Single Sign-On Privacy: We Still Know What You Did Last Summer

Maximilian Westers
Heilbronn University of Applied Sciences
maximilian.westers@hs-heilbronn.de
ORCID: 0000-0003-1666-906X

Andreas Mayer
Heilbronn University of Applied Sciences
andreas.mayer@hs-heilbronn.de
ORCID: 0000-0001-7055-8876

Louis Jannett
Ruhr University Bochum
louis.jannett@rub.de
ORCID: 0000-0001-5448-5929

***Abstract*—Today, Single Sign-On (SSO) is omnipresent on the Internet. Every day, millions of users utilize SSO protocols such as OAuth 2.0 (OAuth) or OpenID Connect 1.0 (OIDC). These protocols allow users to log in to multiple websites or services, called Relying Partys (RPs), using their accounts from major Identity Providers (IdPs) such as Apple, Facebook, and Google. Consequently, these IdPs gain the ability to track their users across the Internet. In return, RPs gain access to an enriched set of the user's personal data stored at the IdP, including names, email addresses, and profile pictures.**

In this paper, we present three novel SSO privacy leaks found in the wild. Contrary to prior work, our leaks occur *automatically* as soon as the user visits the RP, without their consent or awareness, in a non-transparent manner. To prove their prevalence, we conducted a large-scale study on the Tranco top 1M websites. Our measurement shows that 10,931 RPs automatically leak the user's identity, the currently visited RP, and other metadata (e.g., time of access) to the IdPs (*partial leak*). Additionally, 2,947 RPs silently deanonymize users, logging them into their accounts without their awareness (*full leak*). Even worse, 6 RPs leak the user's identity to third parties (*escalated leak*). Besides 4 major IdPs, including Facebook and Google, we identified privacy leaks affecting 1,032 additional, less-popular IdPs. Conversely, 7 IdPs, including Apple and Github, are exemplary in avoiding these leaks.

To protect users, we present our browser extension called SSO PRIVACY GUARD. We demonstrate its effectiveness in preventing all the identified leaks. Furthermore, we discuss if and how emerging initiatives by major browser vendors related to tracking prevention can also improve privacy in the SSO ecosystem. To promote reproducibility, we publicly release the source code and all artifacts, and we plan to release SSO PRIVACY GUARD in the official Chrome and Firefox extension stores.

***Index Terms*—Single Sign-On, OAuth, OpenID Connect, Social Login, Privacy, Web Measurement, GDPR**

1. Introduction

Single Sign-On. SSO has become an indispensable part of user authentication on the Internet. It allows users to log in to multiple websites and services, known as RPs, by using their central account at an IdP. Major IdPs like Apple, Facebook, and Google provide their SSO services to billions of users globally. Today, the most popular SSO protocols are OAuth and OIDC. These protocols are widely adopted by consumer-facing websites for enabling social logins, such as "Sign in with Facebook" [2].

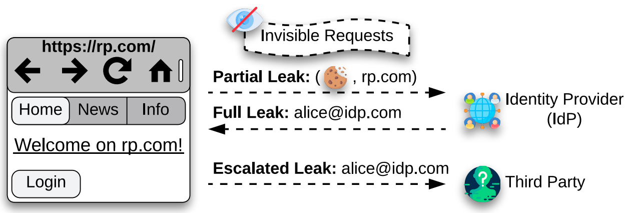


Figure 1: Overview of SSO Privacy Leaks. Partial leaks allow IdPs to track their users across RPs. Full leaks provide RPs access to the user's identity. Escalated leaks provide unknown third parties access to the user's identity. All leaks take place secretly without the user's awareness.

Privacy Risks in SSO. OAuth and OIDC raise inherent privacy concerns as they involve an additional third party into the login process, the IdP. For instance, IdPs inevitably learn which RPs their users log in to, bearing potential for user tracking. Users must further trust IdPs not to share their personal data with RPs without obtaining their explicit permission and awareness. Over the decades, extensive research has acknowledged these privacy risks in SSO protocols. Researchers have (1) conducted formal analyses of privacy in SSO protocols [41], (2) proposed new privacy-enhancing SSO schemes [15, 18, 19, 38], (3) suggested privacy-enhancing changes or extensions to the OAuth and OIDC protocols [20, 43], (4) examined the personal data that IdPs share with RPs [31, 32, 33, 37], and (5) surveyed user perceptions of privacy in SSO [10, 11, 34, 39]. However, efforts to make SSO protocols more privacy-friendly by suggesting new or enhancing existing protocols face significant adoption barriers in practice. Today, OAuth and OIDC are still the most widely prevalent protocols in the web [24]. Thus, further investigation into SSO privacy remains crucial.

User Awareness in SSO. Previous research on SSO assumes that users intentionally click the SSO button on an RP's website. When users explicitly click the SSO button, they make a conscious decision to: (1) share the RP they are currently visiting with the IdP, and (2) share their personal data with the RP for login purposes. In this paper, we are the first to reduce these assumptions by assuming users to only visit an RP's homepage while having an account with the IdP. Additionally, users have no intention to use SSO or log in to the RP's website. Nonetheless, we show that SSO logins are still *automatically* triggered, even on an RP's homepage and without the user's intention or knowledge.

Research Questions. We raise the following questions:

RQ1 *Which SSO privacy leaks can occur if a user visits an RP's homepage without having the intention to log in or use SSO at that moment?*

RQ2 *How many RPs and IdPs are causing these leaks?*

RQ3 *How can we mitigate these SSO privacy leaks?*

In the following, we briefly summarize our findings.

RQ1: Three Novel SSO Privacy Leaks. Early in our research, we noticed that we were automatically logged in to our accounts on some websites, even though we did not intend to sign in. This observation motivated further investigation into these privacy-invasive practices. We manually investigated the reasons and quickly found that SSO logins are triggered automatically, even on the RP's homepage. We classified the observed leaks into three categories: Partial Leaks (PLs), Full Leaks (FLs), and Escalated Leaks (ELs) (see Figure 1). Each leak class builds upon the previous one. For instance, a partial leak must occur before a full leak becomes possible, and a full leak must occur before an escalated leak can take place.

Partial Leaks: We assume users are signed in to their IdP account and have a valid session cookie. When users visit the RP's homepage, the RP automatically triggers the SSO login by sending a request to the IdP that includes the session cookie and the RP's identity. This occurs entirely without user interaction, even without accepting any cookie banners. As a result, IdPs can track their users across all RPs that engage in this privacy-invasive behavior.

Full Leaks: FLs are even more privacy-invasive than PLs. Consider a user who used SSO to sign in to the RP several months ago. Now, the user has cleared the browser history, leaving no trace of that prior sign-in. Despite this, when the user revisits the RP's homepage, the IdP instantly returns the user's identity to the RP. Consequently, RPs can automatically sign users into their accounts and track their activities without their awareness and knowledge. This can even lead to sign-in loops, where users cannot sign out of an RP because they are always automatically signed in again.

Escalated Leaks: ELs maximize the leak severity by sending the user's identity to third parties. Prior work has shown that these third parties are commonly trackers [45]. In this paper, we show that these leaks are even more severe if they occur automatically on an RP's homepage and without the user's awareness.

RQ2: Large-Scale Measurement of SSO Privacy Leaks. To demonstrate the real-world significance of SSO privacy leaks, we are the first to quantify their prevalence in the Tranco top 1M websites. We extended the tool SSO-MONITOR from prior work [24] to identify the three SSO privacy leak classes. Overall, we identified 10,931 RPs exposing 11,189 PLs, 2,947 RPs with FLs, and 6 RPs with ELs. Besides major IdPs like Facebook, Google, and Microsoft, we found 1,032 additional, less-popular IdPs. These results indicate that SSO privacy leaks affect not only major IdPs but the entire SSO ecosystem.

RQ3: Mitigating SSO Privacy Leaks. To mitigate privacy leaks, we discuss several options that enhance user privacy in both the short and long term. Although IdPs offer options to opt out of automatic sign-ins, we show that they fail to prevent all leaks. Thus, we introduce our browser extension SSO PRIVACY GUARD, which successfully mitigates all of the identified leaks. Since relying on browser extensions for long-term privacy is not ideal, we also discuss several browser-level defenses. Major browser vendors have recently advanced in preventing user tracking on the web. For instance, the Federated Credential Management (FedCM) API, which is part of the privacy sandbox initiative [6], claims to provide a new privacy-preserving approach for executing SSO. It uses the trusted browser as a mediator between the RP and IdP to preserve the user's privacy. We investigate if FedCM can uphold its claims.

Contributions. We make the following contributions:

- ► We present three novel SSO privacy leaks that secretly reveal the RP's identity to the IdPs (*partial leak*), the user's identity to RPs (*full leak*), and even worse the user's identity to third-party sites (*escalated leak*). We further demonstrate that RPs can detect if a user is logged into Facebook, even without the user's consent.
- ► We conducted a large-scale study of the SSO privacy leaks by extending prior work [24]. Thereby, we identified 11,189 SSO privacy leaks on 10,931 RPs among the Tranco top 1M websites. This paper is the first to quantify RPs that misuse SSO to secretly sign in users without their awareness.
- ► To protect the privacy of users in the short term, we developed a browser extension called SSO PRIVACY GUARD. SSO PRIVACY GUARD effectively prevents all privacy leaks presented in this paper. Major browser vendors have made recent advancements to combat user tracking on the web. We discuss whether and how these measures can also protect user privacy in SSO in the long term.

Open Science. We have made all the artifacts, including the raw data and source code, publicly available as an open source contribution.[1] They enable researchers to fully reproduce our work.

2. Background: Single Sign-On

In this section, we focus on the generic SSO protocol flow, highlighting the messages and parameters relevant for understanding the SSO privacy leaks. Consider users who want to log in to the RP's website by using an account at an IdP. Therefore, the users visit the RP's login page and click on the SSO button.

