






# SoK: SSO-MONITOR — The Current State and Future Research Directions in Single Sign-On Security Measurements


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

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

Tobias Wich   
ecsec GmbH  
tobias.wich@rub.de  
ORCID: 0000-0002-1180-1570

Christian Mainka   
Ruhr University Bochum  
christian.mainka@rub.de  
ORCID: 0000-0002-4273-645X

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

Vladislav Mladenov   
Ruhr University Bochum  
vladislav.mladenov@rub.de  
ORCID: 0000-0001-9208-9281

**Abstract**—Single Sign-On (SSO) with OAuth 2.0 and OpenID Connect 1.0 is essential for user authentication and authorization on the Internet. Billions of users rely on SSO services provided by Google, Facebook, and Apple. For large-scale measurements on the security of SSO, researchers need to reliably detect SSO implementations in the wild.

In this paper, we survey the current state of 36 SSO measurement tools from prior work and discover gaps leading to blind spots in the SSO landscape that hinder the community from improving large-scale research. We observe unreliable measurements and a lack of reproducibility, making comparisons between studies difficult, if not impossible.

We fill these gaps with SSO-MONITOR, our open-source, modular, and highly extensible framework for large-scale SSO landscape and security measurements. SSO-MONITOR achieves a high accuracy of 93% and, compared to previous tools, significantly improves the reliability of SSO measurements by 19%. It continuously takes snapshots of SSO implementations on the top 1M websites to compose an SSO archive that is reproducible by design. Therefore, it passively monitors the SSO flows and provides an extensive set of landscape and security insights on sso-monitor.me. Our SSO archive allows researchers to perform comprehensive measurements over time and even beyond the scope of SSO.

We use the data in our SSO archive to measure the security of 89k SSO authentication flows on the top 1M websites. Thereby, we discover 33k violations of OAuth Security Best Current Practices and 339 severe security vulnerabilities. They include 30 username and password leaks and 28 token leaks that allow full account takeovers.

**Index Terms**—Single Sign-On, Authentication, Authorization, OAuth, OpenID Connect, Web Archive, SSO Archive

## 1. Introduction

Single Sign-On (SSO) has become an essential part of the Internet. It allows users to log into multiple websites and services with a single set of credentials to reduce the burden of passwords. Today, the most widely adopted SSO protocols are OAuth 2.0 (OAuth) [35] and OpenID Connect 1.0 (OIDC) [93]. Major companies such as Google,

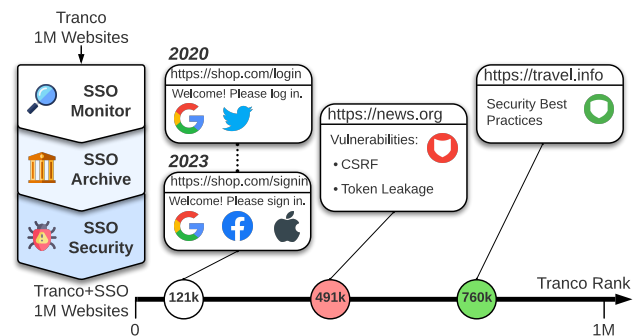


Figure 1: SSO-MONITOR. We continuously iterate over the Tranco 1M websites, detect their SSO flows, store snapshots in our SSO archive, and run security analyses.

Facebook, and Apple provide SSO services to billions of users worldwide. Over the last decades, a long line of research on the security of SSO has been published. Numerous studies uncovered crucial issues with SSO, including formal analyses [23, 24, 25, 26], web security [28, 29, 39, 48, 49, 52, 71, 99, 103, 109, 110], library security [54, 55, 68, 82], and privacy risks [47, 60, 62].

**Challenge of Reliable SSO Measurements.** To prove relevance, prior research investigated numerous SSO implementations on the Internet. Surprisingly, there is no consolidated approach for this. There are lists of popular websites, such as the Tranco top 1M list [45]. However, there is no public list or established approach to find the login pages of websites or detect their SSO logins. Prior work often misses details on how they discovered SSO or considers them out of scope. This is certainly reasonable, since their novelty lies in finding new attacks or methodologies to automatically discover and exploit them. However, reliable detection of SSO is crucial to accurately assess the impact of their attacks.

In this paper, we uncover gaps in the SSO detection algorithms used in previous work. For example, we found that SAAT [28] misses 38% of the SSO buttons. Previous research uses keywords like “Sign in with Google” or “Şununla giriş yap” (Turkish) to search for SSO buttons.

We show that this approach covers the SSO landscape to a great extent, but exhibits multiple drawbacks. For example, it strongly depends on the website's language [110], source code, and set of keywords. Furthermore, Jannett et al. [39] showed that Dual-Window Single Sign-On (DW-SSO) using In-Browser Communication (InBC) instead of HTTP plays a crucial role in the SSO ecosystem. Thus, we found that prior work misses SSO on 12,085 websites.

**Reliability** in SSO measurements is challenging, not well-studied, and exhibits multiple gaps. However, it is essential to accurately measure the impact of SSO security issues on the Internet. As there are no guidelines, we miss many details of the SSO ecosystem and how it evolves. SSO security researchers need a continuously updated list of SSO-enabled websites, similar to what the Tranco list provides for web security researchers.

### Challenge of Reproducible SSO Measurements.

Given the dynamic nature of the web and the reliance on third-party content, recent research [16, 17] has shown that web measurement studies are hardly reproducible. Hantke et al. [34] showed that web archives such as the Internet Archive<sup>1</sup> enable reproducible web security measurements. However, they found that large-scale analyses of *dynamic* websites are particularly difficult, if not impossible, to perform, for example, due to rate limitations. In particular, SSO heavily relies on dynamic execution with JavaScript and poses a significant challenge. The issue is further compounded because web archives often lack relevant data for SSO security research (they do not execute SSO). Therefore, they are not suitable for reproducible SSO measurements. Currently, there is no archive that provides regularly updated snapshots of SSO authentication flows over time. While web security researchers can rely on the Internet Archive, SSO security researchers must rely on the availability, completeness, and usefulness of artifacts.

In this paper, we systematically analyze and compare 36 research papers that contribute SSO measurement tools. Unfortunately, only four provide artifacts with limited data showing a rough view of their research. For example, we cannot measure false negatives if artifacts only provide true positives. Although published in the same year, Ghasemisharif et al. [28] used the Majestic list and Jannett et al. [39] used the Tranco list. In the constantly changing SSO ecosystem, this discrepancy of measurements becomes even more pronounced as time passes.

**Reproducibility** in SSO measurements is a focal point to gain new insights and opportunities for future security research. Artifacts from previous work are frequently unavailable, incomplete, and hard if not impossible to compare. SSO security researchers need a reproducible archive that continuously snapshots SSO flows and stores relevant data in a standardized format.

**SSO-MONITOR.ME: Our SSO Archive for Reliable and Reproducible Security Research.** This paper fills these gaps with SSO-MONITOR. It enables researchers to conduct reliable and reproducible large-scale SSO measurements over time, as shown in Figure 1. In general, we

increased the reliability of SSO detection by 19%. Therefore, we measured existing techniques, discovered gaps, elaborated on improvements, and proposed novel ones. Our SSO archive lists all analyzed websites, their SSO flows, landscape statistics, and gathered data, including HTTP traffic. The latter is particularly useful for artifact evaluations and helps future research estimate the impact of a specific research target back in time. SSO-MONITOR is highly customizable and supports 57 configuration options for scanning websites, such as different browser engines and various methods to discover the login page of a website. It has a modular architecture to facilitate future extensions. As a demonstration, we implemented two modules for security measurements.

(1) *SSO Security Best Current Practices*: Inspired by previous security research [24, 28, 29, 55, 91, 100, 110] and the Security Best Current Practices (SBCPs) [51], we systematize known threats and show how to automate their analysis entirely. We apply a passive flow analysis approach that reveals crucial deviations from the SBCPs.

(2) *Extending Measurements on DW-SSO*: Recently, Jannett et al. [39] revealed the importance of DW-SSO, which is based on InBC instead of HTTP. We demonstrate how to extend their security evaluation using our SSO archive from the Tranco top 1k to the Tranco top 1M.

**Contributions.** We make the following contributions:

- *Current State of SSO Research*: We systematize and compare the reproducibility, reliability, and coverage of security compliance tests of 36 SSO measurement tools published across the past ten years. Table 1 summarizes the gaps identified in previous work. (§3)
- *Reliable SSO Detection*: We improve two existing techniques and introduce three novel ones in Table 5. We can automatically detect 93% of the SSO buttons in our ground truth, having zero false positives. (§4)
- *SSO-MONITOR*: We monitor the SSO ecosystem on an ongoing basis, detect SSO buttons, take snapshots of SSO flows, and store all data in our SSO archive. Our public service provides an overview of historical scans, landscape and security statistics, and artifacts. SSO-MONITOR is open-source<sup>2</sup> and dockerized. (§5)
- *Large-Scale Landscape Measurement*: We present an empirical landscape evaluation of the Tranco top 1M websites. SSO-MONITOR is continuously running since July 2023 and makes all data available on sso-monitor.me. In sum, we detected 88,994 SSO buttons on 45,532 websites. Our SSO archive provides 5 TB of data to support future security research. (§6.1)
- *Large-Scale Security Measurement*: We found that less than every fourth website is compliant with the Security Best Current Practices (10,527 of 45,532). Violations included 34,333 SSO flows that are susceptible to CSRF and 71 leaked OAuth credentials. In 2017, IETF recommended avoiding the *implicit* variants of OAuth because tokens are likely to leak. However, we found 4,111 flows that still use it. (§6.2)
- *Future Directions in SSO Research*: We show that SSO-MONITOR is a generic platform for state-of-the-art SSO security research. Therefore, we automated recently published work [39]. Our large-scale dual-window analysis revealed 339 severe se-

1. Also called Wayback Machine. Available at <https://archive.org>.

2. Available at <https://github.com/RUB-NDS/SSO-Monitor>.

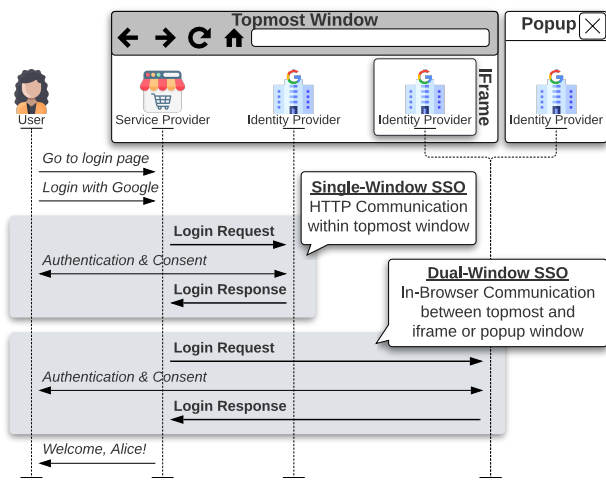


Figure 2: Basic SSO Login Flow. The user wants to log in on the SP’s website, i.e., shop.com. The SP delegates the user’s authentication to the IdP. The IdP issues authentication tokens that the SP uses to authenticate the user.

curity and privacy vulnerabilities, including account takeovers and private data leakage at scale. Our SSO archive fosters in-depth analyses and helps to recognize trends beyond SSO. Based on the current state of research and the applicability of SSO-MONITOR, we suggest several directions for future research. (§7)

**Responsible Disclosure.** We have started the coordinated disclosure of the identified issues (cf. §8.1). We currently participate in active discussions and appreciate being promptly acknowledged and rewarded.

## 2. Background: Single Sign-On Login Flow

Figure 2 shows a simplified Single Sign-On (SSO) login flow in which a user wants to log into a Service Provider (SP)’s website, i.e., shop.com, using her account at an Identity Provider (IdP). Therefore, she visits the SP’s login page and clicks on the “Login with Google” button to initiate the SSO login flow with her Google account.

**Login Request.** The SP issues a login request to the IdP via the user’s browser. In Single-Window Single Sign-On (SW-SSO), the SP overwrites the topmost window with the login request. In Dual-Window Single Sign-On (DW-SSO), the SP opens the login request in an iframe embedded in the topmost window (→ iframe flow) or in a popup window (→ popup flow). The login request contains parameters that are specific to the particular SSO protocol. For example, in OAuth 2.0 (OAuth) [35] and OpenID Connect 1.0 (OIDC) [93], it contains the identity of the SP, the target URL to which the IdP issues the login response, and optional security and privacy parameters.

**Authentication & Consent.** Before the login response is returned to the SP, the user must authenticate on the IdP. In OAuth and OIDC, the user also needs to grant consent to allow the SP to access her resources on the IdP. For example, the user can allow the SP to access her name, email address, and calendar. The IdP only requires the user’s consent once for each SP. If the user is authenticated on the IdP and has already provided consent

for the SP, this step is skipped and the IdP issues the login response immediately without any user interaction.

