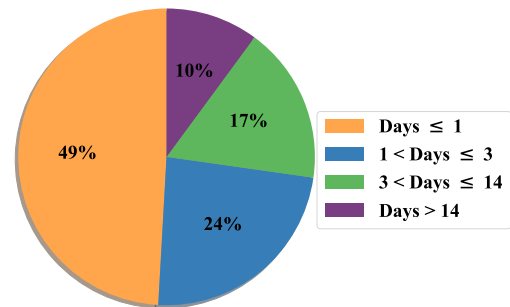
Figure 10: Distribution of Alexa Top Million Sites over the period for which they encrypt session tickets with the identical cryptographic key.

5.2.1 Lifetime of Session Resumptions Mechanisms. The results of measuring the session resumption lifetime are displayed in Table 2. We note that only three browsers from the privacy-friendly group do not support TLS session resumption, the remaining 45 browsers all support at least one resumption mechanisms.

We observe, that two-thirds of all tested browsers allow only for a session resumption lifetime of up to 60 minutes for both resumption mechanisms. Our measurement period is limited to 24 hours, therefore a stated one-day period might, in fact, be longer.

Some browsers such as Chrome, Opera, Firefox, and Safari were tested on multiple platforms. For these browsers, we find that the configuration of the session resumption lifetime is not consistent across the investigated platforms. For example, the desktop version of Safari is configured to a session resumption lifetime of 24 hours, while the iOS version only supports resumptions for session IDs of up to 120 minutes.

Furthermore, Table 2 shows that the lifetime for both investigated resumption mechanisms differ for Cliqz (desktop), Pale Moon (desktop), Firefox (Android), and UC Browser (Android) by more than 23 hours. For browsers on iOS, we observe a homogeneous behaviour in our measurements, which can be explained by Apple's requirement that all browser apps in the App Store use Apple's WebKit framework as rendering engine [8].

5.2.2 Third-party Tracking. We analysed whether the default configuration of browsers allows resuming TLS sessions in the context of different first-party websites. Table 2 shows, that only the desktop browsers 360 Secure Browser, Konqueror, Microsoft Edge, and Sleipnir restrict the session resumption support for third-parties. Note, that the privacy-friendly browsers JonDoBrowser, Tor Browser, and Orbot do not support TLS session resumption mechanisms and thus also prevent third-party tracking in this regard.

Our results show, that third-party tracking via TLS session resumption is feasible for the large majority of investigated popular browsers. However, our results about the session resumption lifetime (see Section 5.2.1) indicate the session resumption lifetime is limited within the majority of investigated browsers. An explanation of our results regarding third-party tracking could be that browser vendors are mostly unaware of third-party tracking via TLS session resumption.

5.3 Evaluation of Real-World User Traffic

In this section, we use real-world data from a DNS traffic data set to assess the impact of different session resumption lifetimes on the achievable length of tracking periods as well as the share of permanently trackable users, respectively using the prolongation attack. Subsequently, we analyse the impact of a chosen session resumption lifetime on the expectable frequency of session resumptions for revisits.

5.3.1 Assumptions and Limitations. Lacking access to a comprehensive data set of actual web browsing activity, we resort to a DNS traffic data set as described in Section 4.3. For our evaluation, we assume that all DNS name resolution queries in the data set led to a TLS session. As discussed in Section 3, we reason that *always on* devices such as smartphones or tablets can achieve runtime durations for browsers and their TLS cache of several days for average users. Therefore, we assume that users would not have cleared the TLS cache of their browser within the tracking period. Note, that the following evaluations only approximate actual website visits since we cannot account for DNS caching effects.

5.3.2 Longest Consecutive Tracking Period. Tracking mechanisms become more capable the longer it is possible to link user behaviour to a known entity. Recall that by using the prolongation attack as described in Section 3.1, the period in which user activities can be linked via session resumption identifiers can be extended beyond a single session resumption lifetime. The extent to which this prolongation is possible depends on the time between consecutive visits of a website.

Definition 5.1. In the context of session resumption we use the term *consecutive tracking period* to refer to a sequence of visits $v_1, \ldots, v_n$ to an online service where no interval between two visits exceeds the given session resumption lifetime l. More formally, we define the predicate P as

$$P(v_1, \ldots, v_n, l) \Leftrightarrow |t_i - t_{i+1}| \le l, \forall i \in \{1, \ldots, n-1\},$$

where t_i denotes the time of visit v_i.