1. https://sso-privacy.me

Authentication Request. The SSO login flow starts with the users requesting access to their accounts. To authenticate the users, the RP sends the authentication request (AuthRequest) to the IdP via the users' browsers. This message contains parameters specific to the particular SSO protocol in use. In OAuth and OIDC, the AuthRequest contains the identity of the RP (`client_id`) and the target to which the IdP must send the authentication response (AuthResponse) (`redirect_uri`).

User Authentication & Consent on the IdP. Before the AuthResponse is sent back to the RP, the users must authenticate to the IdP. Typically, the users have a long-lasting authenticated session with the IdP. In OAuth- and OIDC-based protocols, IdPs are asking the users if they allow the RPs to receive access to their personal data. This consent is given only once when the users employ SSO for the first time to log in to an RP.

Authentication Response. Typically, the AuthResponse is sent as an HTTP redirect from the IdP to the RP. It contains one or more of the following tokens to authenticate the user: `code`, `access_token`, `id_token`. While an `id_token` contains user information directly, an `access_token` is used as authorization to obtain user information from the IdP. In contrast, a `code` must be redeemed for an `access_token` and/or `id_token` by the RP.

SSO via In-Browser Communication. In contrast to the HTTP redirect-based SSO flow, IdPs frequently use In-Browser Communication (InBC) to exchange SSO messages [23]. This approach is applied when SSO is executed in (1) popup windows or (2) iframes. Thereby, messages are exchanged using the postMessage [42, §9.3] or Channel Messaging [42, §9.4] browser APIs. This allows RPs to receive tokens, such as `id_token` or `access_token` without sending HTTP requests to the IdP. Token exchanges between IdPs and RPs are not visible in tools that only monitor HTTP traffic, such as the browser's developer tools.

3. Single Sign-On Privacy Leaks

In this section, we present three different classes of SSO privacy leaks. We consider OAuth 2.0 (OAuth) [21] and OpenID Connect 1.0 (OIDC) [40], as both SSO protocols are widely deployed on consumer websites [9, 24, 25]. However, the presented SSO privacy leaks are generic. Other SSO protocols such as Security Assertion Markup Language (SAML) [13] can face similar issues as well.

Preconditions. There are three preconditions that have to be met: First, the RP has registered to the IdP in beforehand to establish a trust relationship. Second, we assume that the user is authenticated to the IdP. This is reasonable, as using SSO is convenient for users and IdPs often enforce that users have long-lasting sessions. For example, the Chrome browser encourages its user to log in to their Google accounts for synchronizing their browser settings and bookmarks. Third, we assume that the user once has given consent to use the IdP for authenticating on the RP.

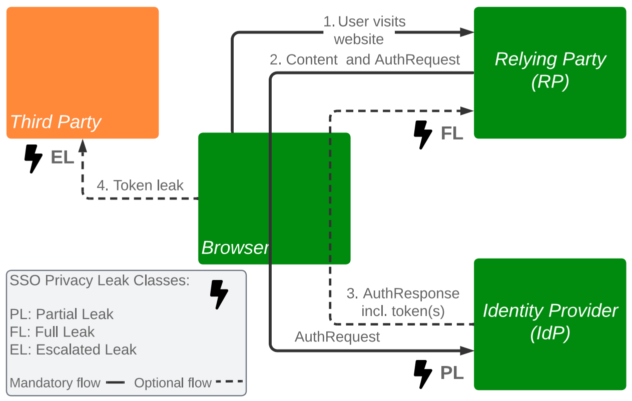


Figure 2: SSO privacy leaks occur when a user visits an RP that secretly starts an SSO protocol flow. The three leak classes build on each other while severity increases.

SSO Privacy Leaks. The SSO privacy leaks occur when a user visits an *honest but curious* RP[2] that secretly starts a standard SSO protocol flow. The user is not aware of this as the initiated protocol run is not indicated to the user (e.g., via the user interface). In the following, we describe how and where the three classes of SSO privacy leaks can occur (see Figure 2):

(1) **User visits RP**: The user navigates a browser to an RP.
(2) **Content and AuthRequest**: The RP returns the website's content *and* an SSO AuthRequest. The browser automatically transfers the AuthRequest, which includes a `client_id` and `redirect_uri`, via an HTTP redirect or JavaScript to the IdP. Here, the Partial Leak (PL) immediately occurs as the IdP can identify the RP based on `client_id` or `redirect_uri`. Moreover, the IdP can identify the user based on the session cookies that are send along with the AuthRequest. As a consequence, the IdP learns if and at what time an identified user has visited the corresponding RP.
(3) **AuthResponse including Token(s)**: The IdP may automatically send an AuthResponse that is forwarded to the RP. This leads to a Full Leak (FL), in which the RP receives access to user information, such as the user's identity or Personally Identifiable Information (PII).
(4) **Token Leak via HTTP request**: If the browser issues further HTTP requests (which is usually the case), tokens might leak to third parties. This can happen by intention (embedding tokens as explicit parameters) or unintentionally (e.g., via the Referer header due to misconfigured Referer policies). We call this leak class Escalated Leak (EL).

The three leak classes build on each other and the information disclosure expands with each class. In consequence, when an EL occurs, the user identity is revealed to the IdP, RP, and to third parties. Additionally, the IdP and third parties learn the RP a user has visited.

2. Those RPs are motivated to gather as much information as possible about their users (e.g., for targeted advertising or monetization).

4. Methodology

In this section, we present our methodology to detect the SSO privacy leaks in the wild. To carry out our methodology, we have extended the publicly available SSO-Monitor[3] [24], which allows to continuously observe the SSO landscape. SSO-Monitor is a framework for (1) loading the Tranco list, (2) managing tasks and multiprocessing, (3) configuring and automating browsers, (4) performing analysis tasks, and (5) collecting and exporting results. We have extended SSO-Monitor by implementing an SSO privacy analysis task in (4), and we have developed scripts to process and evaluate the results in (5). To ensure reproducibility, we saved a screenshot of websites if a privacy leak was detected, stored relevant messages, and captured all traffic in HTTP Archive (HAR) files [35].

Our methodology follows a three-step process:

(1) **PL Detection**: We visit the homepage of a given website using a fresh Chromium browser profile. We detect postMessage, Channel Messaging, and URL fragments by using a browser plugin provided by SSO-Monitor. No other browser plugins are incorporated. To simulate a real user, we interact with the website by randomly moving the mouse and pressing some keys (e.g., Page Up). All HTTP and InBC messages sent and received by the browser are analyzed to detect AuthRequests. We detect them by searching for specific patterns of 11 well-known IdPs provided by SSO-Monitor [24]. Additionally, we use a generic approach to detect further AuthRequests by searching for `client_id` and `redirect_uri` parameters. To be recognized as AuthRequest, both parameters must appear in one request message. AuthRequests found by our generic approach are summarized as "other" IdPs. In addition, we have improved AuthRequest detection for Facebook by adding an additional rule that can detect hidden Facebook-specific OAuth requests. For all PLs found, we categorize to which IdP the AuthRequests are sent.

(2) **FL Detection**: For each well-known IdP found in step (1), we register a new account if possible. Furthermore, we create an account for the first most common IdP out of the list of "other" IdPs. Then all websites that have revealed a PL to a relevant IdP are accessed again. This time with a fresh browser profile, which has an active IdP session. In this case, FLs should not occur, as the user gave no consent. However, to check conformance to the standard, we carry out this step nonetheless. Next, we grant user consent on the IdP by manually performing an SSO login. Finally, we repeat the automatic SSO-Monitor scan with an active IdP session for all websites where we successfully gave consent. Again, we analyze the HTTP and InBC traffic sent and received to detect AuthRequests and corresponding AuthResponse messages.

(3) **EL Detection**: For each website that revealed a FL, we use the recorded HAR file to analyze all HTTP messages to find tokens (e.g., `code`, `access_token`, or `id_token`) that are leaked to third parties. For example, this may happen via the Referer header.

We have consciously decided to search for privacy leaks that are automatically triggered when a user visits the homepage of a website. Additionally, we deliberately refrained from giving consent to any cookie banners. From a privacy perspective, those leaks are the most critical, as users are unaware and cannot prevent them. Furthermore, users have no chance to deny them or give an informed consent (e.g., via the cookie banner).

3. https://sso-monitor.me

5. SSO Privacy Leaks in the Wild

We performed an empirical study of the Tranco[4] [27] top 1M websites to discover SSO privacy leaks described in §3. In summary, we collected over 180 GB of data.

In the following, we present our main findings subdivided into the three SSO privacy leak classes (§5.1 - §5.3). Table 1 summarizes the SSO privacy leaks that we found. As we only analyzed the homepage of each website, we expect many more leaks on subpages. For example, manual tests revealed that SSO privacy leaks can be triggered when a user visits a website's login page, but *before* clicking a login button. Therefore, the found SSO privacy leaks must be considered as a lower boundary. Finally, we analyze in which website categories (§5.4) and countries (§5.5) the leaks occur.

5.1. SSO Partial Leaks

We carried out the scan of the Tranco 1M websites from March 29th to 31st, 2024 as described in step (1) of our methodology (see §4). A total of 784,821 websites were successfully analyzed, and the rest (215,179) were unreachable during our scan (e.g., name resolution failure, connection refused, TLS errors, timeouts, etc.). A detailed list of the encountered errors can be found in Table 4.

For our analysis, we configured SSO-Monitor to classify AuthRequests of the following 11 well-known IdPs: Apple, Facebook, Google, Microsoft, LinkedIn, Twitter, Github, Baidu, QQ, Sina Weibo, and WeChat. Remaining AuthRequests that cannot be mapped to our list of well-known IdPs are categorized as "other" IdPs.