**Login Response.** Thereafter, the login response is returned to the SP via the user’s browser. It contains authentication tokens that the SP uses to authenticate the user or get access to the user’s resources. For instance, the `id_token` is a signed JSON Web Token (JWT) that contains claims about the user’s identity. If leaked, the attacker may take over the user’s account on the SP. In SW-SSO, the IdP uses HTTP Communication such as 302 redirects or HTML `<form>` submissions to overwrite the topmost window with the login response. In DW-SSO, the IdP uses In-Browser Communication (InBC), that is, native browser APIs like `postMessage` [104, §9.3] or *Channel Messaging* [104, §9.4] to send the login response from the iframe or popup to the topmost window. Note that the popup cannot use HTTP to communicate across two distinct browser windows. Instead, it must use JavaScript and InBC to pass the login response to its opener window.

**SSO Integrations.** For implementing SSO, website developers can integrate the IdP-provided SDKs or communicate with the API endpoints of the IdPs. In theory, SDKs are secure out-of-the-box and aim to reduce the implementation efforts of developers. When using APIs, developers gain more flexibility, but they have to take extra care to protect their SSO flows by following the OAuth Security Best Current Practices (SBCPs) [51]. This document covers all threats that are relevant to SSO developers and suggests best practices to mitigate them.

## 3. SSO Measurements: The Current State

Our research began with an exploration of related work measuring SSO implementations on websites and apps. Early studies relied mainly on manual methods to assess a limited number of SPs and IdPs. However, research has shifted towards large-scale measurements using automated tools. To compare the increasing number of tools, we systematically categorized them. We identified several shortcomings in reproducibility (§3.1), reliability (§3.2), and coverage of security compliance tests (§3.3). To the best of our knowledge, this section provides the first systematization of SSO measurement tools, see Table 1.

**Methodology and Scope.** Our initial goal was to understand the current state of the art concerning automated SSO landscape and security measurements. Therefore, we conducted an extensive literature survey of the past ten years. We examined the proceedings of top-tier venues dedicated to security, measurement, and web research (cf. Table 2). We searched for common SSO terms such as “single sign-on”, “oauth” and “openid connect” in the titles and, if available, abstracts. We filtered out all SSO papers that (1) do not consider the state-of-the-art protocols used on the web: OAuth and OIDC, (2) instead consider deprecated or enterprise protocols like BrowserID or SAML, (3) propose new proprietary protocols, (4) conduct formal security proofs without measuring the real

3. <http://portal.core.edu.au/conf-ranks/?search=security&by=all&source=CORE2023&sort=arank&page=1>

4. <http://portal.core.edu.au/conf-ranks/?search=measurement&by=all&source=CORE2023&sort=arank&page=1>

5. <http://portal.core.edu.au/conf-ranks/?search=web&by=all&source=CORE2023&sort=arank&page=1>

			Reproducibility (§3.1)				Reliability (§3.2)				Security (§3.3)									
			Artifact Availability				Detection				Tooling									
			SPs	Tool	Data	Archive	IdPs	LP	SSO	Communication	HTTP	InBC	Targets	IdP	⚡	🛡️	🔍	👤	Attack	👤
[99]	🛡️	BRM-Analyzer	S&P'12	11	○	○	○	🧒 +2	○	○	●	○	●	○	○	○	○	○	○	○
[4]	🛡️	AuthScan	NDSS'13	8	○	○	○	🧒 +2	○	○	●	○	●	○	○	○	○	○	○	○
[105]	🛡️	InteGuard	NDSS'13	5	○	○	○	🧒 +1	○	○	●	○	●	○	○	○	○	○	○	○
[97]	🛡️	IdentMgmtRelations	PAM'14	Alexa 35k	○	●	○	IdP-agnostic	●×2	●×1	○	○	—	—	—	—	—	—	—	—
[110]	🛡️	SSOScan	USENIX'14	Quantcast 20k	●	○	○	🧒	●×2	●×1	●	○	●	○	○	○	○	○	○	○
[53]	🛡️	EsPreSSO	OIS'15	—	●	○	○	—	○	○	●	○	●	○	○	○	○	○	○	○
[78]	🛡️	OAuth-Detector	DIMVA'15	Alexa 10k	●	○	○	IdP-agnostic	●×2	●×1	●	○	●	○	○	○	○	○	○	○
[98]	🛡️	AuthDroid	ACSAC'15	Chinese App Markets 4k	○	○	○	15	○	○	●	○	●	○	○	○	○	○	○	○
[88]	🛡️	MPWA-ZAP	NDSS'16	Missing info	👤	○	○	🧒 +3	○	○	○	○	●	○	○	○	○	○	○	○
[109]	🛡️	OAuthTester	AsiaCCS'16	Alexa 500	○	○	○	🧒 +3	○	○	○	○	●	○	○	○	○	○	○	○
[58]	🛡️	MDiscoveryService	arXiv'16	2 libraries	👤	○	○	—	○	○	○	○	○	○	○	○	○	○	○	○
[55]	🛡️	ProESSOS	EuroS&P'17	8 libraries	○	○	○	—	○	○	○	○	○	○	○	○	○	○	○	○
[108]	🛡️	MobileAppAuth	ACNS'17	SJ.QQ+GPlay 600	○	○	○	🧒 +1	○	○	○	○	●	○	○	○	○	○	○	○
[89]	🛡️	CSRF-checker	EuroS&P'17	Alexa 432	👤	○	○	—	○	○	○	○	●	○	○	○	○	○	○	○
[111]	🛡️	AuthScope	CCS'17	GPlay 200k	○	○	○	🧒	●×2	●×1	●	○	○	○	○	○	○	○	○	○
[7]	🛡️	WPSE	USENIX'18	Alexa 100k	👤	○	○	🧒 +78	●×2	●×1	○	○	○	○	○	○	○	○	○	○
[29]	🛡️	SSOff	USENIX'18	Alexa 1M	○	●	○	🧒 +63	●×3	●×1	●	○	○	○	○	○	○	○	○	○
[107]	🛡️	S3KVetter	USENIX'18	10 SDKs	●	○	○	—	○	○	○	○	○	○	○	○	○	○	○	○
[47]	🛡️	OAuthGuard	SSR'19	Majestic 1k	○	○	○	G	○	○	○	○	○	○	○	○	○	○	○	○
[69]	🛡️	OAUTHLINT	ASE'19	GPlay 600	○	○	○	🧒 +18	○	●×1	○	○	○	○	○	○	○	○	○	○
[90]	🛡️	OVERSCAN	NSS'19	45	○	○	○	🧒	○	○	○	○	○	○	○	○	○	○	○	○
[79]	🛡️	MoSSOT	AsiaCCS'19	Wandoujia+Apkpure 24k	●	○	○	🧒 +2	●×1	●×1	○	○	○	○	○	○	○	○	○	○
[41]	🛡️	Shepherd	MADWeb'20	Alexa 10k	○	○	○	🧒	●×4	●×1	○	○	—	—	—	—	—	—	—	—
[18]	🛡️	Cookie-Hunter	CCS'20	Alexa 1.5M	👤	○	○	🧒	●×2	●×1	○	○	—	—	—	—	—	—	—	—
[5]	🛡️	OCSRF	DIMVA'21	Alexa 50k	○	○	○	🧒	●×2	●×1	○	○	○	○	○	○	○	○	○	○
[103]	🛡️	MoScan	ISSTA'21	Moz 500	●	○	○	🧒 +3	○	○	○	○	○	○	○	○	○	○	○	○
[59]	🛡️	OAuthScope	WPES'21	Alexa 500 × 5	○	○	○	🧒 🍏 +1	●×2	●×1	○	○	—	—	—	—	—	—	—	—
[28]	🛡️	SAAT	S&P'22	Majestic 100k	👤	👤	○	🧒	●×2	●×1	○	○	○	○	○	○	○	○	○	○
[42]	🛡️	SameSite	S&P'22	Alexa 10k	○	○	○	🧒 🍏 +10	●×2	●×1	○	○	—	—	—	—	—	—	—	—
[40]	🛡️	AuthPermCollector	IFIP'22	Ax+Mj+Tc+Cisco 14k	○	●	○	🧒 🍏 +1	●×2	●×1	○	○	—	—	—	—	—	—	—	—
[67]	🛡️	OAuch	RAID'22	—	●	○	○	🧒 +98	○	○	○	○	○	○	○	○	○	○	○	○
[39]	🛡️	DISTINCT	CCS'22	Tranco 1k	●	○	○	🧒 🍏	○	○	○	●	○	○	○	○	○	○	○	○
[68]	🛡️	Cerberus	CCS'22	—	○	○	○	10 libraries	○	○	○	○	○	○	○	○	○	○	○	○
[33]	🛡️	QuoVadis	AINA'23	Majestic 100k	○	○	○	IdP-agnostic	●×1	●×1	●	○	—	—	—	—	—	—	—	—
[61]	🛡️	SSOPrivateEye	arXiv'23	Tranco 500	○	○	○	🧒 🍏	○	●×1	○	○	—	—	—	—	—	—	—	—
[37]	🛡️	OAuthPlayer	ACSAC'23	Tranco 15k	●	○	○	🧒 +15	●×2	●×1	●	○	○	○	○	○	○	○	○	○
	🛡️	SSO-MONITOR	EuroS&P'24	Tranco 1M	●	●	●	🧒 🍏 +7	●×5	●×5	●	●	○	○	○	○	○	○	○	○

LP Login Page ⚡ Offensive tool 🛡️ Defensive tool 🛡️ Security compliance tool 🚶 Active attack 🚶 Passive attack

TABLE 1: Systematization of SSO Measurement Tools. We found that 6 of 36 tools are available (upon request) and enable large-scale analyses by automating the detection of login pages and SSO on websites (cf. gray-shaded rows). However, we identified gaps in their reproducibility, reliability, and coverage of security compliance tests.

Research Field	Rank	Conference and Proceedings
Security <sup>3</sup>	A*	S&P, USENIX, CCS, NDSS
	A	ACSAC, ESORICS, AsiaCCS
Measurement <sup>4</sup>	A*	SIGMETRICS
	A	ESEM, IMC
Web <sup>5</sup>	A*	WWW
	A	WSDM, ICWSM, ISWC, ICWS

TABLE 2: Reputable conferences dedicated to security, measurement, and web research. We skimmed through their linked proceedings to discover SSO measurement tools. We did not include ICSPC, CSF, FC, and ASI-ACRYPT. They are focused on specific areas such as pervasive computing, formal analyses, financial, and crypto.

world, (5) use manual methods, (6) conduct user surveys, (7) focus on different application scenarios such as IoT. We skimmed through the abstracts, introductions, and methods of all remaining candidate papers to determine whether they contributed a new tool. We only considered tools that were presented as a clear contribution in the paper. We did not consider prototype implementations or the use of tools from third parties. If a paper introduced a tool, we further surveyed the paper's references to include additional related work from lower-ranked or unranked venues and publishers. Following this methodology, we could double our scope from 18 tools published on top-tier venues to 36 tools. The first paper dates back to 2012. We do not anticipate relevant prior research as OAuth was first standardized in 2012, followed by OIDC in 2014.

### 3.1. Reproducible SSO Measurements

We use the definitions of the Association for Computing Machinery [3] for (1) repeatability ("Same team, same experimental setup"), (2) reproducibility ("Different team, same experimental setup"), and (3) replicability ("Different team, different experimental setup"). By definition, repeatability can only be achieved by the researchers who performed the measurement. Given the dynamic nature of the web and its dependence on third-party content, recent research [16, 17] has shown that web measurement studies are hardly replicable. Thus, we focus on *reproducibility* and leave repeatability and replicability aside. To reproduce previous SSO measurements, we need the lists of analyzed SPs, the tools (i.e., in the form of source code), and the raw data (i.e., traces of HTTP traffic).

**Lack of Lists.** Measurements on websites relied on six top sites lists: Tranco [45], Moz [94], Majestic[56], Cisco Umbrella [11], and the deprecated Alexa and Quantcast lists. Only Tranco improves the reproducibility of research results by providing permanent references to its lists. Unfortunately, only two tools [37, 39] operate on a *referenced* Tranco list, and three tools [29, 40, 97] provide their lists as artifacts. Measurements on mobile apps relied on the Google Play Store [1], APKPure [44], and three Chinese app stores. For a total of 21 measurements, researchers have no access to the list of analyzed SPs by any means.

**Lack of Available Tools.** Only 12 of the 36 tools are publicly available (●). Two tools are provided upon request (○) and three tools have vanished, e.g., the URL is not reachable (○). The unavailability of tools hinders



progress in SSO research. It leads to redundant work on already solved tasks, i.e., related work reimplements SSO automation pipelines instead of reusing existing ones.

**Lack of Executable Tools.** Out of 12 available tools, six enable *large-scale* measurements on *websites* by automating the detection of login pages and SSO (cf. gray-shaded rows in Table 1). We did not receive access to one tool upon our request. We successfully executed two tools and evaluated their reliability (cf. §3.2). The remaining tools require modifications for decade-old or suspended browsers, or threw exceptions due to missing files.

**Lack of Relevant and Updated Data.** Only three papers share their raw data (●). We obtained access to another dataset upon request (◐). During the analysis, we identified three problems. First, some datasets [40, 97] provide only the IdPs for each domain but lack the login page URLs. Second, the datasets are insufficient to support future work. For example, they miss details for security analyses. Third, the datasets are highly volatile due to frequent changes in login pages and SSO. The only solution to address this problem is to have an archive that continuously captures snapshots of login pages and SSO.