Definition 5.2. We further define the *longest consecutive tracking period lctp* of a sequence of visits $v_1, \ldots, v_n$ given a session resumption lifetime l as the length of the longest subsequence of visits that still fulfils the consecutive tracking period property, or more formally as

$$lctp(v, l) \mapsto \max_{i,j \in \{1, \ldots, n\}, i < j} |t_i - t_j| [P(v_i, \ldots, v_j, l)],$$

where t_i again denotes the time of visit v_i.

Figure 11 shows the longest consecutive tracking period as a function of the session resumption lifetime determined from said DNS data set with the same assumption as stated above. Each data point depicts the maximum tracking period overall websites of the respective user, or, more formally,

$$lctp_{\max}(u, l) \mapsto \max_{s \in S} lctp(v_{u,s}, l),$$

where S denotes the set of all visited sites and $v_{u,s}$ the sequence of visits by user u to site s. We plotted $lctp_{\max}$ for the median user (blue, dotted line) and the average user (green, dashed line) of our data set within total 3862 users.
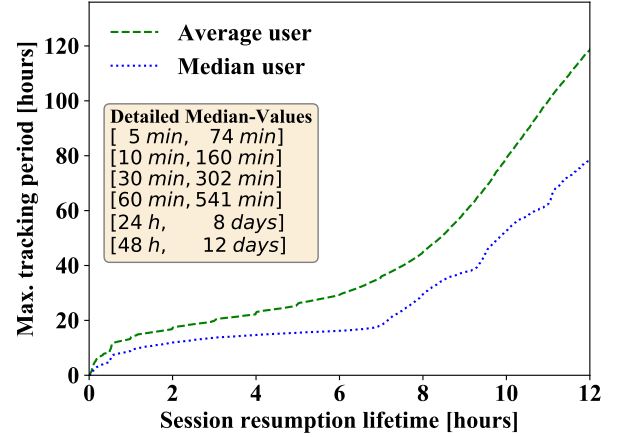
Figure 11: The longest consecutive tracking period per user plotted as a function of the session resumption lifetime for the prolongation attack scenario.

We observe, that the length of the longest consecutive tracking period differs widely among users. While some can be tracked with a session resumption lifetime of 12 hours throughout the whole 61-day period of data collection, the median user can only be tracked for a period of about three days.

We also note that the gradient of these plots exhibits stronger increments for session resumption lifetimes shorter than 20 minutes or longer than seven hours. It seems as if these seven hours were chosen to overcome a user's sleeping phase with reduced online activity because for such longer lifetimes, the tracking period becomes longer than one day.

5.3.3 Share of Permanently Trackable User. Next, we evaluate the share of those users in our DNS data set that can be monitored throughout the 61-days sample period. We uphold the assumptions as stated above.

Definition 5.3. We define a user as permanently trackable by a given site, if the visits to this site fulfil the consecutive tracking period property and the time between the first (resp. last) visit and the start (resp. end) of the sample is less than or equal to the given session resumption lifetime. With the last restriction we ensure, that the user tracking can possibly continue beyond the boundaries of our sample period.

As can be seen in Figure 12, with a session resumption lifetime of seven days, 65% of all users within our data set can be permanently tracked by at least one website. By limiting the session resumption lifetime to 24 hours, the share of permanently trackable user declines to only 1.3%.

5.3.4 Ratio of Resumed Revisits. In the interest of providing empirical data to quantify performance gains from resumed session, we investigate the impact of different session resumption lifetimes on the ratio of resumed revisits. We denote revisits which happen within the session resumption lifetime of the previous visit as *resumed revisit*.