In total, we discovered 11,189 PLs on 10,931 different websites (1.4% out of 784,821 websites). In 258 cases, websites triggered multiple PLs to different IdPs (ⓕ+G: 155, G+Others: 83, ⓕ+Others: 14, ⊞+Others: 6). Interestingly, in all 258 cases, PLs occurred for two IdPs. Out of our list of 11 well-known IdPs, only Facebook (ⓕ), Google (G), Microsoft (⊞), and WeChat (💬) triggered PLs. Other widespread IdPs, such as Apple and LinkedIn do not appear. In the following, we describe the leaks affecting each IdP.

4. List generated on March 20th, 2024. Available at https://tranco-list.eu/list/V99PN.

IdP	#PL	#FL	#EL
ⓕ	4,827 (43.1%)	2,678 (90.9%)	4 (66.7%)
G	4,023 (36.0%)	195 (6.6%)	2 (33.3%)
⊞	190 (1.7%)	34 (1.1%)	0
💬	20 (0.2%)	n/a$^{\alpha}$	n/a$^{\alpha}$
Others	2,129 (19.0%)	40 (1.4%)$^{\beta}$	0$^{\beta}$
$\sum$	11,189	2,947	6

$^{\alpha}$We could not test FLs/ELs, as we were unable to create a 💬 account.
$^{\beta}$Out of 1,032 IdPs, we only tested the first IdP we were able to create an account for.

TABLE 1: Number of SSO privacy leaks found on Tranco top 1M websites. AuthRequests to IdPs that cannot be mapped to our list of 11 well-known IdPs are summarized as "Others". Our findings must be considered as lower boundaries.

Facebook. The PLs to Facebook are mainly generated by the JavaScript SDK[5] that is integrated into the RPs' websites. The SDK fires a request to `https://www.facebook.com/x/oauth/status` when triggered. This request contains the `client_id` that identifies the RP to Facebook. Almost all found RPs use this method. Only three RPs send a default OAuth AuthRequest – indicating a custom OAuth implementation. In total, we found 4,827 hidden AuthRequests (i.e. PLs) to Facebook out of which 85 took place after we simulated user interactions.

Google. PLs to Google are mainly triggered by the Google One Tap SDK[6] (3,996/4,023) which is integrated into the RP website. Similar to Facebook, Google One Tap fires an automatic request to `https://accounts.google.com/gsi/status` containing the `client_id` of the RP. The main difference is that if the user has an active session at the IdP, the One Tap overlay is shown (see Figure 4a).[7] Therefore, users may notice that Google has learned which site they currently visit. Nevertheless, the request was already sent and could not have been prevented from happening in the first place.

The remaining 27 PLs are generated by automatic redirects that occur when the website is loaded. Those redirects contain the `client_id` as well. Again, this leak may be recognized by the user but cannot be prevented beforehand. Out of 4,023 identified AuthRequests, 108 took place after we simulated user interactions.

Microsoft. We found 190 PLs affecting Microsoft. All leaks were triggered automatically without any user interaction by sending a request to the domain `login.microsoftonline.com`. Following endpoints were used:

- `/common/oauth2/authorize`: 81
- `/common/oauth2/v2.0/authorize`: 33
- `/organizations/oauth2/v2.0/authorize`: 10
- `/consumers/oauth2/v2.0/authorize`: 7

5. https://developers.facebook.com/docs/facebook-login/web
6. https://developers.google.com/identity/gsi/web/guides/display-google-one-tap
7. Users can opt out of One Tap if they disable the sign-in prompt flag in their Google Account. Consequently, the leak still occurs, but the One Tab overlay is not displayed anymore.

The remaining 59 requests were sent to unique paths. Interestingly, most of them (56) used a pattern like `/<UUID>/oauth2/v2.0/authorize`, where `v2.0/` is an optional subfolder. The three remaining requests simply replaced the Universally Unique Identifier (UUID) with the domain (e.g., `/mediakind.com/oauth2/authorize`).

WeChat. All the 20 identified RPs initiated PLs to WeChat by sending a request to `open.weixin.qq.com/connect/oauth2/qrconnect`, where `oauth2/` is an optional subfolder, or `https://open.work.weixin.qq.com/wwopen/sso/qrConnect`. The AuthRequests include `appid`, `redirect_uri`, and `scope` but are not limited to these parameters. `appid` is a proprietary equivalent for `client_id`. No user interaction was required.

Others. To find all other IdPs susceptible to PLs, we searched for requests that contain the two parameters `client_id` (or `clientid`, `app_id`, `appid`) and `redirect_uri` (or `redirecturi`). By applying this approach, we identified 2,129 requests (32 after interaction) to 1,032 different IdPs. Out of these IdPs, 209 are used for more than one RP in our dataset. The top five most prevalent IdPs used by multiple RPs are `shop.app` (131), `auth0.com` (97), `q4inc.com` (70), `newscorpaustralia.com` (40), and `b2clogin.com` (37).

5.2. SSO Full Leaks

Next, we analyzed all 10,931 websites that triggered PLs to find FLs. This analysis was carried out on May 15th, 2024. In accordance to our methodology, we first tested if the IdPs conform to the standard and do not leak sensitive tokens without user consent. We visited each website that exposed a PL to an IdP with an active IdP session. Note that we used a fresh browser profile for each IdP and that these profiles only contain IdP-related cookies.

As expected, none of our well-known IdPs leaked the user's identity to the RP. However, in the case of Facebook, this IdP returned "not_authorized" to the RP. As a result, the RP learns that the visitor is currently logged in to Facebook. We found this behavior on 2,025 out of 4,827 websites that triggered PLs to Facebook. Since the RP learns something about the user from the IdP, this meets our definition of FLs. Therefore, we count them as FLs (see Table 1). In addition, we found that `newscorpaustralia.com`, categorized as "other" IdP, is generating FLs without a user having given consent to an RP.

Next, we used the freshly created IdP accounts to give consent to as much RPs as possible. Then, we visited all websites that we gave consent to again. Table 2 summarizes the identified token leaks per IdP. In the following, we discuss our insights for each IdP.

Facebook. Out of the 4,827 PLs, we were able to give consent for 905 RPs. In 2,808 cases, websites embedded the Facebook SDK but did not offer *Login with Facebook* SSO support. Furthermore, 1,043 websites offered a Facebook login but were misconfigured. Therefore, Facebook aborted the SSO protocol run. Finally, 71 websites were not reachable.

In total, we discovered 653 FLs that reveal the identity of the user to the RP. Divided by token types, Facebook leaked 652 `access_tokens` and 1 `id_token`. In 633 cases, the tokens were sent over a non-standardized channel as an `FB-AR` header, in 19 cases via an HTTP `FORM POST`, and in one case as an AuthResponse message.

Google. Out of 4,023 websites triggering PLs to Google, we were able to give consent for 594 RPs. Since our analysis for FLs took place during a global migration phase for Google's Social Login feature, we had to exclude 3,379 pages. These RPs did no longer use the common SSO flow (see §6.3). In 21 cases, websites did not offer *Sign in with Google* support. Furthermore, 16 websites offered Google login but were misconfigured. Finally, 13 websites were not reachable during our analysis.

In total, we discovered 195 FLs that reveal the identity of the user to the RP. The majority (164) are responses transmitted via postMessages. The remaining FLs arise from 22 form posts and 9 AuthResponses. Google leaked 185 `id_tokens`, 9 `codes`, and 1 `access_token`.

Microsoft. Out of 190 websites triggering PLs to Microsoft, we were able to give consent for 127 RPs. In 48 cases, RPs used SSO with Microsoft as a service to allow only a specific subset of users, most likely their employees, to log in. Therefore, our free Microsoft account was not allowed to log in to those RPs. Furthermore, on one website the login functionality was broken and in five cases the websites did not offer SSO with Microsoft. Finally, nine websites were not reachable during our analysis.

In total, we discovered 34 FLs. The majority (32) are responses transmitted by returning an HTTP redirect response with a `Location` header containing the AuthResponse inside a fragment. The remaining two responses were transmitted via form posts. In all cases, a `code` is leaked.

WeChat. In order to use SSO with WeChat, a mobile phone with the WeChat app installed is required. Furthermore, a country-specific phone number is needed to create a new account. Unfortunately, these login requirements rendered it impossible to conduct further analysis for this IdP.

Others. In our PL analysis, we identified 2,129 generic AuthRequests to 1,032 different IdPs. Given the high number of IdPs, it is impossible to generate an account for every single IdP. Furthermore, most of them are not open for public usage, as they do not allow account registration at all or only under special conditions. For example, the Auth0 IdP is used by 97 RPs which represent different companies. In order to create an account, you have to be employed by those companies.

We sorted all other IdPs by popularity and tried to create accounts for them one by one. The first IdP, out of the top most used IdPs, that we were able to create an account for was `newscorpaustralia.com`. This IdP is used by 40 RPs in our dataset. We analyzed the websites using this IdP exemplary to get more insights about SSO privacy leaks of IdPs categorized as "others". Unlike other IdPs, `newscorpaustralia.com` does not offer a standalone login. Instead, the login must be triggered via an RP. Furthermore, as soon as you are logged in to any of those RPs, any other RP gets the user information automatically without any consent was given to those RPs. This means that FLs happen virtually by design. As a direct consequence, we detected FLs on all of these websites. In all cases, an `access_token` token is leaked to the RPs.

IdP	# access_token	# id_token	# code	$\sum$ FL
ⓕ	652	1	0	653
G	1	185	9	195
⊞	0	0	34	34
News Corp AU	40	0	0	40
$\sum$ by token type	693	186	43	922

TABLE 2: Token leaks per IdP.