**Enhancing the Reproducibility:** We introduce the first SSO archive that is reproducible by design. To fill the archive, SSO-MONITOR continuously iterates over the Tranco list, snapshots their SSO flows, and stores the data. We provide the source code and a public service.

### 3.2. Reliable SSO Measurements

When comparing the results of prior work, we identified conflicting statements due to deficiencies in reliability. These gaps are mainly due to prioritizing measurements on SSO-enabled websites over their comprehensive detection. For example, Calzavara et al. [7] admit that their tool is “not meant to provide a comprehensive coverage of the deployment of OAuth 2.0 on the web”. Assessing the reliability of all tools was largely successful, except one tool [96] that lacked methodological details and code.

**IdP Coverage.** Previous surveys assessed an average of 13 IdPs, with a median of 3. Facebook and Google stand out as the most prevalent IdPs, appearing in 25 and 15 surveys, respectively. Three papers [7, 29, 67] offered an extensive perspective by analyzing 65, 80, and 100 IdPs. We identified conflicting statements in previous studies related to the coverage of IdPs. For example, Grüner et al. [33] document that RTBMarket is the third most used IdP, which is inconsistent with all previous studies. The authors admit that this statistic appears to be inaccurate. According to [28, 29, 33], Facebook is the most prevalent IdP, which contrasts with Jannett et al. [39] and this paper. We found that this disagreement mainly arises from accurately measuring InBCs, which are widely used in Google SSO. In addition, the time frame of the research is important. During early experiments in 2022 (after the acquisition of Twitter [20]), we found that SSO with Twitter was frequently removed or non-functional.

**IdP-Centric vs. IdP-Agnostic.** Most studies adopted the IdP-centric approach, which identifies a predetermined set of IdPs using unique patterns for each IdP. Only a few papers [33, 78, 97] applied the IdP-agnostic approach,

which identifies all IdPs using general patterns that apply to all IdPs. The IdP-agnostic approach offers flexibility, but suffers from low reliability. First, prior studies [33, 97] report an inevitable increase in false positives. Second, it fails to recognize IdPs that do not exhibit general patterns.

**Login Page Detection.** Previous research employed an average of two techniques to locate the login page of a website. Among these strategies, crawling for login-related links significantly outnumbers other methods. Four studies deemed search engines useful, while three probed known login URLs. The extensive variety implies a fundamental challenge: determining the most consistent approach or combination of techniques. Only Jonker et al. [41] examined the effectiveness of four strategies for locating pages with username and password credentials. Our research augments their findings by evaluating five methodologies for identifying pages with SSO logins, encompassing a new approach and two enhanced ones.

**SSO Detection.** In contrast, the reliability of detecting SSO buttons remains largely unexplored. Prior work relied on a *single* approach: using string matching algorithms to identify keywords, i.e., “login with facebook”. This approach suffers from false negatives because SSO buttons can have various shapes and forms. For example, they can be nested structures like `<a><img/><span/></a>`. Our research revealed that previous methods often overlooked nested elements such as `<span>` and `<div>`. Moreover, keywords make this approach heavily reliant on the language and the specific set of chosen keywords.

**Unreliability of Data.** We discovered conflicting results that make it difficult, if not impossible, to compare different studies. For example, Ghasemisharif et al. [28] identified 2,689 Facebook SSO buttons on the Majestic top 100k. In the same year, Khodayari et al. [42] reported 3,328 Facebook SSO buttons on the Alexa top 10k. Finding more SSO buttons on a 10× smaller set is remarkable and warrants further investigation.

**Unreliability of Tools.** We executed two tools [28, 37] to compare their reliability with SSO-MONITOR on 1k websites (cf. Table 4). For detecting Facebook, we outperform both tools by factors of two [28] and five [37].

**Large-Scale InBC Analysis.** Most of prior work analyzed the HTTP traffic in SSO flows. Others [68, 69] statically analyzed the source code of SSO implementations. To the best of our knowledge, Jannett et al. [39] were the first to perform a semi-automatic analysis of InBC in DW-SSO. We enhance their findings by fully automating the analysis of InBC across millions of websites.

**Enhancing the Reliability:** We evaluate 10 login page and SSO detection techniques, improve two of them, and introduce three novel ones. As a result, we discovered more than 17k previously unknown SSO buttons.

### 3.3. SSO Security Measurements

Most of the research papers (27 of 36) contribute an offensive tool (🔧) that runs active attacks (👤) on SP implementations. However, we found gaps in the coverage of the security compliance tests (🛡️), which lead to blind spots in previous security measurements.

**SSO Security Best Current Practices.** The SBCPs are maintained in an active draft [51], which has been

updated 24 times in the last six years. Multiple new attacks, mitigations, and security extensions have been incorporated. For example, the Proof Key for Code Exchange (PKCE) security extension was standardized in 2015 [72], and has been recommended since 2019 [51, §2]. The mitigation of mix-up attacks has been included since 2018 [51, §4.4] and was standardized in 2022 [75]. In the same year, Jannett et al. [39] discovered attacks on InBC flows, which have recently been merged with the SBCP in early 2023 [51, §4.18]. The dynamic nature of this draft requires tools to adapt constantly to changes.

**Gaps in Security Compliance Tests.** The vast majority of tools have measured CSRF protections in OAuth and OIDC. For mitigation, the SBCP recommends using random parameters, specified as `state` or `nonce`, or implement PKCE. Prior tools checked if the `state` is used [78, 90], if its value is random [7], and if its value is validated [5, 103, 109]. To our knowledge, the tools focused on OAuth and missed measuring the `nonce` parameter, which provides similar mitigation for OIDC. PKCE validation was only measured on the IdPs [67, 68], but its use on SPs remains unclear. Beyond CSRF, Zhou et al. [110] measured insecure flows and secret leaks. Although they did not find any leaked secrets, we identified 71 of them 10 years later.

**Enhancing the Compliance Coverage:** Constant updates to cover new threats and mitigations are critical. Prior work does not measure the adoption of PKCE, mix-up attacks, and InBC security at scale. Updating the CSRF compliance tests is necessary as well. Our measurement aligns with the latest SBCP [51], showing that only 10,527 of 45,532 SPs are compliant (23%).

## 4. Reliable Single Sign-On Detection

For large-scale web measurements, researchers implemented various techniques to identify login pages and SSO buttons. Unfortunately, there is still considerable ambiguity concerning their reliability (cf. §3.2). In this section, we enhance existing approaches, propose novel ones, and measure their reliability. Our goal is to optimize the automated detection of login pages and SSO buttons.

### 4.1. Manual vs. Automatic SSO Detection

Building a ground truth allows us and other researchers to estimate the success rate and the reliability of individual login page and SSO detection techniques.

**Methodology.** We visited the top 1k Tranco<sup>6</sup> [45] sites in June 2023 and checked whether they are reachable, provide one or multiple login pages, and support SSO. On foreign sites, we used Chrome’s language translation. We clicked on login-related links (depth 2) and issued queries to Google to search for login pages. For each login page, we determined its support for a publicly accessible IdP with state-of-the-art SSO protocols like OAuth and OIDC. We further analyzed if the SSO (1) is functional or throws errors, (2) requires user interaction (e.g., submitting email), (3) is executed in the topmost, popup, or iframe window, and (4) is an API or SDK integration.

6. Available at <https://tranco-list.eu/list/6Z2X>.

	Web-sites	SSO Buttons								Integration	
		All	⚠	👤	⚙	🪟	📄	🗑	📌	API	SDK
SSO	Google (G)	276	331	2	17	312	141	133	56	206	124
	Facebook (f)	213	213	9	15	189	103	106	0	158	51
	Apple (a)	155	155	0	6	149	96	59	0	112	43
	Twitter (t)	51	51	13	9	29	23	15	0	38	0
	QQ (Q)	28	28	0	12	16	14	14	0	28	0
	GitHub (G)	27	27	0	14	13	25	2	0	27	0
	Microsoft (M)	25	25	0	3	22	20	5	0	25	0
	LinkedIn (in)	23	23	2	3	18	10	12	0	22	0
	WeChat (W)	21	21	0	8	13	12	9	0	21	0
	Sina Weibo (S)	21	21	0	8	13	10	11	0	21	0
	All	347	895	26	95	774	454	366	56	658	218
	Other IdPs (×37)	...	...	...	...	...	...	...	...	...	...
	All	351	988	27	115	846	505	407	56	751	218
SSO or Password		750	No Auth.	132	Not Reachable	118	Total	1k			

⚠ Broken 👤 User Interaction ⚙ Automatable 🪟 Topmost 📄 Popup 🗑 IFrame

TABLE 3: Ground Truth of the Tranco Top 1k websites. We found 988 SSO buttons on 351 websites (35%). Due to broken SSO buttons or required user interaction, we can automate 846 (86%) of them. *Continued in Table 8.*

	All	SSO Buttons									
		G	f	a	t	M	in	W	S	Q	S
Ground Truth (G)	774	312	189	149	29	16	13	22	18	13	13
SSO-MONITOR	716	294	175	139	23	22	16	13	12	11	11
False Negatives	Timeouts	21	6	6	3	2	0	2	1	0	1
	CAPTCHAs	18	5	5	5	2	0	1	0	0	0
	Banners	7	4	2	0	0	0	1	0	0	0
	≠ 200 OK	6	1	1	0	1	0	0	1	0	1
	Crashes	3	1	0	0	1	0	0	0	0	1
	Blockages	2	1	0	1	0	0	0	0	0	0
	> Viewport	1	0	0	1	0	0	0	0	0	0
	All	58	18	14	10	6	0	2	3	1	2

TABLE 4: SSO-MONITOR’s False Negatives. We could automatically detect 93% of the SSO buttons on the Tranco top 1k sites. In sum, 58 SSO buttons could not be detected due to timeouts, CAPTCHAs, cookie banners, erroneous status codes, Playwright crashes, website blockages, and buttons exceeding the browser’s viewport.

**Ground Truth.** In total, 75% of the top 1k websites offer login pages and 35% support SSO (cf. Table 3). The remaining sites do not offer user authentication (13%) or are not reachable (12%). Websites can include multiple buttons for supporting SSO with different IdPs. Overall, we found 988 SSO buttons from 47 IdPs (cf. Table 8), with the top 10 IdPs providing 895 of them. The three most widely supported IdPs are Google, Facebook, and Apple, which is in line with recent work [39, 59] but differs from former research [28, 29, 33]. Surprisingly, developers prefer custom API integrations over the use of SDKs. If we consider only Google, Facebook, and Apple, DW-SSO with popups and iframes is more prevalent than SW-SSO. This is in line with prior work [39]. SW-SSO still exceeds DW-SSO if we consider all IdPs. Note that we could not determine the windows and integrations for some broken SSO buttons. SSO buttons throwing exceptions or requiring user interaction impede the automated detection. We decided to implement the top 10 IdPs to cover 99% of all SSO-enabled websites. Thus, SSO-MONITOR should automatically detect 774 SSO buttons on 306 websites.

**SSO-MONITOR.** Table 4 shows that SSO-MONITOR automatically identified 716 of 774 SSO buttons on the Tranco top 1k websites (93%). We missed most of the 58 false negatives due to timeouts, CAPTCHAs, and cookie banners. We have not encountered false positives because we verify if SSO flows are initiated (see §5.2.2).

## 4.2. Reliable SSO Detection Techniques

Table 5 compares the reliability of five login page detection and five SSO detection techniques. Therefore, we executed one scan for each technique (10 scans in total) on the Tranco top 1k websites. We measure the overall reliability of each technique compared to the ground truth and based on the total number of identified SSO buttons. We also highlight the uniqueness of each technique by measuring the hits that only this technique can achieve.

**Login Page Detection.** The first part of Table 5 shows that most of prior work crawled websites to determine their login pages. The crawling process often starts on a website’s homepage, which, according to our measurements, only holds SSO buttons with 8% probability. Instead, our results indicate that search engines are the most reliable technique, followed by crawling and testing common paths and subdomains. Although crawling and testing well-known URLs perform similarly well, the latter contributes with more SSO not found by other techniques.

We contribute to three techniques as follows (⊕):

(1) *Search Engines:* Although prior work used different search engines, including Bing [7, 28, 41], DuckDuckGo [28, 29], Startpage [28, 41], and Ask.com [41], we found two drawbacks: First, search engines are blocking IPs or triggering CAPTCHAs for rate-limiting. Second, search engines have different and constantly changing interfaces. Each interface requires an individual parser to extract the search results, introducing significant engineering efforts. Thus, we recommend using metasearch engines that return accurate results even if individual search engines are failing due to rate limitation. They distribute the load to more engines, provide convenient APIs, and outsource the parsing of search results from researchers to the open-source community. Out of nine search engines, we found that Qwant (406), Bing (365), and Google (310) are most reliable. On average, the login page was included in the 1.7<sup>th</sup> search result (334x1<sup>st</sup>, 74x2<sup>nd</sup>, 37x3<sup>rd</sup>, ...).

(2) *Paths and Subdomains:* Prior work tested paths that are well-known to appear in login page URLs, such as /login and /signin. We extended this technique to subdomains, increasing the overall number of SSO buttons by 11% from 309 (only paths) to 349 (paths and subdomains).