Definition 5.4. We define as *resumption ratio* of a sequence of visits $v_1, \ldots, v_n$ by the same user to an online service given a
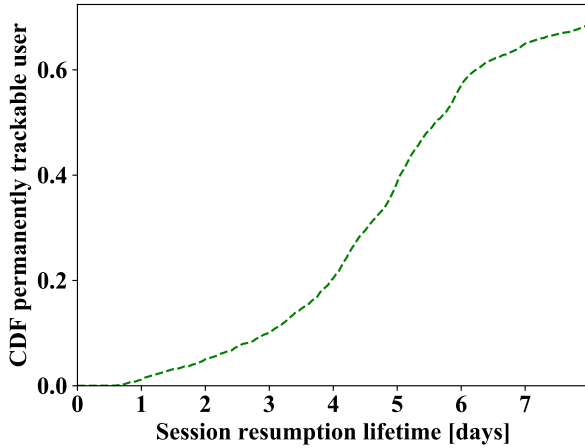
Figure 12: Cumulative distribution of permanently trackable user over session resumption lifetime for the prolongation attack scenario.
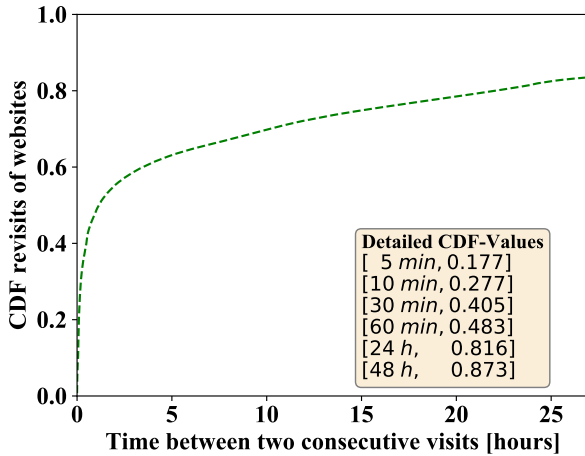

Figure 13: Cumulative distribution of revisits of websites plotted over the time between two visits by the same user.

session resumption lifetime l as

$$rr(v, l) \mapsto \frac{|\{i \in \{1, \ldots, n-1\} \wedge |t_i - t_{i+1}| \leq l\}|}{n-1},$$

where t_i denotes the time of visit v_i.

Figure 13 shows the cumulative distribution of website revisits as a function of the interval between two visits by the same user.

We observe a large share of 17.7% of all revisits can be resumed with a session resumption lifetime of five minutes. It can be further observed, that the probability of a revisit decreases continuously with the time since the last visit. Moreover, we find that about half of all revisits occur during the first hour and 95.2% within the first week.

6 COUNTERMEASURES

Ultimately, this leads us to a discussion of potential countermeasures. **A complete protection against tracking via TLS session**

resumption is achieved by deactivating this feature as it is practised by the privacy-friendly JonDoBrowser and Tor Browser. A strict deactivation consequently excludes a browser from performance gains, such as the reduced number of round trips of TLS 1.3 0-RTT session resumption.

RFCs on session resumption mechanisms such as TLS 1.3 [19] should be extended in a way, that they exclude a possible prolongation of the lifetime of session resumption mechanisms (see Section 3.1). We recommend that a server-initiated renewal of a session identifier must not lead to a prolongation of client-side expiry dates. Instead, **a client must stick to the expiry date of the initial session identifier**. This protects against the prolongation attack (see Section 3.1).

The **recommended upper limit of the session resumption lifetime in TLS 1.3 [19] of seven days should be reduced** to hinder tracking based on this mechanism. We propose an upper lifetime limit of ten minutes based on our empirical observations. We note, that more than 80% of the Alexa Top Million Sites restrict the session resumption lifetime to less or equal to ten minutes by their own choice and 27, 7% of all revisits of a site occur during this period. Furthermore, the average visit duration of popular websites is of the order of ten minutes [13], thus this lifetime limit hinders the correlation of multiple page visits by the same user.

Browser vendors should address the issue of third-party tracking via TLS session resumption, either by deactivating session resumption for third-parties or by allowing only session resumptions to third-parties if the first party site is identical.

Furthermore, considering that TLS 1.3 1-RTT session resumption does not lead to a reduced number of round trips compared to the full handshake, the temporal gains of this mechanism have its origin in the reduced computational complexity of the abbreviated handshake and are rather small. For a latency of 133 ms, which is in the range of 3G mobile network connections, we approximate the temporal gain of TLS 1.3 1-RTT to be in an order of 1% compared to the full handshake. Thus, **due to the low temporal gains, it seems acceptable to deactivate TLS 1.3 1-RTT session resumption** for privacy reasons.

TLS 1.3 0-RTT provides higher temporal gains by reducing the number of required round trips. However, it is not a replacement of TLS 1.3 1-RTT due to its reduced security guarantees (see Section 2.4). Thus, by **limiting the support for session resumption to TLS 1.3 0-RTT** the number of resumed sessions should decrease, while the temporal gains are rather high.