Summary. In total, we gave consent to 1,666 RPs (ⓕ: 905, G: 594, ⊞: 127, News Corp AU: 40) and found 922 FLs. Not all RPs with an active consent contribute to FLs. This can be explained by the fact that SDKs have automatic logins that developers can activate[8]. Thus, SDKs always initiate the AuthRequest but the IdP issues the AuthResponse only if the automatic login is activated.

5.3. SSO Escalated Leaks

For each found FL, we scanned the corresponding HAR file in order to detect ELs. First, we extracted privacy sensible tokens (i.e., `access_token`, `id_token`, and `code`) out of the detected FLs. Next, we searched for these tokens in the corresponding HAR files. Our search strategy considers cross-site requests to third parties, including the query string, cookies, all other header fields, and the message body.

In total, we found six RPs that leaked tokens to third parties from which four were Facebook `access_tokens` and two tokens from Google. One of those tokens is a `code`. Therefore, this may be uncritical if suitable security measures are in place (e.g., correct verification of the state parameter and prevention of code reusing). However, the second token is an `id_token` containing PII, such as name and email address.

In contrast to other work [12, 29], the number of ELs found is quite low. Typically, an EL occurs after the IdP redirects the user back to the RP (AuthResponse). This redirect contains the authentication tokens such as `access_tokens`, `id_tokens`, or `codes` within the URL. Consequently, tokens may be disclosed to third parties in follow-up requests, for instance, in the Referer header. However, our FLs are mostly sent through InBC or in a header of an HTTP response. Therefore, such leaks are less likely to occur.

5.4. Website Categories and Reputation

Categories. We used Trellix URL Ticketing System [8] to classify the categories of websites that triggered SSO privacy leaks. If Trellix assigns multiple categories to a

8. https://developers.google.com/identity/gsi/web/guides/automatic-sign-in-sign-out

website, we use the first category listed. For PL and FL the top three website categories are Online Shopping (PL: 13%, FL: 14%), Business (PL: 13%, FL: 8%), and Entertainment (PL: 7%, FL: 11%). Visiting these websites may reveal much about likes and interests of the users. Interestingly, we found websites in even more privacy sensible categories, such as Health (PL: 167, FL: 50), NGO/Advocacy (PL: 78, FL: 30), Religion/Ideologies (PL: 23, FL: 8), and other user traits (e.g., dating, politics, and sexual orientation). These categories may reveal much about problems, fears, needs, and attitudes of users. The FL-to-PL ratio varies widely by category. In Motor Vehicles, 101 websites had PLs, but only 8% of them exposed FLs. In Politics/Opinion, 46% of PL exposing websites also had FLs. The full table of categorized websites, inclusive the ELs, can be found in Table 5 in the Appendix.

Reputation. Trellix employs an automated system to analyze various security attributes of a URL, such as its content, its presence on the Internet, domain behavior, and other factors.[9] It assigns a reputation score that reflects the potential risk when accessing the URL.

Out of 10,931 sites, 10,107 (92%) exposed a minimal, 89 (0.8%) medium, and 23 (0.2%) a high risk. The remaining 712 (7%) websites are unverified. All ELs were found on websites with minimal risk.

5.5. Website Geolocation

We analyzed the hosting country of all websites exposing PLs (10,931) and FLs (2,947) based on the top-level domain (TLD) and IP Geolocation. First, we attributed the country to websites based on the country code top-level domain (ccTLD) (PLs: 4,260, FLs: 1,362). For the remaining websites, we used *cdncheck*[10] to exclude all domains that facilitate a Content Delivery Network (CDN) (PL: 991, FL: 148). Finally, we used MaxMind GeoLite2 Free Geolocation database [7] to determine the country of all remaining websites (PL: 5,644, FL: 1,432). Due to missing DNS entries, geolocation was undetermined for 36 websites with PLs and 5 websites with FLs. We categorized those cases and all CDN-based websites as unknown country. In total, we identified SSO privacy leaks in 148 (PL) and 102 (FL) countries, while the entire 1M dataset covers 249 countries.

Top 10 Countries. Table 3 presents a comparison of the top 10 countries hosting websites with SSO privacy leaks against their representation in the Tranco 1M dataset. Overall, France (+192%), Canada (+95%), and Poland (+70%) exhibit the most significant positive deviations, indicating a higher proportion of PLs relative to their representation in the Tranco 1M. In contrast, Russia (-63%) and Japan (-44%) have notable negative deviations. Since Facebook is blocked in Russia [14], these significantly fewer PLs on websites hosted in Russia are reasonable. Hungary (+633%), Poland (+250%), Australia (+167%), and Italy (+118%) have substantially higher proportions of FL compared to their representation in the Tranco 1M. On the other hand, Japan (-25%) and Canada (-18%) have lower proportions of FLs. In total, the top 10 countries comprise 58% of all PLs and FLs in the Tranco 1M dataset.

Effect of GDPR on EU-based Websites. The General Data Protection Regulation (GDPR) should be enforced both on websites hosted in the European Union (EU) and when EU citizens access content located on websites outside the EU. We investigated whether the proportion of EU-hosted websites with privacy leaks differs from their representation in the Tranco 1M dataset. Interestingly, 27.7% of all websites susceptible to PLs are located in the EU. In the case of FLs, the percentage is slightly higher (30.7%). Compared to 18.9% of EU-hosted sites in the Tranco 1M, we see no positive effect of the GDPR on websites hosted in the EU.

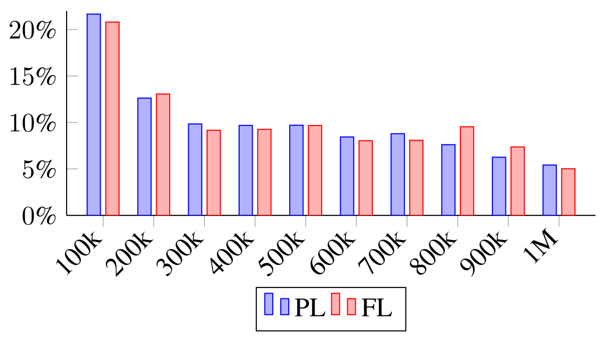


Figure 3: Percentage of the Tranco top 1M websites with SSO privacy leaks (per website rank).

5.6. Distribution of SSO Privacy Leaks

Finally, we focus on the distribution of websites susceptible to SSO privacy leaks. Figure 3 divides the analyzed Tranco 1M websites into groups of 100k (starting with those ranked highest). For each group of 100k it shows the percentage of PLs and FLs. Due to the low number of findings, we excluded ELs from the figure. We find that more popular sites are more likely susceptible to privacy leaks. This is relatively alarming because more popular sites are likely to impact the privacy of more users.

6. Defenses

This section presents defenses to prevent the privacy leaks. First, we explore if and how users can opt out from SSO privacy leaks (§6.1). Next, we introduce a new browser extension called SSO PRIVACY GUARD, which automatically blocks all SSO privacy leaks (§6.2). Then, we discuss FedCM, which aims to enable privacy-preserving SSO in the long term (§6.3). Finally, we show the effects of the ongoing deprecation of third-party cookies on SSO (§6.4).

6.1. Can Users Opt Out of Privacy Leaks?

There are two ways RPs can integrate SSO: (1) Manually communicate with the IdP's Application Programming In-

9. https://trustedsource.org/download/ts_wd_reference_guide.pdf, p. 8
10. https://github.com/projectdiscovery/cdncheck

CC	# **PLs** (%)	Tranco 1M (%)	Δ	CC	# **FLs** (%)	Tranco 1M (%)	Δ
US	3,316 (30.3%)	320,479 (32.0%)	-5.3%	US	954 (32.4%)	320,479 (32.0%)	+1%
FR	795 (7.3%)	24,709 (2.5%)	+192%	DE	146 (5.0%)	53,181 (5.3%)	+6%
DE	582 (5.3%)	53,181 (5.3%)	± 0%	PL	103 (3.5%)	9,847 (1.0%)	+250%
CA	452 (4.1%)	22,341 (2.2%)	+95%	FR	89 (3.0%)	24,709 (2.5%)	+20%
BR	257 (2.4%)	19,945 (2.0%)	+20%	BR	88 (3.0%)	19,945 (2.0%)	+50%
RU	231 (2.1%)	57,827 (5.7%)	-63%	JP	71 (2.4%)	31,684 (3.2%)	-25%
JP	198 (1.8%)	31,684 (3.2%)	-44%	IT	71 (2.4%)	11,161 (1.1%)	+118%
IN	185 (1.7%)	16,381 (1.6%)	+6%	AU	70 (2.4%)	9,071 (0.9%)	+167%
PL	183 (1.7%)	9,847 (1.0%)	+70%	HU	64 (2.2%)	3,398 (0.3%)	+633%
IT	180 (1.4%)	11,161 (1.1%)	+27%	CA	54 (1.8%)	22,341 (2.2%)	-18%
$\sum$	6,379 (58.4%)	567,555 (56.7%)		$\sum$	1,710 (58.0%)	505,816 (50.5%)	

TABLE 3: Top 10 countries hosting websites with SSO privacy leaks compared to the distribution in the Tranco 1M (CC: ISO country code, Δ: deviation to Tranco 1M).

terface (API) for SSO. (2) Use a Software Development Kit (SDK) provided by the IdP. Depending on the integration method, users have different options to opt out of the leaks.

SSO APIs. SSO protocols usually require the RP to manually interact with the IdP's API, leaving no centralized option for users to opt out. Since each RP implements SSO individually, they all have to provide separate mechanisms to opt out. Users are forced to opt out on every RP separately, which is unrealistic in practice.