(3) *Sitemap:* Crawling is quite invasive and puts an excessive load on the servers. If available, a sitemap can reduce the hundreds of requests produced by crawling down to a single request for the `sitemap.xml` file or a few requests for nested sitemaps. Ideally, they list the URLs that a crawler would find on the site. In practice, we found that sitemaps do not provide the login page reliably.

**SSO Detection.** The second part of Table 5 shows that prior work searched for keywords to detect SSO buttons. The majority considered the website’s DOM tree, and one paper [28] inspected the accessibility tree. Our measurement shows that the DOM provides the most reliable information (68%). Detecting logos in screenshots and keywords in the accessibility tree attains similar accuracy. However, the logo detection contributes more SSOs that cannot be found by other techniques (4%). Matching patterns on InBCs contributes most SSO buttons, as it uniquely identifies certain SSO SDKs (14%).

We introduce three novel techniques as follows (⊕):

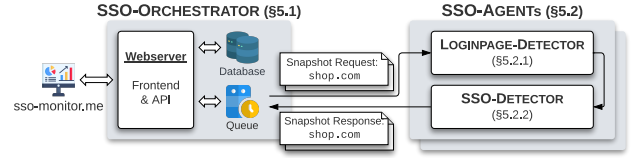


Figure 3: SSO-MONITOR’s Design and Architecture. The SSO-ORCHESTRATOR coordinates the continuous archiving, connects to long-term storage, and provides the snapshots. Multiple SSO-AGENTS retrieve snapshot requests from the queue to analyze a specific domain.

(1) *Logos of IdPs in Screenshot:* Searching for keywords depends on the website’s language, source code, and set of keywords. The DOM introduces additional complexity with nested iframes and shadow DOMs. The accessibility tree reduces this complexity and is less intrusive (fewer CAPTCHAs). However, it does not provide additional information than the DOM and does not detect any SSO buttons beyond the DOM. In addition, it suffers from inaccessible websites, e.g., aliexpress.com. Thus, we propose a novel visual-based SSO detection technique that operates independently of the website and browser. The main challenge was to find a fast but reliable approach that detects IdP logos of various sizes, styles, and colors in screenshots of the login page. Thereby, we could add 30 SSO buttons that none of prior work was able to detect.

(2) *Patterns in InBCs:* Jannett et al. [39] used a semi-automatic method for identifying predefined patterns<sup>7</sup> in URLs *after* they manually clicked the SSO buttons. In contrast, our approach is fully automatic. We compose a new set of patterns and match them on the InBCs to detect SSO buttons *before* they are clicked, i.e., on page load.

(3) *API Calls in Browser:* We are the first to measure the FedCM API on websites. This API was shipped in Chrome 108 in late 2022 [22] and is not widely used today. However, even subtle changes in SSO can have a significant impact. For instance, SSO with Apple was introduced in 2019 but quickly became one of the most prevalent IdPs. We expect to monitor a similar adoption of the FedCM API in 2024, following Chrome’s third-party cookie phaseout. Google already recommends developers to migrate to FedCM [32]. Adjusting the default configuration of SDKs will have an instant effect on the SSO landscape and security that we look forward to measuring.

## 5. SSO-MONITOR.ME: Our SSO Archive for Reproducible and Reliable Security Research

This section puts the SSO detection into practice with SSO-MONITOR, our tool that monitors millions of websites to compose a reproducible and reliable SSO archive. SSO-MONITOR consists of two components, as shown in Figure 3: SSO-ORCHESTRATOR (§5.1) and SSO-AGENT (§5.2). Together, they automate the continuous archiving, login page detection, and SSO detection.

7. [https://github.com/RUB-NDS/DISTINCT/blob/master/src/distinct-browser/distinct-chrome-extension/src/content\\_sdk.js](https://github.com/RUB-NDS/DISTINCT/blob/master/src/distinct-browser/distinct-chrome-extension/src/content_sdk.js)




Technique	Description	Prior Work	# SSO Buttons Unique	Total
<b>Part 1: Login Page Detection Techniques</b>				
⊕ Search Engines	Search engines crawl the web with an infinite depth, instantly provide up-to-date results, make the data searchable via keywords, and use internal rankings to provide optimized results. Metasearch engines perform a majority-vote on aggregated search results from multiple search engines and return accurate results even if requested search engines are failing.	[7, 28, 29, 41]	92 (12%)	477 (62%) → 246 LPs
Crawling	Crawling traverses the DOM tree of a website’s homepage and subpages for keywords referencing the login page, such as “login” and “account”. Passive crawling only considers the data that is available on the crawled page, such as href attributes of <a> elements. Active crawling clicks on links, buttons, and other elements in an instrumented browser to trigger navigations, the opening of new popups, and JavaScript events.	[5, 7, 18, 29, 33, 37, 40, 41, 42, 59, 78, 79, 97, 110, 111]	45 (6%)	360 (47%) → 159 LPs
⊕ Paths and Subdomains	The URLs of login pages often exhibit similar patterns in their paths and subdomains, such as shop.com/login and login.shop.com. While they do not invariably reflect the login page in all cases, they may still redirect to the login page. Requesting a set of well-known paths and subdomains that do not generate error messages may reveal the website’s login page.	[28, 29, 41]	52 (7%)	349 (45%) → 155 LPs
Homepage	Websites embed SSO buttons straight on their homepage to allow seamless sign-ins and sign-ups without requiring users to visit a dedicated login page. For instance, Google even recommends developers to embed the Google One Tap SDK on their homepages [80].	[5, 18, 37, 40, 41, 42, 59, 78, 97, 110, 111]	8 (1%)	60 (8%) → 46 LPs
⊕ Sitemap	Sitemaps are XML files listing all URLs of a website, along with additional metadata such as their importance, last updates, and change frequency [81]. They were particularly designed to inform search engines about the pages that are available for crawling, including the login page.	–	1 (0%)	26 (3%) → 16 LPs
			Ground Truth:	774
<b>Part 2: SSO Detection Techniques</b>				
Keywords in DOM Tree	The DOM tree contains nodes that are a programmatic representation of the elements on a website. All SSO buttons can be mapped to nodes in the DOM tree. Iterating over the DOM tree and searching for SSO-related keywords like “Sign in with Google” or more generic terms like “google” in the element’s content or attributes can reveal SSO buttons. XPath [106] and CSS selectors [15] are appropriate tools for scraping keywords.	[5, 7, 18, 29, 33, 37, 40, 41, 42, 59, 61, 69, 78, 79, 97, 110, 111]	21 (3%)	528 (68%)
⊕ Logos of IdPs in Screenshot	SSO buttons commonly embed icons with logos of the IdPs (e.g.,  ,  ,  ). Using image analysis techniques to detect IdP logos in screenshots of the login page works independent of the website’s language and source code. The coordinates of identified logos in proportion to the login page screenshot provide sufficient information to trigger SSO.	–	30 (4%)	485 (63%)
Keywords in Accessibility Tree	The accessibility tree contains nodes that are relevant for visual presentation (e.g., screen readers) [92]. HTML elements are enriched with aria attributes such that the browser gains meaningful information about the elements to enhance the website’s accessibility. If the website puts focus on its accessibility, SSO-related keywords may appear in nodes of the tree.	[28]	0 (0%)	482 (62%)
⊕ Patterns in In-Browser Communications	Certain SSO buttons from IdPs like Google and Facebook are embedded as iframes. SPs use JavaScript SDKs from IdPs that communicate with these framed SSO buttons through InBC. This communication takes place in the background, even before the SSO buttons are clicked. To detect these SSO buttons, we can listen to the InBC during page initialization.	–	106 (14%)	106 (14%)
⊕ API Calls in Browser	Chrome provides a native web API for SSO authentication, called Federated Credential Management (FedCM) API [22]. To use this API, IdPs have to implement five endpoints and SPs have to invoke the navigator.credentials.get function. By overwriting this function, we can intercept all API calls and detect if websites are using it for SSO.	–	1 (0%)	1 (0%)
			Ground Truth:	774

TABLE 5: Systematization of Login Page Detection and SSO Detection Techniques. Prior research focused on crawling websites to determine their login pages (→ part 1) and searching for keywords to detect SSO buttons (→ part 2). We systematize all techniques and measure their reliability in terms of their total number of identified SSO buttons. Overall, we advance the state-of-the-art with contributions to six techniques (⊕). *More technical details in §5.2.1 and §5.2.2.*



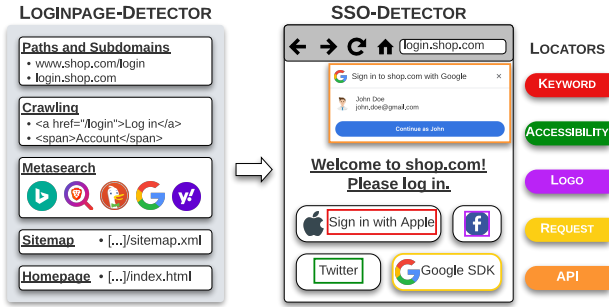


Figure 4: SSO-AGENT. We gather login pages of a domain using five techniques. We iterate over each login page and apply five locators to detect SSO buttons. This modular architecture paves the way for future extensions.

## 5.1. SSO-ORCHESTRATOR

The SSO-ORCHESTRATOR manages the archiving of the SSO landscape. Therefore, it continuously iterates over the top 1M domains ranked by the Tranco list [45]. For each domain, a new snapshot request is pushed to a queue. To periodically update the SSO archive, the iteration is repeated once all domains have been processed. SSO-ORCHESTRATOR retrieves the captured snapshots from the SSO-AGENTS, stores them in a database, and provides access to them via its web interface and API.

**Configurable Snapshot Requests.** Snapshot requests are highly configurable with 57 configuration options. These include the browser’s user agent, a list with search engines, search terms to find the login page, and prioritized detection strategies for SSO buttons.

**Artifacts.** SSO-MONITOR provides multiple artifacts:

- (1) The archive displays all snapshots for a particular domain over time.
- (2) We compute statistics on snapshots and compare snapshots with different configuration options or over a certain period of time.
- (3) We provide all snapshots in a publicly accessible API.
- (4) The downloadable Tranco+SSO list includes all snapshots in a JSON file, similar to Tranco [45] and Common Crawl [12].

## 5.2. SSO-AGENT

Multiple SSO-AGENTS are fetching snapshot requests from the queue to execute them in parallel. To take a snapshot of any Tranco domain, the SSO-AGENT executes two modules, see Figure 4: (1) The LOGINPAGE-DETECTOR navigates to the domain and aggregates a set of login pages. (2) The SSO-DETECTOR detects SSO buttons on the login pages and clicks them to capture the SSO flows.

**Browser.** We use Playwright [21] to instrument either a Chromium, Firefox, or WebKit browser. For each snapshot request, we can configure several browser settings, for instance, the viewport size, user agent, and language. We embed five extensions to dismiss cookie banners [84]. All HTTP traffic and InBC is recorded in HAR files, which is a standardized, JSON formatted archive file format [36].

**Running Example.** This section exemplifies how the SSO-AGENT takes a snapshot of the domain shop.com.

**5.2.1. LOGINPAGE-DETECTOR.** In compliance with the HTTPS upgrade policy [38], the LOGINPAGE-DETECTOR

first visits <https://shop.com>, and if it fails to load, resorts to <http://shop.com>. It follows redirects, waits until the page is loaded, and catches navigation exceptions such as DNS errors and timeouts. In the following, the domain shop.com resolves to <https://www.shop.com/index.html>. Depending on the configuration, we execute up to five techniques to aggregate a set of login pages that share the same eTLD+1 as the resolved domain (i.e., shop.com).

(1) *Paths and Subdomains:* To avoid false positives, we first probe a random path and subdomain that do not exist. If this probe returns an erroneous status code like 404, we continue testing 33 paths like /login and 5 subdomains like login.shop.com. This set replicates all login pages in our ground truth. If the probe returns a status code indicating success like 200, we cannot distinguish between existent and non-existent paths and subdomains.

(2) *Crawling:* We composed a set of 21 ranked regular expressions matching all login pages in our ground truth. On a website’s homepage, we (a) crawl all `<a>` elements and apply our regular expressions to filter and prioritize their href attributes, and (b) click on all elements like `<button>` and `<span>` that contain one of 24 login-related keywords, note the post-click URLs, and apply the regular expressions on them for filtering and prioritization.

(3) *Metasearch:* We submit the search term shop.com login to the open-source metasearch engine SearXNG [31, 73], which aggregates results from 14 search engines: Bing, Brave, DuckDuckGo, Google, Mojeek, Qwant, Startpage, Wiby, Yahoo, Seznam, Goo, Naver, Petal Search, and Yep.

(4) *Sitemap:* We use the open-source Python sitemap parser [30] to fetch all sitemaps of a website. A website’s sitemap structure can grow significantly in complexity and size, as sitemaps can reference each other. Thus, we extended the parser to skip all sitemaps beyond a depth of 1 and size of 50MB to save time and storage.

(5) *Homepage:* We add the website’s homepage to the set of login pages, i.e., <https://www.shop.com/index.html>.

**5.2.2. SSO-DETECTOR.** The SSO-DETECTOR applies up to five locators on each login page to detect SSO buttons of the top 10 IdPs, as shown in Table 3.