Finally, we reported in Section 5.1.3 that at least 10% of the Alexa Top Million Sites use their STEK for a period of at least two weeks. Since the STEK makes it possible to decrypt all sessions that use session tickets from such a site, we suggest that **a minimum STEK change rate should be standardised** to reduce this vulnerable period.

7 RELATED WORK

The impact of TLS on the privacy of its users has already been a focus of previous research. Wachs et al. [24] show in their work, that the TLS *client certificate authentication* transmits unique client certificates in plaintext and thus allows a passive eavesdropper to re-identify and track users. However, websites do not widely

use TLS *client certificate authentication* due to its complexity [24]. Thus, it is less feasible to be employed by online tracking service to observe users' browsing behaviour.

Empirical research by Husák et al. [7] investigated the feasibility of monitoring TLS handshakes to fingerprint and identify clients. They found, that especially the supported cipher suite lists vary among various client applications and their versions. This allows them, to infer the client application with a certain precision based on the observed cipher suit list. While this result may be beneficial to network security monitoring to detect anomalies, it is not suitable for commercial user tracking because of the few observed TLS client configurations during the handshake, do not allow to uniquely distinguish users.

Springall et al. [21] investigated security problems of TLS session resumption such as STEK lifetime, forward secrecy and TLS state sharing based on measurements of the Alexa Top Million Sites.

However, to the best of our knowledge, the feasibility of user tracking based on TLS session resumption has not been investigated so far. The privacy implications of session resumption have only been of concern to software projects. The Tor Browser disabled session resumption due to privacy considerations [17]. Moreover, the chromium project lists session resumption mechanisms as capable to allow client identification [10].

8 CONCLUSION

In this paper, we studied the feasibility of user tracking via TLS session resumption. For that, we evaluated the configuration of popular browsers and online services as well as behavioural user patterns.

Our results indicate, that most major browsers support TLS session resumption. Only three privacy-friendly browsers deactivated this feature. Almost all investigated browsers support tracking periods of at least 30 minutes based on TLS session resumption. We even observed several browsers with session resumption lifetimes of at least 24 hours, e.g., Firefox or Safari. Additionally, we investigated whether browsers protect against third-party tracking based on TLS session resumption, which allows to re-identify users across different websites in which the third-parties are embedded as well. Our results indicate that the majority of tested browsers in their standard configuration does not protect users against such third-party tracking.

In addition to the browsers, we also checked the TLS configurations of web servers delivering the Alexa Top Million Sites. We found that the majority of these websites use resumption lifetimes of up to ten minutes, which might be an indication that tracking based on TLS session resumption is not widely applied yet. However, we also observe that especially big players like Google and Facebook use exceptionally long session resumption lifetimes of 28 and 48 hours, respectively. Nevertheless, as longer session resumption lifetimes decrease the load on web servers significantly, this is not a clear indication that the tracking technique is used by them.

As the main contribution of this paper, we present a *prolongation attack* against the TLS standard, which allows to extend the tracking period beyond the session resumption lifetime. Based on a real-world data set on DNS traffic from 2010 with 3862 users we found that with a session resumption lifetime of 24 hours there is one

website in the dataset that would be able to track the average user over a period of eight days. The draft of TLS 1.3 [19] proposes an upper session resumption lifetime of seven days. We analysed that such a configuration of session resumption mechanisms would make 65% of all users permanently trackable by at least one single website from our data set. However, compared to today the data set contains fewer web requests coming from mobile devices. They render tracking based on TLS session resumption easier, because of their always on-property and as they amplified the frequency users access certain (major) websites.

To mitigate the presented privacy problems, we propose countermeasures to the TLS standard and to browser vendors. The most effective technique is to disable TLS session resumption in browsers completely.

In summary, we hope that our work leads to greater awareness of the privacy risks coming from TLS session resumption and fosters further research on this topic.