SSO SDKs. If the RP integrates SSO with an SDK provided by the IdP, users could theoretically opt out once at the central IdP. The IdP can then configure all their SDK instances on all websites to stop sign-in prompts and similar, which are responsible for the privacy leaks discussed in this paper. Google, for example, allows users to opt out of Google One Tap prompts (see. Figure 5a) and FedCM prompts (see. Figure 5b). In the Google account settings / Chrome browser settings, users can disable sign-in prompts. We found that if users disable the Google One Tap prompt, it no longer appears, although a request containing the user's and RP's identity is still sent to Google. Instead of returning the prompt, Google returns an erroneous status code. However, the PL is not prevented, as information still leaks to Google. Our case study of 11 IdPs (see. §5.1) revealed that they do not offer such mechanisms.

Consent Renewal. Even worse, the SSO privacy leaks undermine the automatic consent removal feature implemented by IdPs. When users do not access a previously authorized RP for an extended period, the IdP revokes their consent. For instance, Facebook and Microsoft remove consent after 90 days of inactivity, while Google does so after six months. However, the privacy leaks constantly retrigger the SSO logins, ensuring that the consent expiry date is continually renewed. Consequently, consent to RPs is never removed if users actively visit the RPs.

6.2. Our Solution: SSO PRIVACY GUARD

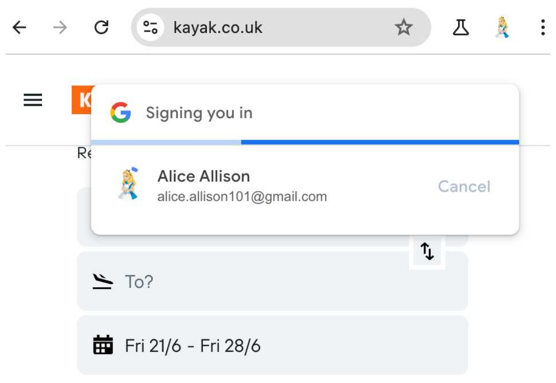
To effectively defend against *all* SSO privacy leaks identified in this paper (PLs, FLs, and ELs), we introduce SSO PRIVACY GUARD. This browser extension intercepts all SSO messages and applies both IdP-specific and generic rules on each request to detect SSO AuthRequests. Our approach aims to block these requests by using a strategy that is orthogonal to the Google One Tap SDK.

Figure 4a: The Google One Tap Experience. The Google One Tap SDK streamlines the sign-in process. When a user visits a website using this SDK, a prompt appears in the top-right corner, as shown in Figure 4a. This prompt notifies the user of the automatic sign-in and initiates a three-second countdown. If the user does not click the "cancel" button during this period, the SDK completes the sign-in automatically. Should the user log out, the websites may reactivate the SDK, triggering the automatic sign-in once more. This feature can occasionally result in a deadlock situation, preventing the user from logging out.

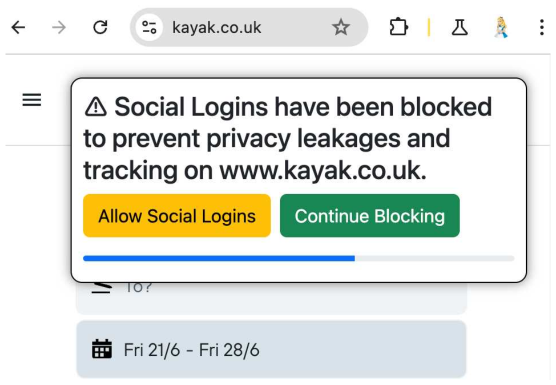
Figure 4b: Reversing the Google One Tap Experience. SSO PRIVACY GUARD reverses this experience. Rather than automatically signing in the user, which leads to PLs and FLs, SSO PRIVACY GUARD proactively detects and blocks all SSO requests by default. When SSO PRIVACY GUARD blocks an SSO request, it displays a prompt inspired by the Google One Tap UI, see Figure 4b. Within a ten-second window, users are presented with three choices: First, by clicking the "Allow Social Logins" button, they can unblock the request and enable SSO. Second, they can choose to click the "Continue Blocking" button to keep the prompt dismissed while still blocking the requests. Third, ignoring the prompt will automatically continue to block the request, which then vanishes after ten seconds.

Memorizing the User's Choice. The extension records the user's choice to either allow or block SSO requests on a per-site basis. This feature enables users to permit SSO on certain websites while blocking them on others. Importantly, the prompt appears only once per website. If a user decides to block SSO – either by clicking the button or letting the timeout expire – the decision is saved and future SSO requests are automatically blocked without repeated prompts. The same applies if a user chooses to allow SSO. While the prompt is shown just once, users can modify their preferences for each website later in the extension settings.

User-Initiated SSO. SSO PRIVACY GUARD blocks all SSO requests that occur without user awareness, such as those embedded in invisible iframes. However, it allows all user-initiated SSO requests to proceed. This design ensures

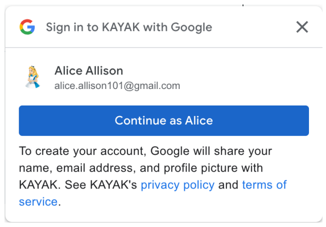


(a) Automatic Google SSO on kayak.co.uk.


kayak.co.uk
Social Logins have been blocked to prevent privacy leakages and tracking on www.kayak.co.uk.
Allow Social Logins
Continue Blocking
Fri 21/6 - Fri 28/6

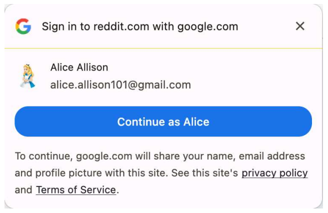
(b) SSO Privacy Guard blocks the Google SSO.

Figure 4: SSO Privacy Guard effectively blocks all privacy leaks discussed in this paper.



To create your account, Google will share your name, email address, and profile picture with KAYAK. See KAYAK's privacy policy and terms of service.

(a) Google One Tap Prompt.


Sign in to reddit.com with google.com
Alice Allison
alice.allison101@gmail.com
Continue as Alice

To continue, google.com will share your name, email address and profile picture with this site. See this site's privacy policy and Terms of Service.

(b) FedCM Prompt.

Figure 5: Comparison of the Google One Tap Prompt on kayak.co.uk and the FedCM Prompt on reddit.com.

that the extension does not block SSO requests when users have consciously chosen to engage in the SSO login flow by clicking the SSO button. We analyze information from HTTP request headers and data enriched by the extension APIs to determine if a request results from explicit user interaction or occurs without the user's knowledge.

Proof of Effectiveness. To demonstrate the effectiveness of SSO Privacy Guard, we tested it against all 10,931 websites having at least one SSO privacy leak. In all cases, SSO Privacy Guard was triggered and effectively prevented the privacy leaks.

6.3. The Federated Credential Management API

The Federated Credential Management (FedCM) API [3] is a new browser API that enables privacy-preserving SSO. It uses the browser to mediate between the IdP and RP. FedCM is part of the privacy sandbox initiative [6], which aims to enhance or create web technologies while fully preserving user privacy. FedCM displays a browser-mediated dialog, letting users select accounts from their IdPs to log into websites. Currently, the FedCM API is available in Chrome (since v108) and as an experimental feature in Firefox.

Figure 5a: The Google One Tap Prompt. The FedCM API succeeds the Google One Tap SDK [4]. Google One Tap shows a sign-in prompt on the RP's website, as illustrated in Figure 5a. This prompt asks the users if they want to sign in on the RP using their Google account. Technically, the prompt is an iframe operated by Google and embedded on the RP's website. The iframe displays both the user's name and the RP's name, allowing Google to know both the user and the RP they are visiting. We define this as a PL.

Figure 5b: The FedCM Prompt. The FedCM API displays similar information, including the user's and RP's name, as shown in Figure 5b. Unlike the Google One Tap prompt, the FedCM prompt is a native browser interface and not an iframe. The browser acts as an intermediary that is located between the RP and IdP, knowing both the user and the RP being visited. Since the browser is a trusted entity that already has this information, this seams reasonable. However, the RP remains unaware of the user's Google account, and the IdP does not know the user who is visiting the RP. This prevents all PLs while displaying a similar sign-in prompt.

Prevention of Privacy Leaks. FedCM employs a straightforward and effective approach: The IdP never receives a request containing both the user's and RP's names. Instead, when an RP invokes the FedCM API, the browser sends two separate requests: First, the browser sends a request with the user's session cookies to the IdP's accounts endpoint. The IdP receives the user's identity, but no information to identify the RP. For example, the browser removes the `Origin` and `Referer` headers. The IdP then returns the user's account information like the name, email, and profile picture, which is shown in the FedCM prompt (see Figure 5b). Second, the browser sends a request with the RP's name to the IdP's metadata endpoint. The IdP receives the RP's identity but no information to identify the user. For example, the browser removes all session cookies. The IdP then returns the privacy policy and terms of service that the RP pre-configured at the IdP. By separating the user's and RP's identities into two requests, the IdP does not receive any PLs. Certain security mechanisms are in place to prevent curious IdPs from correlating the requests.

FedCM in the Wild. Since April 2024, Google has been discontinuing its Google One Tap API [4] and began migrating to FedCM [3]. To assess the migration to FedCM and its impact on user privacy, we conducted two analyses of PLs. The first one took place from March 29th to March 31st. The second one took place on May 15th. For the second run, we only analyzed pages that triggered a PL in the first run. This allowed us to observe the adoption rate of new privacy-preserving web technologies like FedCM and their effect on the SSO privacy. Google generally recommends developers to use FedCM when implementing SSO on their websites. According to the migration guide [4], websites using the Google One Tap SDK should automatically switch to FedCM by default.