(1) The *DOM-Locator* traverses all clickable elements in a website’s DOM tree, such as `<button>`, `<div>`, and `<custom>`. We extract the text and attributes of elements to search for (a) 45 multilingual keywords per IdP, such as “login with google” and “continuar con facebook” (b) 15 language-independent keywords, such as “google” and “fb”. When combined, both sets of keywords replicate the HTML snippets of the SSO buttons in our ground truth.

(2) The *Accessibility-Locator* uses the Chrome DevTools Protocol (CDP) to fetch a website’s accessibility tree [10], which is searched for the same keywords.

(3) The *Logo-Locator* applies an OpenCV template matching algorithm [64] to identify IdP logos in a screenshot of the login page. The algorithms slide the logo over the screenshot to compare the logo and patch of the screenshot under the logo. Since IdP logos can have different shapes and sizes, we use multiple logos for each IdP and scale them. We consider the logo to be found if its highest pixel match exceeds a certain threshold.

(4) The *InBC-Locator* intercepts and matches InBCs (i.e., *postMessage* and *Channel Messaging*) against a set of recognition patterns to detect SSO buttons from SDKs.

(5) The *API-Locator* hooks into the `navigator.credentials` properties to intercept Credential Management (CM) and Federated Credential Management (FedCM) API calls. We detect websites requesting password, SSO, WebAuthn [102], or passkey [66] credentials.

Finally, we click on the coordinates of all located SSO buttons to verify if SSO flows are started. Clicking the coordinates instead of specific elements triggers all underlying event listeners. We detect SSO messages, such as the login request, based on recognition patterns we extracted from our ground truth. For example, Apple's SSO is running on <https://appleid.apple.com/auth/authorize>. Although the verification step is time-consuming, it eliminates false positives (identified buttons not starting SSO).

### 5.3. Limitations

**Domain Aliases.** There is no *reliable* way to check whether the owner of the Tranco domain (i.e., bit.ly) is responsible for managing the login on the resolved domain (i.e., bitly.com). Thus, we cannot ascertain the authenticity of the login pages on the resolved domains. Tracker Radar's entity list has been used to determine whether the same company owns two distinct domains [76, 95]. However, especially lower-ranked domains may not appear in the list<sup>8</sup> and different domains owned by the same company may still have independent logins<sup>9</sup>.

**Unreachable Domains.** Some Tranco domains are not reachable, primarily due to failed DNS lookups and timeouts. Although that number can be reduced by increasing the timeout, it does not scale well for 1M domains. Prior work [76] matched domains to URLs listed in the Chrome User Experience Report [14], which contains frequently visited URLs. However, it misses lower-ranked domains.

**Anti-Bot Measures and User Interaction.** We missed websites that served CAPTCHAs or other anti-bot measures (e.g., ebay.com). Additionally, some cookie banners were not removed by the browser extensions and blocked all user input. Finally, websites may require explicit user interaction. For example, mail.ru requires users to submit their Gmail address to trigger SSO with Google.

**Browser Window.** The width and height of the browser window are important factors. Larger screenshots require more time to be matched against the logos. Smaller viewports might miss logos located outside of it. With a viewport size of 1920x1080 pixels, we only missed a single site in our ground truth that embeds SSO buttons outside the browser's viewport, see Table 4.

**Browser Extensions.** Since Playwright only supports browser extensions in Chromium, we cannot automatically handle cookie banners in Firefox and WebKit.

**Nested IFrames and Shadow Roots.** Playwright cannot detect SSO buttons if they are included in closed-mode shadow roots [77] or nested iframes because the Locator API [50] does not traverse them. Although we enter each iframe on the main page, we do not scan nested iframes with an infinite depth to avoid complexity. Instead, the Logo-Locator detects SSO independently of the DOM.

8. I.e., the list misses [deesidepiper.co.uk](https://deesidepiper.co.uk) redirecting to [scotsman.com](https://scotsman.com).

9. I.e., Verizon Media owns [techcrunch.com](https://techcrunch.com) and [yahoo.com](https://yahoo.com).

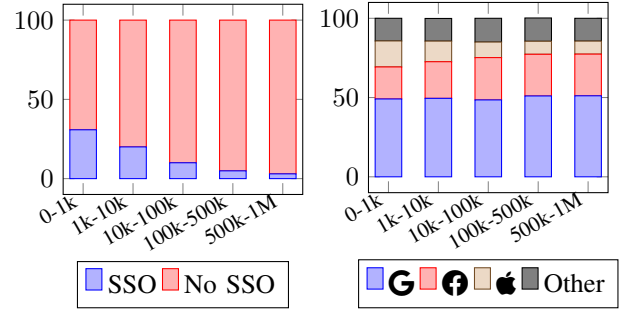


Figure 5: SSO and IdPs. The SSO support decreases on lower-ranked sites. Contrary to prior work, Google is the most prevalent IdP, followed by Facebook and Apple.

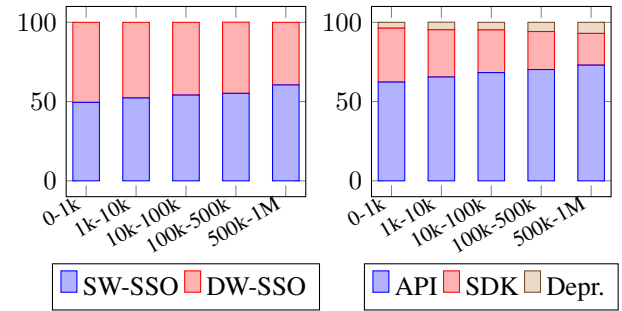


Figure 6: Flows and Integrations. DW-SSO with popups and iframes correlates with SDK integrations. DW-SSO and SDKs decrease on lower-ranked sites. Deprecated SDKs are more often integrated on less prominent sites.

## 6. Reproducible and Reliable SSO Measurements on the Tranco Top 1M Websites

SSO-MONITOR archived over 5 TB of SSO snapshots obtained from the Tranco<sup>10</sup> [45] top 1M websites in July 2023. In this section, we demonstrate that our SSO archive is valuable and paves the way for reproducible SSO landscape (§6.1) and security (§6.2) measurements.

### 6.1. SSO Landscape

In total, SSO-MONITOR identified 88,994 SSO buttons on 47,730 login pages, see Table 6. We mapped the login pages to the Tranco domains and found that 5% of the top 1M websites implement SSO (45,532).

**Methodology.** We execute *all* five techniques from Table 5 to collect login pages, add them to a list, and remove duplicates. We analyze a maximum of eight login pages for each domain. For performance reasons, we *sequentially* execute four techniques from Table 5 to detect SSO buttons, following the order in Table 6. For instance, we do not use the Logo-Locator to search for an SSO button that was already found by the DOM-Locator. Thus, the hits for some techniques are lower. We excluded the accessibility tree as it did not contribute any hits that only this technique can achieve. Note that we differentiate between login pages with SSO support and SSO buttons, as a single login page can integrate multiple SSO buttons.

10. Available at <https://tranco-list.eu/list/6Z2X>.

Login Page Detection		SSO Detection	
Search Engines	18,423	Keywords in DOM Tree	71,676
Crawling	17,803	Logos of IdPs in Screenshot	5,222
Paths and Subdomains	18,426	Patterns in InBCs	12,085
Homepage	3,949	API Calls in Browser	11
Sitemap	1,212	Keywords in Accessibility Tree	–
All Login Pages	47,730	All SSO Buttons	88,994

TABLE 6: The Top 1M SSO Landscape. The combination of multiple login page and SSO detection techniques expands the overall coverage. *More details in Table 9.*

**Domain Ranking Correlations.** Figure 5 illustrates that the SSO support decreases with the domain ranking: 30% of the top 1k but only 3% of the lower 500k websites support SSO. Google dominates, followed by Facebook and Apple. Interestingly, Facebook increases and Apple decreases on the lower-ranked websites. In Figure 6, we confirm the valuable contribution of Jannett et al. [39]: DW-SSO flows exchanging tokens via InBCs are relevant for the SSO landscape. The authors found that DW-SSO exceeds SW-SSO for the top 1k sites and three IdPs. Our results suggest that SW-SSO and DW-SSO are balanced for the top 1k sites and 10 IdPs. SW-SSO still exceeds DW-SSO on the lower-ranked domains. Notably, the usage of SDKs correlates with DW-SSO and decreases for less prominent websites. We further found that deprecated SDKs are more commonly deployed on the lower ranks.

#### Best Practices for Large-Scale SSO Measurements.

Table 6 summarizes the login pages and SSO buttons for each technique. Table 9 provides a more fine-grained view on different IdPs, integrations (API vs. SDK), and flows (SW-SSO vs. DW-SSO). With this data, we can verify or falsify the observations in our ground truth (cf. §4.2).

Regarding the login page detection, we observe that searching for paths and subdomains is the most promising approach, closely followed by search engines. This observation disagrees with Jonker et al. [41], who found crawling to be most successful. This mismatch can have three reasons: First, the authors used a fixed execution order starting with crawling and stopping when SSO was found (we scan all login pages). Second, the authors used three search engines (we use 14). Third, the authors did not test common subdomains like login.shop.com. Thus, our improvements to both techniques lead to more reliable and robust results. On the other side, we show that homepages and sitemaps only make a minimal contribution.

Regarding the SSO detection, we learn that using keywords to detect SSO is not sufficient. We have to apply a visual-based logo detection and match patterns on InBCs. Otherwise, we miss 17,307 SSO buttons (19%). Today, the FedCM API for SSO authentication is only sparsely deployed (0.01%). However, we expect it to expand in 2024 due to Chrome’s third-party cookie phaseout [32].

## 6.2. SSO Security

We analyzed the security of 88,983 login requests of 45,532 SPs that SSO-MONITOR executed and recorded in July 2023 on the Tranco top 1M websites. We found that 10,527 SPs follow the current security recommendations.

**Scope.** Over the years, numerous efforts have been made to find attacks in OAuth and OIDC. These efforts finally led to the still evolving Security Best Current

§	Attack	Detection		Mitigation		Check
4.1	Insufficient Redirect Validation	👤	📄	IdP		
4.2	Leakage via Referer Headers	👤	📄	IdP	SP	Security Header, Insecure Flows
4.3	Leakage via Browser History	👤	📄	IdP	SP	PKCE, nonce, Insecure Flows ⇒ 71 leaked secrets ⇒ 4,111 insecure flows
4.4	Mix-Up Attacks	👤	📄		SP	Issuer Binding ⇒ 10,707 missing protection
4.5	Authorization Code Injection	👤	📄		SP	PKCE, nonce ⇒ 49,009 missing protection
4.6	AT Injection	👤	📄	IdP	SP	at_hash, Insecure Flows
4.7	Cross Site Request Forgery	👤	📄		SP	PKCE, state, nonce ⇒ 34,333 missing protection
4.8	PKCE Downgrade Attack	👤	📄	IdP	SP	
4.9	AT Leakage at the RS	👤	📄	IdP	SP	
4.10	Misuse of ATs	👤	📄	IdP	SP	
4.11	Open Redirection	👤	📄	IdP	SP	
4.12	307 Redirect	👤	📄	IdP		HTTP Redirect
4.13	Reverse Proxies	👤	📄	IdP	SP	
4.14	Refresh Token Protection	👤	📄	IdP		
4.15	Client Impersonating	👤	📄	IdP		
4.16	Clickjacking	👤	📄	IdP		Security Header
4.17	AS Redirecting to Phisher	👤	📄	IdP		
4.18	In-Browser Attacks	👤	📄	IdP	SP	Wildcard Target

Active (👤) Passive (👤) Frontend (📄) Backend (📄)

Authentication needed (📄) Authentication not needed (📄)

TABLE 7: OAuth Security Best Current Practices [51]. We can use the snapshots in our SSO archive to passively measure attacks that do not require authentication and take place in the frontend (👤 & 📄 & 📄 ≐ 4 attacks).

Practices (SBCPs) [51]. Table 7 lists all attacks from the SBCPs. Failure to meet these recommendations poses the risk of the SP being susceptible to exploitation. The column “Detection” classifies the detection modalities of the attack. We retrieve the login requests from our SSO archive, thus we focus on vulnerabilities that can be detected passively (👤) through frontend communication (📄) and do not require authentication (📄). We include four vulnerabilities into our scope and shaded them gray in Table 7. In §7, we further show that SSO-MONITOR can be easily extended to run authenticated security evaluations.

**Methodology.** We evaluate security-related parameters in three steps: First, we filter the recorded HTTP traffic and InBC for the relevant login requests. Second, we extract the values like query parameters and headers. Third, we check whether countermeasures against known threats are deployed. For example, we check if a certain security parameter is used and if its value mitigates the attack. Table 7 provides an overview of all required checks.

**State and Nonce.** The OAuth parameter `state` as well as the OIDC parameter `nonce` are security-relevant parameters in the login request. If implemented correctly, they provide CSRF and replay protection. Out of 88,983, only 61% login requests use a state and 5% use a nonce. To our surprise, we found 13 login requests having multiple `state` parameters in the same request. In total, 2,578 states and 199 nonces were not randomly chosen and thus cannot provide CSRF protection.