REFERENCES

[1] Z. Durumeric, E. Wustrow, and J A. Halderman. 2013. ZMap: Fast Internet-wide Scanning and Its Security Applications.. In *USENIX Security Symposium*, Vol. 8. 47–53.
[2] S. Englehardt and A. Narayanan. 2016. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1388–1401.
[3] P. Eronen, H. Tschofenig, H. Zhou, and J. A. Salowey. 2008. Transport Layer Security (TLS) Session Resumption without Server-Side State. RFC 5077. (Jan. 2008). https://doi.org/10.17487/RFC5077
[4] A. Gómez-Boix, P. Laperdrix, and B. Baudry. 2018. Hiding in the Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale. In *WWW 2018: The 2018 Web Conference*.
[5] D. Herrmann, C. Banse, and H. Federrath. 2013. Behavior-based tracking: Exploiting characteristic patterns in DNS traffic. *Computers & Security* 39 (2013), 17–33.
[6] D. Herrmann, M. Kirchler, J. Lindemann, and M. Kloft. 2016. Behavior-based tracking of Internet users with semi-supervised learning. In *Privacy, Security and Trust (PST), 2016 14th Annual Conference on*. IEEE, 596–599.
[7] M. Husák, M. Čermák, T. Jirsík, and P. Čeleda. 2016. HTTPS traffic analysis and client identification using passive SSL/TLS fingerprinting. *EURASIP Journal on Information Security* 2016, 1 (2016), 6.
[8] Apple Inc. 2018. App Store Review Guidelines. (2018). Retrieved May 15, 2018 from https://developer.apple.com/app-store/review/guidelines/
[9] Alexa Internet Inc. 2018. Alexa Top 1,000,000 Sites. (2018). Retrieved March 15, 2018 from http://s3.amazonaws.com/alexa-static/top-1m.csv.zip
[10] A. Janc and M. Zalewski. 2014. Technical analysis of client identification mechanisms. (2014). Retrieved March 15, 2018 from https://www.chromium.org/Home/chromium-security/client-identification-mechanisms
[11] M. Kirchler, D. Herrmann, J. Lindemann, and M. Kloft. 2016. Tracked without a trace: linking sessions of users by unsupervised learning of patterns in their DNS traffic. In *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*. ACM, 23–34.
[12] P. Laperdrix, W. Rudametkin, and B. Baudry. 2016. Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints. In *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 878–894.
[13] SimilarWeb LTD. 2018. Top Websites Ranking. (2018). Retrieved May 5, 2018 from https://www.similarweb.com/top-websites
[14] NetMarketShare. 2018. Browser Market Share. (2018). Retrieved March 15, 2018 from https://netmarketshare.com/browser-market-share.aspx
[15] P. Papadopoulos, N. Kourtellis, and E. P. Markatos. 2018. Cookie Synchronization: Everything You Always Wanted to Know But Were Afraid to Ask. *arXiv preprint arXiv:1805.10505* (2018).
[16] J. Papenbrock. 2018. Aktuelle Marktanteile der Browser. (2018). Retrieved March 15, 2018 from https://www.browser-statistik.de/statistiken/
[17] M. Perry. 2012. Disable TLS Session resumption and Session IDs. (2012). Retrieved March 15, 2018 from https://trac.torproject.org/projects/tor/ticket/4099
[18] The OpenSSL Project. 2018. OpenSSL Cryptography and SSL/TLS Toolkit. (2018). Retrieved March 15, 2018 from https://www.openssl.org/
[19] Eric Rescorla. 2018. *The Transport Layer Security (TLS) Protocol Version 1.3*. Internet-Draft TLS13. Internet Engineering Task Force. https://datatracker.ietf.

org/doc/html/draft-ietf-tls-tls13-28 Work in Progress.
[20] E. Rescorla and T. Dierks. 2008. The Transport Layer Security (TLS) Protocol
     Version 1.2. RFC 5246. (Aug. 2008). https://doi.org/10.17487/RFC5246
[21] D. Springall, Z. Durumeric, and J A. Halderman. 2016. Measuring the security
     harm of TLS crypto shortcuts. In *Proceedings of the 2016 Internet Measurement
     Conference*. ACM, 33–47.
[22] StatCounter. 2018. Browser Market Share Worldwide. (2018). Retrieved March
     15, 2018 from http://gs.statcounter.com/browser-market-share
[23] StatCounter. 2018. Desktop vs Mobile vs Tablet Market Share Worldwide. (2018).
     Retrieved March 15, 2018 from gs.statcounter.com/platform-market-share/
     desktop-mobile-tablet/worldwide
[24] M. Wachs, Q. Scheitle, and G. Carle. 2017. Push away your privacy: Precise
     user tracking based on TLS client certificate authentication. In *Network Traffic
     Measurement and Analysis Conference (TMA), 2017*. IEEE, 1–9.