A comparison of the pre- and post-FedCM analyses revealed that 3,379 of the 4,023 RPs no longer triggered PL to Google as a result of their adoption of FedCM. Thus, FedCM reduces PLs affecting Google by nearly 84%, significantly improving user privacy. We examined the remaining 644 websites and discovered that 235 RPs manually disabled FedCM in the Google One Tap SDK by setting the parameter `has_opted_out_fedcm=true`. We expect these websites to continue generating privacy leaks. Interestingly, we found 116 RPs that still trigger PLs despite opting for `has_opted_out_fedcm=false`. This is reasonable as Google gradually rolls out FedCM over time. We anticipate these remaining privacy leaks will disappear once Google completes its roll-out of FedCM for all RPs. However, we expect RPs that have explicitly opted out of FedCM or that manually integrate the Google SSO APIs to remain vulnerable to privacy leaks.

Future Expectations. FedCM shows that browser vendors are aware of our leaks and are introducing new privacy-preserving browser APIs to mitigate them. Our measurements confirm that these efforts are improving user privacy. However, FedCM adoption is limited due to the lack of cross-browser support and limited adoption by RPs and IdPs. Currently, only Chromium-based browsers support the FedCM API[11] However, our research shows that other IdPs are starting to implement FedCM support, such as Spotify[12] and Paypal[13]. In addition, popular open source libraries such as KeyCloak are currently working on implementing FedCM[14]. We look forward that browser vendors and IdPs are adopting FedCM to resolve SSO privacy leaks. Until then, we encourage users to install SSO PRIVACY GUARD for protecting their privacy.

6.4. Deprecation of Third-Party Cookies

Major browsers are deprecating third-party cookies to enhance user privacy and prevent tracking. Firefox and Safari already block third-party cookies by default. Chrome began phasing out third-party cookies for 1% of their users in 2024, with plans to extend this to more users in early 2025 [5]. Blocking third-party cookies aims to end cookie-based user tracking on the web.

11. https://caniuse.com/mdn-api_federatedcredential.
12. https://spotify.com/.well-known/web-identity
13. https://paypal.com/.well-known/web-identity
14. https://github.com/keycloak/keycloak/issues/16834

Third-Party Cookies in SSO. Third-party cookies impact SSO protocols because they involve a third-party entity, the IdP. For instance, the leak from the IdP's login prompt embedded on the RP's website would become ineffective. However, since the deprecation of third-party cookies affects all SSO protocols by default, browsers have designed exceptions specifically for SSO. These exceptions aim to prevent tracking through third-party trackers while still allowing SSO logins to function properly.

Browser Heuristics for SSO. Browsers implement special heuristics [1] to block trackers while still allowing IdPs to access third-party cookies. These heuristics are fine-tuned to ensure that popular SSO SDKs from Google, Facebook, and others function properly. As a result, the deprecation of third-party cookies stops cookie-based tracking but does not prevent the SSO privacy leaks.

7. Discussion

7.1. Who takes Responsibility?

Given the identified SSO privacy leaks, the question arises: who is responsible? Today, many IdPs provide their own SDKs and code generators for integrating SSO on websites. Our study shows that for Google and Facebook, these SDKs cause most of the leaks. This is not surprising as IdPs recommend using their official SDKs. In rare cases, we have found SSO privacy leaks on websites using custom SSO code. Therefore, IdPs could improve privacy with little effort by adapting their SDKs. Interestingly, other major IdPs, such as Apple, also provide SDKs but do not experience these SSO privacy leaks. This proves that privacy-protecting SDKs are possible.

However, our findings also indicate that RPs contribute to the problem. Many websites either do not offer SSO logins or have non-functional implementations. Despite this, SDKs remain integrated, suggesting that website operators did not remove them. Independent of this, RPs can always solve privacy leaks on their own. However, curious RPs always have the option to deploy SSO in a privacy-invasive and stealthy manner.

7.2. Limitations

Inaccessible IdPs. Some IdPs are not open for public registration, such as enterprise IdPs. In addition, IdPs may have special login requirements, such as country-specific phone numbers. For such IdPs, we can only detect PLs.

Duplicate RPs. Sometimes, multiple Tranco websites redirect users to the same RP. For instance, visiting https://basketballfansearch.com redirects the user to https://www.bing.com. Consequently, some RPs may be counted multiple times in our study.

Leak Visibility. Our tests revealed that websites may display automatic login prompts through SSO, making the leaks visible to the user. For Google, this occurs via the One Tap overlay (see Figure 5a). For Facebook and other

IdPs, the RP decides whether to show such a prompt to users. Regardless of the visibility of leaks, they still leak information even before they are visible.

Active IdP session. In our SSO privacy leak scenario, the user must be authenticated to the IdP. However, PLs occur even when the user has no active session with the IdP. In such cases, the IdP does not know the user's identity but can still track how the user navigates through the web. Furthermore, once the user logs into their IdP account, the gathered tracking data can be linked to the real identity.

7.3. Tracking via Cookies and Social Plugins

Cookie-based vs. SSO-based Privacy Leaks. With PLs, the IdP gathers information about which users visit which RPs, often without the users' knowledge. For instance, users may log in to an RP, log out, and clear their cookies. If user revisit the same RP, they may be automatically logged in again without noticing. Thus, the SSO-based privacy leaks discussed in this paper do not rely on the RP's cookies. Even if users clear all their stored cookies, SSO leaks remain possible, while cookie-based tracking is prevented.

Social Plugins vs. SSO Privacy Leaks. Social plugins, like Facebook's Like and Share button, are notorious for privacy violations [26]. These plugins enable social networks to track individuals' browsing behavior, similar to PLs. However, our discovered SSO privacy leaks are even more severe. They expose individuals' full identities to RPs (i.e., FL) as well as third parties (i.e., EL). In 2019, the Court of Justice of the European Union (CJEU) decided that websites using social plugins are jointly responsible with the social network for the collection and transmission of user data [36]. This emphasizes the need for websites to obtain explicit user consent before using such plugins. Therefore, social plugins are only accessible to users in the EU if they are (1) logged into their social networking account, and (2) have provided explicit consent on the website (e.g., via the cookie banner). In contrast, our measurement suggests that SSO privacy leaks are effective independent of the user's consent. Thus, we try to engage in discussions with legal experts to review if these invasive SSO privacy leaks violate the GDPR. Please note that our large-scale scan was conducted within the EU. By law, the GDPR applies not only to companies within the EU but also to any company offering services in the EU.

7.4. Ethical Considerations

Scanning. We followed prior research [24] in implementing best scanning practices. Since we only loaded the homepage of each website and limited our interactions to a few key presses, this should not cause any operational problems for the websites. We set up reverse DNS on our scanning server, and deployed a disclaimer page to inform about our research. We only used our own test accounts and did not interfere with other user accounts or data.

Disclosure. Since SDKs are the main source of the privacy leaks, we started our responsible disclosure process by contacting the IdPs. Next, we reached out to the website operators that produce ELs, as these leaks have the highest impact on user privacy. We have also initiated discussions with the European Center of Digital Rights[15] (NOYB) to explore violations of the GDPR.

8. Related Work

It is well-known that current SSO protocols such as OAuth and OIDC are not designed for privacy. Issues range from leaking user-specific parameters up to user tracking and full identity disclosure.

Privacy of SSO Protocols. Fett et al. [18] addressed privacy concerns subject to the IdP in 2015. They saw a major downside in the user's privacy, as IdPs can track at which RP a user authenticates. To improve privacy, a new web-based SSO system was proposed and formally proven secure. In 2016, Isaakidis et al. [22] tackled this privacy issue by adapting the OAuth protocol to include persistent, unlinkable pseudo-identities. Therefore, IdPs do not learn the account ownership of their users. In 2016, Maheswaran et al. [30] proposed CryptoBook, which added a cryptographic anonymization layer on top of OAuth. Although it makes use of OAuth, it is an entirely different credential system that requires a user to obtain a credential from multiple credential producers (i.e., IdPs). Additionally, Hammann et al. proposed two privacy-friendly extensions to the OIDC protocol to achieve login unlinkability with respect to the IdP and colluding RPs [20]. In 2021, Zhang et al. [44] introduced EL PASSO, an SSO system that merges the security of anonymous credentials with the convenience of OIDC. This system shields users from tracking by RPs or IdPs and allows disclosing only minimal user information. However, in all cases, substantial changes in the way the protocols operate are necessary. Despite previous efforts in improving the privacy of OAuth and OIDC, we still do not see these enhancements being implemented.

Li and Mitchell [28] systematically analyzed how IdPs can track users' interactions with RPs. They conclude that the OAuth and OIDC protocols need to be significantly changed to mitigate these privacy concerns. In our paper, we accept the fact that OAuth and OIDC have inherent privacy issues. In contrast, we show that current deployments misuse standard SSO protocol runs to stealthily leak private information without the user's awareness.

In 2017, Farooqi et al. [17] revealed how leaked Facebook access tokens are widely abused by collusion networks to generate fake likes and comments. They further studied how these networks are stealing tokens from popular third-party services and proposed appropriate mitigation strategies. However, in their privacy model access token leakage is done intentionally by the user and requires the OAuth implicit flow. In contrast, our privacy leaks are generic and happen secretly without any user interaction.

Calzavara et al. [12] and Li et al. [29] developed browser plugins to detect and prevent security and privacy issues

15. https://noyb.eu/

on OAuth/OIDC protocol flows. In contrast to our paper, they do not address inherent privacy leaks to IdPs. Both may detect leakage of sensitive parameters such as `code`, `access_token`, and `id_token` when divulged via the Referer header by HTTP requests from embedded resources on the RP website. However, they fail to protect against PLs and FLs, as both carry out benign SSO protocol flows. In contrast, SSO PRIVACY GUARD mitigates all three leaks.