- The three most common `state` values are: `state` (149), 1 (84), `STATE` (76).
- The three most common `nonce` values are: `nonce` (114), `[NONCE]` (17), 0 (9).

Altogether, we found 6,386 states and 718 nonces with insufficient entropy for CSRF protection (lower than 128 bit [65]). Moreover, 3,637 hold URLs, which could potentially be abused for open redirects [9]. We did not exploit these vulnerabilities, as they require active testing for verification, which is beyond the scope of this paper. Interestingly, 12,360 states and 130 nonces include further structures, such as arrays, JWTs, and Base64.

**PKCE.** The Proof Key for Code Exchange (PKCE) mechanism was initially designed to protect mobile

SPs [72]. It has also been proven to prevent injections and CSRF attacks. We found that PKCE is only used in 1,445 authentication flows. 81 out of these flows use PKCE as a replacement to `state` and `nonce` parameters. When considering the use of PKCE, `state`, or `nonce` to protect against CSRF, then 34,333 flows lack all these parameters. Interestingly, Facebook enabled PKCE in mid 2022 as an experimental feature [63]. One year later, we measured that 201 of 23,265 Facebook SSO flows use PKCE. We hope to measure the adoption of Facebook's PKCE protection in the long term.

**Insecure Flows.** We identified login requests violating the SSO protocol as described in the following:

- (1) *OAuth vs. OIDC*: The SP requests an OIDC token while starting OAuth in 4,760 authentication flows. This is not a security issue per se, but could escalate further to more serious threats, since OAuth is designed for authorization and not authentication.
- (2) *Secret Leakage*: Shared secrets between SPs and IdPs were leaked in 71 login request. These secrets must only be sent from the SPs to the IdPs in their backend communication. If the shared secret is compromised, an attacker can impersonate the affected SP.
- (3) *Forbidden Flows*: During the first years of OAuth, it was common practice to transport an `access_token` through the user's browser. This is nowadays deprecated [51] because such tokens can leak to attackers, i.e., in `Referer` headers. However, we identified 4,111 SSO buttons still having this security issue.

**Mix-up Mitigation.** In 2016, Fett et al. [24] discovered a new vulnerability called mix-up attack, which affects only SPs supporting multiple IdPs. In OAuth and OIDC, this attack is prevented with the `issuer` parameter [75, 93]. However, none of the investigated IdPs provides the `issuer` parameter according to [75]. Alternatively, the login request must contain a distinct `redirect_uri` parameter for each IdP. In total, we found 25,671 SPs supporting *multiple* IdPs that must prevent mix-up attacks. In 14,964 SPs, we found the following countermeasures: (1) 7,370 SPs are protected through the `issuer` parameter as they only use OIDC. (2) 6,825 use individual `redirect_uri` parameters for each IdP. (3) 769 use OIDC and individual `redirect_uri` parameters for each remaining IdP. Of the remaining 10,707 SPs, (1) 1,715 use OIDC, but fail to use distinct `redirect_uri` parameters and (2) 8,992 use neither OIDC, nor distinct `redirect_uri` parameters.

## 7. SSO Measurements: Future Research

Practical-oriented security research in the field of SSO often consists of two parts. First, researchers must create a list of websites supporting SSO. Using SSO-MONITOR, this step is already solved for future research. Second, researchers conduct specific experiments for their studies. One common way is to use a browser extension [18, 28, 29, 39] and automate visiting SSO-enabled websites. This section shows how to use SSO-MONITOR as a framework for generic SSO experiments and comprehensive analyses on a large scale with minimal effort.

### 7.1. Scaling Token Leaks from Tranco 1k to 1M

As a proof of demonstration, we show how a prior study could have been done by using SSO-MONITOR. First, we selected a recent study [39] that conducted a manual security evaluation of the Tranco top 1k websites. The authors implemented a browser extension called DISTINCT to measure the security of DW-SSO. This variant uses InBC like `postMessage` to transfer authentication tokens between SPs and IdPs. The authors *manually* discovered and clicked the SSO buttons, authenticated on the IdPs, and granted consent. We demonstrate how to extend SSO-MONITOR to perform automatic investigations on all SSO websites listed in the Tranco top 1M. Note that DISTINCT was not designed for automated evaluations, but we have expanded it with minimal effort.

**Authentication and Consent on IdP.** For executing the full SSO login flow, we automated the authentication and consent for Google, which is the most frequently supported IdP (cf. Table 3). Other IdPs could have been integrated similarly, although technical constraints exist.<sup>11</sup>

**Token Leaks in the Wild.** The Wildcard Receiver Attack (WRA) is a vulnerability that can be detected by passive scanning with the DISTINCT browser extension. For ethical reasons, we did not alter the HTTP parameters. In the WRA, the login response or other authentication-relevant messages are sent to the wildcard origin \*. The wildcard origin allows any website to receive the message, including malicious ones. We measured this implementation flaw on all websites in our SSO archive. SSO-MONITOR automatically executed the SSO logins and DISTINCT identified the WRA.

**Results.** In 2022, Jannett et al. [39] measured WRAs on 11 websites in the Tranco 1k. One year later, we found WRAs on 339 websites in the Tranco 1M, whereby only two of them were in the top 1k. By investigating the leaks, we can confirm these patterns: (1) SSO authentication tokens leaked on 28 sites, enabling the attacker to take over the victim's account. (2) Additionally, leaked information included emails (93), names (71), and profile pictures (6).

However, we identified *new leak types* resulting from the WRA that have not been discovered in the Tranco 1k: (1) The most severe leak concerned the victim's credentials. Their usernames and passwords leaked on 30 sites. Normally, there should not be any credentials if SSO is used. However, the SP created some random credentials for the account. This leak enables the attacker to log into the victim's account. (2) The `state` parameter leaked on 58 sites, enabling CSRF attacks. (3) We found evidence of additional authentication tokens and personal data leaks. For example, we noticed that 8 leaked JWTs that are commonly used by websites as authentication tokens. An Indian credit company leaked a particularly large amount of personal data, including the victim's loan amount, monthly salary, bank account, and work experience. Although IdP-issued tokens follow similar patterns<sup>12</sup>, the automatic detection of custom tokens or personal data in partially encoded messages is a difficult problem that we leave open for future work.

**Vulnerable SSO as a Service (SSOaaS).** We observed similarities between leaked messages on different

11. Facebook blocks IPs that are issuing many SSO login attempts.

12. I.e., Google's `access_token` always starts with `ya29`.



websites. Further analysis revealed that these websites integrated libraries from SSOaaS providers. The WRA affected all SPs integrating vulnerable libraries. For example, we found 31 account takeovers in webshops caused by two insecure Shopify [85] SSO plugins. Two vulnerable providers induced WRAs on 136 websites.

## 7.2. Flexible Extensibility

**Geographic Location.** Future work should measure SSO on websites from different vantage points, for example, located in the United States, Europe, Russia, and China. For example, the US version of *usatoday.com* supports SSO with Facebook, Google, Apple, and Twitter, but the EU version warns that “this feature isn’t currently supported”. In preliminary studies, Saito et al. [71] already showed that SSO in the US is twice as widespread as in Japan by comparing the use of Google’s, Facebook’s, Twitter’s, and Yahoo’s SSO on the top 500 websites. Morkonda et al. [59] investigated the user data requested by Google, Facebook, Apple, and LinkedIn on the top 500 sites in five countries (Australia, Canada, Germany, India, United States). Roth et al. [70] showed that there are multiple inconsistencies in the security of web applications based, among others, on the IP address. Senol et al. [76] measured that email leaks on websites are more common in the US compared to the EU. SSO-MONITOR could be extended with BrowserStack’s IP geolocation feature [83] that simulates different locations hosted in 45+ countries.

**Deep Parameter Inspection.** Prior work investigated the use of SSO security parameters, but did not inspect their content. For instance, it is not trivial to determine whether a parameter’s content contains sufficient randomness. As shown by Drakonakis et al. [18], key insights could be deeply encoded in parameters (i.e., URL, HTML Form, Base64, Deflate), included in nested structures (i.e., XML, JSON, JWT), or contain seemingly random but predictable values (i.e., UNIX timestamps). Future work should build up on our archived data and implement a deep parameter inspection framework that can reliably detect encodings, nested structures, and structured data like timestamps to automatically unfold parameters.

**Cross-Message Parameter Tracing.** Our SSO archive supports future work in tracing SSO parameters between messages contained in HTTP traffic and INBCs. For example, query, fragment, or body parameters from the login request may be included in HTTP headers of other messages or stored in browser storage containers like *localStorage*. Deriving these parameter relationships and dependencies across multiple prior or subsequent messages can uncover a wide range of new security and privacy issues. For example, this approach can detect SSO parameters leaked in HTTP headers to third parties.

**Active Parameter Manipulations.** Upon reporting our identified security gaps, the developers implemented the necessary security parameters. However, our evaluation of these fixes revealed that while the appropriate parameters were added, there was a lack of sufficient validation. This motivates incorporating active parameter manipulations into SSO-MONITOR. Addressing two critical aspects of active parameter manipulations is vital as we move forward. First, we must carefully evaluate the ethical implications of manipulating security parameters.

Second, we must actively modify them to ascertain the robustness of the implemented validation mechanisms.

**Studying the Authenticated Web.** We see our SSO archive as a baseline for measurements on the authenticated web. As a result, post-login security measurements such as security attributes of session cookies and secure storage of tokens in the browser can be provided. We also pave the way for active measurements on the authenticated web with the extensible architecture of SSO-MONITOR.

## 8. Discussion

### 8.1. Ethical Considerations

**Data Collection.** We took adequate measures and scanning best practices [19, §5] to avoid overloading websites and networking infrastructure. For instance, we restricted our number of clicks and used sleeps between them to reduce our request rate. We adhered to the instructions in the website’s *robots.txt* [43]. We configured our scanning server with reverse DNS and deployed a disclaimer page that informs about our research, presents our institution’s imprint, and explains how websites can opt out of our scanning scope. We did not use CAPTCHA solving software [6] or services to bypass anti-bot measures [8]. For security tests, we used our own testing accounts and did not interfere with any user accounts or data other than ours. We only performed passive analyzes and resisted actively modifying the traffic.

**Responsible Disclosure.** We used well-established security reporting mechanisms from prior work [18, 46, 86, 87] to collect the contact emails. For instance, we considered the *security.txt* file<sup>13</sup> [74], the WHOIS record, off-the-shelf search engine [57] and website [2] email crawlers, and the standard aliases *security@*, *abuse@*, *webmaster@*, and *info@*. We sent the emails from our institutional email address to verify our identity and maximize credibility. While we participate in active discussions with the vendors, some of them have already resolved the issues. We appreciate being acknowledged and rewarded with bug bounties.

### 8.2. Improving the SSO Security for the Users

**Security by Default.** This paper reveals for one more time that existing SBPs are often ignored and poorly implemented by the Tranco top 1M websites. The question of how this situation could be improved is raised. For example, the deployment of TLS on websites has greatly improved, since browsers penalize websites that do not support TLS. Similarly, this would be possible for IdPs. For example, they could drop login requests without following best practices. Future research should focus on how secure default configurations can be deployed.

**Browser Authentication APIs.** Browsers already provide web APIs for password authentication. For example, the CM API [13] allows websites to store and retrieve user passwords. Web APIs for SSO authentication could improve the security and privacy of SSO in the long term. Thereby, the implementation of SSO is outsourced

13. Only 11,183 of 1M domains provide a *security.txt* file.

from website developers to browser vendors, making it less prone to implementation flaws. In 2022, Google proposed the FedCM API [22] for privacy-preserving identity federation. As of today, the API is only implemented in Chrome. However, Mozilla is implementing a prototype in Firefox and Apple has expressed interest [22]. We measured the use of FedCM on 1M websites and found a currently low adoption rate of only two IdPs and 11 SPs.

**Developers.** The research community should investigate the reasons why developers do not follow SBCPs. For example, Geierhaas et al. [27] studied the security of password programming tasks solved by 179 freelancers. They implemented a programming aid called “Let’s Hash” that provides assistance to developers in the form of code snippets and examples. We recommend conducting similar developer studies with SSO implementation tasks and extending “Let’s Hash” with secure SSO guidelines.

## 9. Conclusion

This paper sheds light on the challenges of conducting reproducible and reliable SSO landscape and security measurements. Through a systematic analysis of the current state of SSO measurement tools, we identify gaps in both aspects that hinder future large-scale SSO research. Therefore, we propose SSO-MONITOR, our public SSO archive storing snapshots of SSO implementations over time to provide a valuable resource for future security research. To the best of our knowledge, we are the first to provide a research-oriented SSO archive for the Internet. Researchers can easily run their instance or access our continuously updated data corpus on sso-monitor.me.

Following Tranco [45] and the Internet Archive [101], we strongly believe that an expanding archive of the SSO landscape and its evolution on security is a valuable contribution for academia, industry, and standardization bodies. Hence, we encourage utilizing our SSO archive for reproducible and reliable research, even beyond the scope of SSO. Moving forward, we will continue to maintain and improve SSO-MONITOR for our and other research. The suggested future research directions serve as preliminary starting points. We plan to expand our archive to include a comprehensive collection of web authentication mechanisms beyond SSO, including passwords and passkeys.