Real-World Privacy Implications of SSO. Morkonda et al. [31] released an empirical study to understand the privacy implications of OAuth-based SSO on 2,500 top websites across five countries in 2021. By analyzing the requested data from intentionally started SSO flows, it was shown that popular RPs often request differing amounts of user data for each IdP they offer. They also found that privacy-friendly choices are typically the last option on login pages. In addition, Dimova et al. [16] showed that RPs often request more privacy related information from IdPs than necessary. They argue that 18.53% of the RPs request access to more personal information about the user than needed. In 2022, Morkonda et al. [32] released SSOPrivateEye (SPEye), a browser extension prototype that extracts and displays permission request information from SSO login options in RPs. This helps users to make informed login decisions regarding privacy *before* starting a SSO flow. The extension supports three IdPs (Apple, Facebook, and Google) and is triggered when the user intentionally starts an SSO login on an RP. In contrast, SSO PRIVACY GUARD detects attempts to leak privacy information through stealthy SSO protocol runs. It supports 11 IdPs and can detect generic OAuth/OIDC AuthRequests.

Recently, Pham et al. [37] analyzed the effect of third-party tracking activities of RPs before and after logins with Google and Facebook SSO. While the authors focused on leaks to third parties based on advertising and tracking blacklists, we investigate leaks to IdPs and RPs that occur through standard SSO protocol runs.

9. Conclusion

This paper presents three new privacy leaks to the SSO research community. We found that IdPs can track their users across all RPs without their permission and awareness (*partial leak*). Even worse, the user's identity is leaked automatically to RPs (*full leak*) and unknown third parties (*escalated leak*). We quantified the prevalence of these leaks with a measurement of the Tranco top 1M websites. In total, 10,931, 2,947, and 6 RPs are affected, respectively.

We identified the root cause of most leaks as IdP-provided SDKs, making them responsible for the majority of these leaks. Besides popular IdPs like Google, Facebook, and Microsoft, we found 1,032 additional less-popular IdPs to be affected. These results imply that SSO privacy leaks affect the entire SSO ecosystem. However, our study also raises hope, as some IdPs (e.g., Apple) respect user privacy and prevent these leaks.

To protect user privacy in the short term, we implemented the browser extension SSO PRIVACY GUARD, which effectively blocks all leaks. Additionally, major browser vendors like Google are standardizing and adopting a new native browser API for privacy-preserving SSO, called FedCM. We discussed how the trusted browser, when used as a mediator between RPs and IdPs, can enhance user privacy in SSO. We extended our discussion with a real-world measurement, confirming that SSO privacy leaks significantly decrease with the rapid adoption of FedCM. We hope this will spark new research on how browsers can improve user privacy in SSO in the long term.

Acknowledgments

We thank the reviewers for their valuable feedback. Louis Jannett was supported by the research project "North-Rhine Westphalian Experts in Research on Digitalization (NERD II)", sponsored by the state of North Rhine-Westphalia – NERD II 005-2201-0014.

References

[1] 3pcd-exemption-heuristics/explainer.md at main · amaliev/3pcd-exemption-heuristics. https://github.com/amaliev/3pcd-exemption-heuristics/blob/main/explainer.md. (Accessed on 05/25/2024).

[2] Facebook Login. https://developers.facebook.com/docs/facebook-login/. (Accessed on 05/27/2024).

[3] Federated Credential Management API Overview — Privacy Sandbox — Google for Developers. https://developers.google.com/privacy-sandbox/cookies/fedcm. (Accessed on 05/24/2024).

[4] Migrate to FedCM — Authentication — Google for Developers. https://developers.google.com/identity/gsi/web/guides/fedcm-migration. (Accessed on 05/24/2024).

[5] Prepare for Phasing Out Third-Party Cookies — Privacy Sandbox — Google for Developers. https://developers.google.com/privacy-sandbox/3pcd. (Accessed on 05/25/2024).

[6] Privacy Sandbox — Google for Developers. https://developers.google.com/privacy-sandbox. (Accessed on 05/24/2024).

[7] MaxMind GeoLite2 Free Geolocation Data. https://dev.maxmind.com/geoip/geolite2-free-geolocation-data. (Accessed on 05/20/2024).

[8] Trellix – Cutomer URL Ticketing System. https://trustedsource.org/en/feedback/url. (Accessed on 05/09/2024).

[9] Calvin Ardi and Matt Calder. The Prevalence of Single Sign-On on the Web: Towards the Next Generation of Web Content Measurement. In *Proceedings of the 2023 ACM on Internet Measurement Conference*, IMC '23, pages 124–130. Association for Computing Machinery. ISBN 9798400703829. doi: 10.1145/3618257.3624841. URL https://dl.acm.org/doi/10.1145/3618257.3624841.

[10] David G. Balash, Xiaoyuan Wu, Miles Grant, Irwin Reyes, and Adam J. Aviv. Security and Privacy Perceptions of Third-Party Application Access for Google Accounts. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 3397–3414. USENIX Association. ISBN 978-1-939133-31-1. URL https://www.usenix.org/conference/usenixsecurity22/presentation/balash.

[11] Lujo Bauer, Cristian Bravo-Lillo, Elli Fragkaki, and William Melicher. A Comparison of Users' Perceptions of and Willingness to Use Google, Facebook, and Google+ Single-Sign-On Functionality. In *Proceedings of the 2013 ACM Workshop on Digital Identity Management*, pages 25–36. Association for Computing Machinery. ISBN 978-1-4503-2493-9. doi: 10.1145/2517881.2517886. URL https://dl.acm.org/doi/10.1145/2517881.2517886.

[12] Stefano Calzavara, Riccardo Focardi, Matteo Maffei, Clara Schneidewind, Marco Squarcina, and Mauro Tempesta. WPSE: Fortifying Web Protocols via Browser-Side Security Monitoring. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1493–1510. USENIX Association. ISBN 978-1-939133-04-5. URL https://www.usenix.org/conference/usenixsecurity18/presentation/calzavara.

[13] Scott Cantor, John Kemp, Rob Philpott, and Eve Maler. Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, 15.03.2005, 2005. URL http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf.

[14] Elizabeth Culliford and Elizabeth Culliford. Russia Blocks Facebook, Accusing It of Restricting Access to Russian Media. *Reuters*, March 2022. URL https://www.reuters.com/business/media-telecom/russia-blocks-facebook-accusing-it-restricting-access-russian-media-2022-03-04/.

[15] Arkajit Dey and Stephen Weis. PseudoID: Enhancing Privacy in Federated Login. In *Hot Topics in Privacy Enhancing Technologies*, pages 95–107, 2010.

[16] Yana Dimova, Tom Van Goethem, and Wouter Joosen. Everybody's Looking for SSOmething: A Large-Scale Evaluation on the Privacy of OAuth Authentication on the Web. In *Proceedings on Privacy Enhancing Technologies*, volume 2023', pages 452–467, 2023.

[17] Shehroze Farooqi, Fareed Zaffar, Nektarios Leontiadis, and Zubair Shafiq. Measuring and Mitigating Oauth Access Token Abuse by Collusion Networks. In *Proceedings of the 2017 Internet Measurement Conference*, IMC '17, page 355–368, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450351188. doi: 10.1145/3131365.3131404. URL https://doi.org/10.1145/3131365.3131404.

[18] Daniel Fett, Ralf Küsters, and Guido Schmitz. SPRESSO: A Secure, Privacy-Respecting Single Sign-On System for the Web. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, page 1358–1369, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450338325. doi: 10.1145/2810103.2813726. URL https://doi.org/10.1145/2810103.2813726.

[19] Ge Gao, Yuan Zhang, Yaqing Song, and Shiyu Li. PrivSSO: Practical Single-sign-on Authentication against Subscription/Access Pattern Leakage. pages 1–1. ISSN 1556-6021. doi: 10.1109/TIFS.2024.3392533. URL https://ieeexplore.ieee.org/abstract/document/10506704.

[20] Sven Hammann, Ralf Sasse, and David Basin. Privacy-Preserving OpenID Connect. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, ASIA CCS '20, page 277–289, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450367509. doi: 10.1145/3320269.3384724. URL https://doi.org/10.1145/3320269.3384724.

[21] D. Hardt. The OAuth 2.0 Authorization Framework. RFC 6749, IETF, October 2012. URL http://tools.ietf.org/rfc/rfc6749.txt.

[22] Marios Isaakidis, Harry Halpin, and George Danezis. UnlimitID: Privacy-Preserving Federated Identity Management Using Algebraic MACs. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*, pages 139–142, 2016.

[23] Louis Jannett, Vladislav Mladenov, Christian Mainka, and Jörg Schwenk. DISTINCT: Identity Theft using In-Browser Communications in Dual-Window Single Sign-On. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. Association for Computing Machinery, 2022. ISBN 978-1-4503-9450-5. doi: 10.1145/3548606.3560692.

[24] Louis Jannett, Maximilian Westers, Tobias Wich, Christian Mainka, Andreas Mayer, and Vladislav Mladenov. SoK: SSO-Monitor - The Current State and Future Research Directions in Single Sign-On Security Measurements. In *IEEE 9th European Symposium on Security and Privacy (Euro S&P)*. IEEE, July 2024. URL https://sso-monitor.me/paper.pdf.

[25] Oscar Järpehult, Fredrik Josefsson Ågren, Madeleine Bäckström, Linn Hallonqvist, and Niklas Carlsson. A Longitudinal Characterization of the Third-Party Authentication Landscape. In *2022 IFIP Networking Conference (IFIP Networking)*, pages 1–9, June 2022. doi: 10.23919/IFIPNetworking55013.2022.9829804. ISSN: 1861-2288.

[26] Georgios Kontaxis, Michalis Polychronakis, Angelos D. Keromytis, and Evangelos P. Markatos. Privacy-Preserving Social Plugins. pages 631–646, 2012. URL https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/kontaxis.