## Acknowledgments

We thank the reviewers for their valuable feedback. This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC 2092 CASA – 390781972. 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] *Android-Apps auf Google Play*. <https://play.google.com/store/games>. (Accessed on 10/29/2023).
- [2] arifszn. *email-scraper*. [Online; accessed 8. Oct. 2022]. Oct. 2022. URL: <https://github.com/arifszn/email-scraper> (visited on 10/08/2022).
- [3] Association for Computing Machinery. *Artifact Review and Badging - Current*. <https://www.acm.org/publications/policies/artifact-review-and-badging-current>. (Accessed on 10/29/2023).
- [4] Guangdong Bai, Jike Lei, Guozhu Meng, Sai Sathyanarayan Venkatraman, Prateek Saxena, Jun Sun, Yang Liu, and Jin Song Dong. “AUTH-SCAN: Automatic extraction of web authentication protocols from implementations”. In: *20th Network and Distributed System Security Symposium (NDSS)* (2013).
- [5] Michele Benolli, Seyed Ali Mirheidari, Elham Arshad, and Bruno Crispo. “The Full Gamut of an Attack: An Empirical Analysis of OAuth CSRF in the Wild”. In: *Detection of Intrusions and Malware, and Vulnerability Assessment*. Ed. by Leyla Bilge, Lorenzo Cavallaro, Giancarlo Pellegrino, and Nuno Neves. Vol. 12756. Cham: Springer International Publishing, 2021, pp. 21–41. DOI: 10.1007/978-3-030-80825-9\_2. URL: [https://link.springer.com/10.1007/978-3-030-80825-9\\_2](https://link.springer.com/10.1007/978-3-030-80825-9_2) (visited on 10/21/2023).
- [6] *Buster: Captcha Solver for Humans - Chrome Web Store*. <https://chrome.google.com/webstore/detail/buster-captcha-solver-for-mpbjkejclgfgadiemmefgebjfooffhl>. (Accessed on 04/21/2023).
- [7] 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)*. Baltimore, MD: USENIX Association, Aug. 2018, pp. 1493–1510. ISBN: 978-1-939133-04-5. URL: <https://www.usenix.org/conference/usenixsecurity18/presentation/calzavara>.
- [8] *CaptchaAI - The 1st “reCAPTCHA”, “hCaptcha” OCR Solver*. <https://captchaai.com/>. (Accessed on 04/21/2023).
- [9] Vikrant Singh Chauhan. *Stealing OAuth Tokens of Connected Microsoft Accounts via Open Redirect in Harvest App*. Oct. 21, 2023. URL: <https://eval.blog/research/microsoft-account-token-leaks-in-harvest> (visited on 11/03/2023).
- [10] *Chrome DevTools Protocol - Accessibility domain*. <https://chromedevtools.github.io/devtools-protocol/tot/Accessibility/>. (Accessed on 06/30/2023).
- [11] *Cisco Popularity List*. <https://umbrella-static.s3-us-west-1.amazonaws.com/index.html>. (Accessed on 10/29/2023).
- [12] *Common Crawl*. <https://commoncrawl.org/>. (Accessed on 06/19/2023).
- [13] *Credential Management API - Web APIs — MDN*. [https://developer.mozilla.org/en-US/docs/Web/API/Credential\\_Management\\_API](https://developer.mozilla.org/en-US/docs/Web/API/Credential_Management_API). (Accessed on 04/26/2023).
- [14] *CrUX on BigQuery - Chrome Developers*. <https://developer.chrome.com/docs/crux/bigquery/>. (Accessed on 04/21/2023).

- [15] *CSS selectors - CSS: Cascading Style Sheets* — MDN. [https://developer.mozilla.org/en-US/docs/Web/CSS/CSS\\_selectors](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_selectors). (Accessed on 10/30/2023).
- [16] Nurullah Demir, Matteo Große-Kampmann, Tobias Urban, Christian Wressnegger, Thorsten Holz, and Norbert Pohlmann. “Reproducibility and Replicability of Web Measurement Studies”. In: *Proceedings of the ACM Web Conference 2022*. WWW ’22. New York, NY, USA: Association for Computing Machinery, Apr. 25, 2022, pp. 533–544. ISBN: 978-1-4503-9096-5. DOI: 10.1145/3485447.3512214. URL: <https://dl.acm.org/doi/10.1145/3485447.3512214> (visited on 10/25/2023).
- [17] Nurullah Demir, Jan Hörnemann, Matteo Große-Kampmann, Tobias Urban, Norbert Pohlmann, Thorsten Holz, and Christian Wressnegger. “On the Similarity of Web Measurements Under Different Experimental Setups”. In: *Proceedings of the 2023 ACM on Internet Measurement Conference*. IMC ’23. New York, NY, USA: Association for Computing Machinery, Oct. 24, 2023, pp. 356–369. DOI: 10.1145/3618257.3624795. URL: <https://dl.acm.org/doi/10.1145/3618257.3624795> (visited on 10/25/2023).
- [18] Kostas Drakonakis, Sotiris Ioannidis, and Jason Polakis. “The cookie hunter: Automated black-box auditing for web authentication and authorization flaws”. In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 2020, pp. 1953–1970.
- [19] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. “ZMap: Fast Internet-wide Scanning and Its Security Applications”. In: *22nd USENIX Security Symposium (USENIX Security 13)*. Washington, D.C.: USENIX Association, Aug. 2013, pp. 605–620. ISBN: 978-1-931971-03-4. URL: <https://www.usenix.org/conference/usenixsecurity13/technical-sessions/paper/durumeric>.
- [20] *Elon Musk Completes \$44 Billion Deal to Own Twitter - The New York Times*. <https://www.nytimes.com/2022/10/27/technology/elon-musk-twitter-deal-complete.html>. (Accessed on 04/03/2024).
- [21] *Fast and reliable end-to-end testing for modern web apps — Playwright Python*. <https://playwright.dev/python/>. (Accessed on 06/19/2023).
- [22] *Federated Credential Management API - Chrome Developers*. <https://developer.chrome.com/docs/privacy-sandbox/fedcm/>. (Accessed on 04/26/2023).
- [23] Daniel Fett, Pedram Hosseini, and Ralf Küsters. *An Extensive Formal Security Analysis of the OpenID Financial-grade API*. Jan. 31, 2019. arXiv: 1901.11520. URL: <http://arxiv.org/abs/1901.11520> (visited on 12/09/2020). preprint.
- [24] Daniel Fett, Ralf Küsters, and Guido Schmitz. “A Comprehensive Formal Security Analysis of OAuth 2.0”. In: *23rd ACM SIGSAC Conference on Computer and Communications Security (CCS)* (2016), pp. 1204–1215.
- [25] Daniel Fett, Ralf Küsters, and Guido Schmitz. “Paper: An Expressive Model for the Web Infrastructure: Definition and Application to the BrowserID SSO System”. In: *35th IEEE Symposium on Security and Privacy (S&P)*. IEEE Computer Society, 2014.
- [26] Daniel Fett, Ralf Küsters, and Guido Schmitz. “The web SSO standard openid connect: In-depth formal security analysis and security guidelines”. In: *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE. 2017, pp. 189–202.
- [27] Lisa Geierhaas, Anna-Marie Orloff, Matthew Smith, and Alena Naiakshina. “Let’s Hash: Helping Developers with Password Security”. In: *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*. Boston, MA: USENIX Association, Aug. 2022, pp. 503–522. ISBN: 978-1-939133-30-4. URL: <https://www.usenix.org/conference/soups2022/presentation/geierhaas>.
- [28] M. Ghasemisharif, C. Kanich, and J. Polakis. “Towards Automated Auditing for Account and Session Management Flaws in Single Sign-On Deployments”. In: *2022 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, May 2022, pp. 1524–1524. DOI: 10.1109/SP46214.2022.9833753. URL: <https://doi.org/10.1109/SP46214.2022.9833753>.
- [29] Mohammad Ghasemisharif, Amruta Ramesh, Stephen Checkoway, Chris Kanich, and Jason Polakis. “O Single Sign-Off, Where Art Thou? An Empirical Analysis of Single Sign-On Account Hijacking and Session Management on the Web”. en. In: *USENIX*. USENIX Association, 2018, p. 19.
- [30] *GitHub - mediacloud/ultimate-sitemap-parser: Ultimate Website Sitemap Parser*. <https://github.com/mediacloud/ultimate-sitemap-parser>. (Accessed on 06/29/2023).
- [31] *GitHub - searxng/searxng: SearXNG is a free internet metasearch engine which aggregates results from various search services and databases. Users are neither tracked nor profiled*. <https://github.com/searxng/searxng>. (Accessed on 06/29/2023).
- [32] Google. *Migrate to FedCM — Authentication — Google for Developers*. <https://developers.google.com/identity/gsi/web/guides/fedcm-migration>. (Accessed on 11/01/2023). Sept. 2023.
- [33] Andreas Grüner, Alexander Mühle, Nils Rümmler, Adnan Kadric, and Christoph Meinel. “Quo Vadis, Web Authentication? — An Empirical Analysis of Login Methods on the Internet”. In: *Advanced Information Networking and Applications*. Ed. by Leonard Barolli. Vol. 655. Cham: Springer International Publishing, 2023, pp. 471–479. DOI: 10.1007/978-3-031-28694-0\_45. URL: [https://link.springer.com/10.1007/978-3-031-28694-0\\_45](https://link.springer.com/10.1007/978-3-031-28694-0_45) (visited on 05/23/2023).
- [34] Florian Hantke, Stefano Calzavara, Moritz Wilhelm, Alvise Rabitti, and Ben Stock. “You Call This Archaeology? Evaluating Web Archives for Reproducible Web Security Measurements”. In: *ACM CCS 2023*. ACM CCS. Nov. 2023. URL: <https://publications.cispa.saarland/3953/>.

- [35] D. Hardt. *The OAuth 2.0 Authorization Framework*. RFC 6749. IETF, Oct. 2012. URL: <http://tools.ietf.org/rfc/rfc6749.txt>.
- [36] *HTTP Archive (HAR) format*. <https://w3c.github.io/web-performance/specs/HAR/Overview.html>. (Accessed on 06/19/2023).
- [37] Tommaso Innocenti, Matteo Golinelli, Kaan Onarlioglu, Bruno Crispo, and Engin Kirda. “OAuth 2.0 Redirect URI Validation Falls Short, Literally”. In: *Annual Computer Security Applications Conference (ACSAC)*. Dec. 2023. DOI: 10.1145/3627106.3627140.
- [38] *Intent to Prototype: HTTPS Upgrades*. <https://groups.google.com/a/chromium.org/g/blink-dev/c/mgJqym5-Xek/m/0EAN6v7CCQAJ>. (Accessed on 06/28/2023).
- [39] 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*. CCS ’22: 2022 ACM SIGSAC Conference on Computer and Communications Security. Los Angeles, CA, USA: ACM, Nov. 2022. ISBN: 978-1-4503-9450-5. DOI: 10.1145/3548606.3560692.
- [40] 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)*. ISSN: 1861-2288. June 2022, pp. 1–9. DOI: 10.23919/IFIPNetworking55013.2022.9829804.
- [41] Hugo Jonker, Jelmer Kalkman, Benjamin Krumnow, Marc Slegers, and Alan Verresen. “Shepherd: Enabling Automatic and Large-Scale Login Security Studies”. In: *CoRR* abs/1808.00840 (2018). arXiv: 1808.00840. URL: <http://arxiv.org/abs/1808.00840>.
- [42] Soheil Khodayari and Giancarlo Pellegrino. “The State of the SameSite: Studying the Usage, Effectiveness, and Adequacy of SameSite Cookies”. In: *43rd IEEE Symposium on Security and Privacy (S&P ’22)*. May 2022. URL: <https://publications.cispa.saarland/3504/>.
- [43] Martijn Koster, Gary Illyes, Henner Zeller, and Lizzi Sassman. *Robots Exclusion Protocol*. RFC 9309. Sept. 2022. DOI: 10.17487/RFC9309. URL: <https://www.rfc-editor.org/info/rfc9309>.
- [44] *Laden Sie APK auf Android mit kostenloser Online -APK -Downloader - APKPure herunter*. <https://m.apkpure.com/de/>. (Accessed on 10/29/2023).
- [45] 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. Feb. 2019. DOI: 10.14722/ndss.2019.23386. URL: <https://tranco-list.eu/list/6Z2X>.
- [46] Frank Li, Zakir Durumeric, Jakub Czyz, Mohammad Karami, Michael Bailey, Damon McCoy, Stefan Savage, and Vern Paxson. “You’ve Got Vulnerability: Exploring Effective Vulnerability Notifications”. In: *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, Aug. 2016, pp. 1033–1050. ISBN: 978-1-931971-32-4. URL: <https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/li>.
- [47] 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*. 2019, pp. 35–44.
- [48] Wanpeng Li and Chris J. Mitchell. “Analysing the Security of Google’s Implementation of OpenID Connect”. In: (2016), pp. 357–376. DOI: 10.1007/978-3-319-40667-1\_18. URL: [http://dx.doi.org/10.1007/978-3-319-40667-1\\_18](http://dx.doi.org/10.1007/978-3-319-40667-1_18).
- [49] Guannan Liu, Xing Gao, and Haining Wang. “An Investigation of Identity-Account Inconsistency in Single Sign-On”. In: *Proceedings of the Web Conference 2021*. New York, NY, USA: ACM, Apr. 2021, pp. 105–117. ISBN: 9781450383127. DOI: 10.1145/3442381.3450085. URL: <https://dl.acm.org/doi/10.1145/3442381.3450085>.
- [50] *Locator — Playwright Python*. <https://playwright.dev/python/docs/api/class-locator>. (Accessed on 04/21/2023).
- [51] Torsten Lodderstedt, John Bradley, Andrey Labunets, and Daniel Fett. *OAuth 2.0 Security Best Current Practice*. Internet-Draft draft-ietf-oauth-security-topics-15. <http://www.ietf.org/internet-drafts/draft-ietf-oauth-security-topics-15.txt>. IETF Secretariat, Apr. 2020. URL: <http://www.ietf.org/internet-drafts/draft-ietf-oauth-security-topics-15.txt>.
- [52] Christian Mainka, Vladislav Mladenov, Florian Feldmann, Julian Krautwald, and Jörg Schwenk. “Your Software at My Service: Security Analysis of SaaS Single Sign-On Solutions in the Cloud”. In: *6th Edition of the ACM Workshop on Cloud Computing Security (CCSW)*. CCSW ’14. Scottsdale, Arizona, USA: ACM, 2014, pp. 93–104. ISBN: 978-1-4503-3239-2. DOI: 10.1145/2664168.2664172. URL: <http://doi.acm.org/10.1145/2664168.2664172>.
- [53] Christian Mainka, Vladislav Mladenov, Tim Günther, and Jörg Schwenk. “Automatic Recognition, Processing and Attacking of Single Sign-On Protocols with Burp Suite”. In: *Open Identity Summit*. 2015.
- [54] Christian Mainka, Vladislav Mladenov, and Jörg Schwenk. “Do not trust me: Using malicious IdPs for analyzing and attacking Single Sign-On”. In: *IEEE 1st European Symposium on Security and Privacy (Euro S&P)*. IEEE, Mar. 2016.
- [55] Christian Mainka, Vladislav Mladenov, Tobias Wich, and Jörg Schwenk. “SoK: Single Sign-On Security – An Evaluation of OpenID Connect”. In: *IEEE 2nd European Symposium on Security and Privacy (Euro S&P)*. Apr. 2017.