[27] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium*, NDSS 2019, February 2019. doi: 10.14722/

ndss.2019.23386. URL https://tranco-list.eu/.

[28] Wanpeng Li and Chris J Mitchell. User Access Privacy in OAuth 2.0 and OpenID Connect. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 664–6732, 2020. doi: 10.1109/EuroSPW51379.2020.00095.

[29] Wanpeng Li, Chris J Mitchell, and Thomas Chen. OAuthGuard: Protecting User Security and Privacy with OAuth 2.0 and OpenID Connect. In *Proceedings of the 5th ACM Workshop on Security Standardisation Research Workshop*, pages 35–44, 2019.

[30] John Maheswaran, Daniel Jackowitz, Ennan Zhai, David Isaac Wolinsky, and Bryan Ford. Building Privacy-Preserving Cryptographic Credentials from Federated Online Identities. In *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*, CODASPY '16, page 3–13, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450339353. doi: 10.1145/2857705.2857725. URL https://doi.org/10.1145/2857705.2857725.

[31] Srivathsan G Morkonda, Sonia Chiasson, and Paul C van Oorschot. Empirical Analysis and Privacy Implications in OAuth-based Single Sign-On Systems. In *Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society*, pages 195–208, New York, NY, USA, nov 2021. Association for Computing Machinery. ISBN 9781450385275. doi: 10.1145/3463676.3485600.

[32] Srivathsan G. Morkonda, Sonia Chiasson, and Paul C. Oorschot. SSOPrivateEye: Timely Disclosure of Single Sign-On Privacy Design Differences, 2022. URL http://arxiv.org/abs/2209.04490.

[33] Srivathsan G. Morkonda, Sonia Chiasson, and Paul C. van Oorschot. "Sign in with ... Privacy": Timely Disclosure of Privacy Differences among Web SSO Login Options, 2023. URL https://arxiv.org/abs/2209.04490.

[34] Srivathsan G. Morkonda, Sonia Chiasson, and Paul C. van Oorschot. Influences of Displaying Permission-related Information on Web Single Sign-On Login Decisions, 2023. URL http://arxiv.org/abs/2308.13074.

[35] Jan Odvarko. HTTP Archive (HAR) Format v1.2, August 2012. URL http://www.softwareishard.com/blog/har-12-spec/.

[36] Official Journal of the European Union (OJEU). Case c-40/17: Fashion ID GmbH & CoKG v Verbraucherzentrale NRW eV, July 2019. URL https://eur-lex.europa.eu/legal-content/en/TXT/?uri=CELEX:62017CJ0040.

[37] Tien-Huy Pham, Quoc-Huy Vo, Ha Dao, and Kensuke Fukuda. SSOLogin: A Framework for Automated Web Privacy Measurement With SSO Logins. In *Proceedings of the 18th Asian Internet Engineering Conference*, AINTEC '23, pages 69–77, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400709395. doi: 10.1145/3630590.3630599. URL https://doi.org/10.1145/3630590.3630599.

[38] Victor Sucasas, Georgios Mantas, Saud Althunibat, Leonardo Oliveira, Angelos Antonopoulos, Ifiok Otung, and Jonathan Rodriguez. A Privacy-Enhanced OAuth 2.0 based Protocol for Smart City Mobile Applications. 74:258–274. ISSN 01674048. doi: 10.1016/j.cose.2018.01.014. URL https://linkinghub.elsevier.com/retrieve/pii/S0167404818300361.

[39] San-Tsai Sun, Eric Pospisil, Ildar Muslukhov, Nuray Dindar, Kirstie Hawkey, and Konstantin Beznosov. What Makes Users Refuse Web Single Sign-On? An Empirical Investigation of OpenID. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*, SOUPS '11. Association for Computing Machinery. ISBN 978-1-4503-0911-0. doi: 10.1145/2078827.2078833. URL https://doi.org/10.1145/2078827.2078833.

[40] The OpenID Foundation (OIDF). OpenID Connect Core 1.0, February 2014. URL http://openid.net/specs/openid-connect-core-1_0.html.

[41] Kailong Wang, Guangdong Bai, Naipeng Dong, and Jin Song Dong. A Framework for Formal Analysis of Privacy on SSO Protocols. In *Security and Privacy in Communication Networks*, pages 763–777. Springer International Publishing. ISBN 978-3-319-78813-5. doi: 10.1007/978-3-319-78813-5_41.

[42] WHATWG. HTML Living Standard, 2022. URL https://html.spec.whatwg.org/.

[43] Rongwu Xu, Sen Yang, Fan Zhang, and Zhixuan Fang. MISO: Legacy-Compatible Privacy-Preserving Single Sign-On Using Trusted Execution Environments. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, volume 10, page 352–372. IEEE, July 2023. doi: 10.1109/eurosp57164.2023.00029. URL http://dx.doi.org/10.1109/EuroSP57164.2023.00029.

[44] Z. Zhang, M. Krol, A. Sonnino, L. Zhang, and E. Rivière. EL PASSO: Efficient and Lightweight Privacy-preserving Single Sign On. In *Proceedings on Privacy Enhancing Technologies*, volume 2021, pages 70–87. Privacy Enhancing Technologies Symposium Advisory Board, April 2021. doi: 10.2478/popets-2021-0018. URL https://openaccess.city.ac.uk/id/eprint/26898/.

[45] Yuchen Zhou and David Evans. SSOScan: Automated Testing of Web Applications for Single Sign-On Vulnerabilities. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 495–510, San Diego, CA, August 2014. USENIX Association. ISBN 978-1-931971-15-7. URL https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/zhou.

10. Appendix

Error Type	Count
Network errors	**154,250**
NAME_NOT_RESOLVED	92,515
CONNECTION_REFUSED	34,188
HTTP_RESPONSE_CODE_FAILURE	8,196
SSL_PROTOCOL_ERROR	4,469
CONNECTION_RESET	2,938
SSL_VERSION_OR_CIPHER_MISMATCH	2,835
ADDRESS_UNREACHABLE	2,210
CONNECTION_CLOSED	1,674
TOO_MANY_REDIRECTS	1,473
SSL_UNRECOGNIZED_NAME_ALERT	967
ABORTED	775
HTTP2_PROTOCOL_ERROR	633
INVALID_AUTH_CREDENTIALS	591
EMPTY_RESPONSE	388
INVALID_RESPONSE	280
INVALID_REDIRECT	32
BAD_SSL_CLIENT_AUTH_CERT	31
SSL_SERVER_CERT_BAD_FORMAT	25
SOCKET_NOT_CONNECTED	17
HTTP2_INADEQUATE_TRANSPORT_SECURITY	6
QUIC_PROTOCOL_ERROR	2
SSL_KEY_USAGE_INCOMPATIBLE	2
RESPONSE_HEADERS_TOO_BIG	1
HTTP2_SERVER_REFUSED_STREAM	1
UNEXPECTED_PROXY_AUTH	1
Timeout	**60,738**
Unknown errors	**191**
$\sum$	**215.179**

TABLE 4: Errors while scanning the Tranco top 1M.

Category	#PL	#FL	#EL	Category	#PL	#FL	#EL	Category	#PL	#FL	#EL
Online Shopping	1,449	417	1	Interactive Web Apps	62	5		Parked Domain	11		
Business	1,391	226	1	Tech/Business Forums	60	23		Pharmacy	9	2	
Entertainment	771	317		Content Server	56	4		Prof. Networking	8	6	
General News	727	156		Art/Culture/Heritage	47	20		Tobacco	7		
Travel	669	218		Dating/Personals	46	27		Mobile Phone	7	3	
Unknown Category	608	46		Government/Military	45	13		Sexual Materials	5	3	
Internet Services	507	98		Social Networking	42	25		Web Ads	4		
Marketing/Merch.	467	179		Weapons	37	9		Media Downloads	4	1	
Education/Reference	394	112		Pot. Illegal Software	33	5		Anonymizers	4	1	
Games	284	152		Internet Radio/TV	29	11		School Cheating	3	1	
Gambling	279	44		Search Engines	28	2		Resource Sharing	3	1	
Forum/Bulletin Boards	256	39		Politics/Opinion	26	12		Gambling Related	3	3	
Fashion/Beauty	239	92		PUBs$^{\alpha}$	25	2		Digital Postcards	3	2	
Sports	237	67	1	Alcohol	25	6		Web Phone	2	1	
Blogs/Wiki	207	102		Religion/Ideologies	23	8		Text Translators	2		
Finance/Banking	201	32		Stock Trading	20	4		Nudity	2		
Job Search	181	16	3	Shareware/Freeware	20	9		Discrimination	2	1	
Health	167	50		Technical Information	18	4		Visual Search Engine	1	1	
Real Estate	157	33		Personal Pages	17	5		Malware	1	1	
Software/Hardware	147	37		Pers. Network Storage	17	4		Spam URLs	1		
Public Information	130	57		Media Sharing	17	10		Profanity	1	1	
Recreation/Hobbies	102	50		Web Mail	15	2		P2P/File Sharing	1		
Motor Vehicles	101	8		Humor/Comics	15	10		Instant Messaging	1		
Portal Sites	95	52		Major Global Religions	14	6		Information Security	1		
Advocacy/NGO	78	30		Streaming Media	13	5		Game/Cartoon Violence	1		
Auctions/Classifieds	76	19		Provocative Attire	13	6		Drugs	1		
Restaurants	68	20		Chat	13	6		Anonymizing Utilities	1	1	
Pornography	66	6		Malicious Sites	12						
								$\sum$	10,931	2,947	6

$^{\alpha}$ Potentially unwanted programs

TABLE 5: Website categories susceptible to SSO privacy leaks classified by Trellix [8].