- [56] *Majestic Million*. <https://majestic.com/reports/majestic-million>. (Accessed on 10/29/2023).
- [57] maldevel. *maldevel/EmailHarvester: Email addresses harvester*. <https://github.com/maldevel/EmailHarvester>. (Accessed on 10/08/2022). 2022.
- [58] Vladislav Mladenov, Christian Mainka, and Jörg Schwenk. “On the Security of Modern Single Sign-On Protocols – Second-Order Vulnerabilities in OpenID Connect”. In: (Jan. 7, 2016).
- [59] 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*. New York, NY, USA: ACM, Nov. 2021, pp. 195–208. ISBN: 9781450385275. DOI: 10.1145/3463676.3485600. arXiv: 2103.02579. URL: <http://arxiv.org/abs/2103.02579>.
- [60] Srivathsan G. Morkonda, Sonia Chiasson, and Paul C. Oorschot. *SSOPrivateEye: Timely Disclosure of Single Sign-On Privacy Design Differences*. Sept. 9, 2022. arXiv: 2209.04490 [cs]. URL: <http://arxiv.org/abs/2209.04490> (visited on 04/24/2023). preprint.
- [61] Srivathsan G. Morkonda, Sonia Chiasson, and Paul C. van Oorschot. “Sign in with ... Privacy”: *Timely Disclosure of Privacy Differences among Web SSO Login Options*. Aug. 17, 2023. arXiv: 2209.04490 [cs]. URL: <http://arxiv.org/abs/2209.04490> (visited on 10/21/2023). preprint.
- [62] Srivathsan G. Morkonda, Paul C. van Oorschot, and Sonia Chiasson. “Exploring Privacy Implications in OAuth Deployments”. In: *arXiv:2103.02579 [cs]* (Mar. 2021). arXiv: 2103.02579 [cs].
- [63] *OIDC Code Flow with PKCE for Manually Built Facebook Login Flows*. <https://developers.facebook.com/docs/facebook-login/guides/advanced/oidc-token/>. (Accessed on 08/01/2023). 2023.
- [64] *OpenCV: Template Matching*. [https://docs.opencv.org/4.x/d4/dc6/tutorial\\_py\\_template\\_matching.html](https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html). (Accessed on 06/30/2023).
- [65] OWASP. *Owasp Top Ten*. 2022. URL: <https://owasp.org/www-project-top-ten/> (visited on 10/09/2022).
- [66] *Passkeys (Passkey Authentication)*. <https://fidoalliance.org/passkeys/>. (Accessed on 06/29/2023).
- [67] Pieter Philippaerts, Davy Preuveneers, and Wouter Joosen. “OAuth: Exploring Security Compliance in the OAuth 2.0 Ecosystem”. en. In: *25th International Symposium on Research in Attacks, Intrusions and Defenses*. Limassol Cyprus: ACM, Oct. 2022, pp. 460–481. ISBN: 978-1-4503-9704-9. DOI: 10.1145/3545948.3545955. URL: <https://dl.acm.org/doi/10.1145/3545948.3545955> (visited on 07/06/2023).
- [68] Tamjid Al Rahat, Yu Feng, and Yuan Tian. *Cerberus: Query-driven Scalable Security Checking for OAuth Service Provider Implementations*. May 15, 2022. arXiv: 2110.01005 [cs]. URL: <http://arxiv.org/abs/2110.01005> (visited on 09/20/2022).
- [69] Tamjid Al Rahat, Yu Feng, and Yuan Tian. “OAUTHLINT: An Empirical Study on OAuth Bugs in Android Applications”. In: *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. 2019, pp. 293–304. DOI: 10.1109/ASE.2019.00036.
- [70] Sebastian Roth, Stefano Calzavara, Moritz Wilhelm, Alvis Rabitti, and Ben Stock. “The Security Lottery: Measuring Client-Side Web Security Inconsistencies”. In: *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 2047–2064. ISBN: 978-1-939133-31-1. URL: <https://www.usenix.org/conference/usenixsecurity22/presentation/roth>.
- [71] Takamichi Saito, Satoshi Shibata, and Tsubasa Kikuta. “Comparison of OAuth/OpenID Connect Security in America and Japan”. en. In: *Advances in Networked-Based Information Systems*. Ed. by Leonard Barolli, Kin Fun Li, Tomoya Enokido, and Makoto Takizawa. Vol. 1264. Series Title: Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, 2021, pp. 200–210. DOI: 10.1007/978-3-030-57811-4\_19. URL: [http://link.springer.com/10.1007/978-3-030-57811-4\\_19](http://link.springer.com/10.1007/978-3-030-57811-4_19) (visited on 09/23/2020).
- [72] N. Sakimura, J. Bradley, and N. Agarwal. *Proof Key for Code Exchange by OAuth Public Clients*. RFC 7636. IETF, Sept. 2015. URL: <http://tools.ietf.org/rfc/rfc7636.txt>.
- [73] *searxng/searxng - Docker Image — Docker Hub*. <https://hub.docker.com/r/searxng/searxng>. (Accessed on 06/29/2023).
- [74] *security.txt: Proposed standard for defining security policies*. <https://securitytxt.org/>. (Accessed on 05/09/2023).
- [75] K. Meyer zu Selhausen and D. Fett. *OAuth 2.0 Authorization Server Issuer Identification*. RFC 9207. IETF, Mar. 2022. URL: <http://tools.ietf.org/rfc/rfc9207.txt>.
- [76] Asuman Senol, Gunes Acar, Mathias Humbert, and Frederik Zuiderveen Borgesius. “Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission”. In: *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 1813–1830. ISBN: 978-1-939133-31-1. URL: <https://www.usenix.org/conference/usenixsecurity22/presentation/senol>.
- [77] *ShadowRoot: mode property - Web APIs — MDN*. <https://developer.mozilla.org/en-US/docs/Web/API/ShadowRoot/mode>. (Accessed on 05/08/2023).
- [78] Ethan Shernan, Henry Carter, Dave Tian, Patrick Traynor, and Kevin Butler. “More Guidelines Than Rules: CSRF Vulnerabilities from Noncompliant OAuth 2.0 Implementations”. In: *Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 2015, pp. 239–260.
- [79] Shangcheng Shi, Xianbo Wang, and Wing Cheong Lau. “MoSSOT: An Automated Blackbox Tester

- for Single Sign-On Vulnerabilities in Mobile Applications”. In: *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*. Asia CCS ’19. New York, NY, USA: Association for Computing Machinery, July 2, 2019, pp. 269–282. ISBN: 978-1-4503-6752-3. DOI: 10.1145/3321705.3329801. URL: <https://dl.acm.org/doi/10.1145/3321705.3329801> (visited on 10/24/2023).
- [80] *Sign in with Google features — Authentication — Google for Developers*. <https://developers.google.com/identity/gsi/web/guides/offerings>. (Accessed on 08/01/2023).
- [81] *sitemaps.org - Home*. <https://www.sitemaps.org/>. (Accessed on 06/29/2023).
- [82] Juraj Somorovsky, Andreas Mayer, Jörg Schwenk, Marco Kampmann, and Meiko Jensen. “On Breaking SAML: Be Whoever You Want to Be”. In: *21st USENIX Security Symposium*. Bellevue, WA, Aug. 2012.
- [83] *Specify Geolocation for Playwright tests — BrowserStack Docs*. <https://www.browserstack.com/docs/automate/playwright/ip-geolocation>. (Accessed on 04/24/2023).
- [84] *SSO-Monitor Configuration*. [https://sso-monitor.me/static/schema/landscape\\_analysis.json](https://sso-monitor.me/static/schema/landscape_analysis.json). (Accessed on 06/19/2023).
- [85] *Start and grow your e-commerce business - 3-Day Free Trial - Shopify USA*. <https://www.shopify.com/>. (Accessed on 07/25/2023).
- [86] Ben Stock, Giancarlo Pellegrino, Frank Li, Michael Backes, and Christian Rossow. “Didn’t You Hear Me? – Towards More Successful Web Vulnerability Notifications”. In: *Proceedings of the 25th Annual Symposium on Network and Distributed System Security (NDSS ’18)*. Feb. 2018. URL: <https://publications.cispa.saarland/1190/>.
- [87] Ben Stock, Giancarlo Pellegrino, Christian Rossow, Martin Johns, and Michael Backes. “Hey, You Have a Problem: On the Feasibility of Large-Scale Web Vulnerability Notification”. In: *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, Aug. 2016, pp. 1015–1032. ISBN: 978-1-931971-32-4. URL: <https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/stock>.
- [88] Avinash Sudhodanan, Alessandro Armando, Roberto Carbone, and Luca Compagna. “Attack Patterns for Black-Box Security Testing of Multi-Party Web Applications”. In: *23rd Network and Distributed System Security Symposium (NDSS)*. The Internet Society, 2016. URL: <https://www.ndss-symposium.org/wp-content/uploads/2017/09/attack-patterns-black-box-security-testing-multi-party-web-applications.pdf>.
- [89] Avinash Sudhodanan, Roberto Carbone, Luca Compagna, Nicolas Dolgin, Alessandro Armando, and Umberto Morelli. “Large-Scale Analysis & Detection of Authentication Cross-Site Request Forgeries”. In: *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. 2017 IEEE European Symposium on Security and Privacy (EuroS&P). Paris: IEEE, Apr. 2017, pp. 350–365. ISBN: 978-1-5090-5762-7. DOI: 10.1109/EuroSP.2017.45. URL: <https://ieeexplore.ieee.org/document/7961990/> (visited on 10/18/2023).
- [90] Karin Sumongkayothin, Pakpoom Rachtrachoo, Arnuphap Yupuech, and Kasidit Siriporn. “OVER-SCAN: OAuth 2.0 Scanner for Missing Parameters”. In: *Network and System Security*. Ed. by Joseph K. Liu and Xinyi Huang. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 221–233. ISBN: 978-3-030-36938-5. DOI: 10.1007/978-3-030-36938-5\_13.
- [91] San-Tsai Sun and Konstantin Beznosov. “The devil is in the (implementation) details: an empirical analysis of OAuth SSO systems”. In: *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 2012, pp. 378–390.
- [92] *The Accessibility Tree*. <https://web.dev/the-accessibility-tree/>. (Accessed on 06/30/2023).
- [93] The OpenID Foundation (OIDF). *OpenID Connect Core 1.0*. The OpenID Foundation (OIDF), Feb. 2014. URL: [http://openid.net/specs/openid-connect-core-1\\_0.html](http://openid.net/specs/openid-connect-core-1_0.html).
- [94] *Top 500 Most Popular Websites - Moz*. <https://moz.com/top500>. (Accessed on 10/29/2023).
- [95] *tracker-radar/entity\_map.json at main · duckduckgo/tracker-radar · GitHub*. [https://github.com/duckduckgo/tracker-radar/blob/main/build-data/generated/entity\\_map.json](https://github.com/duckduckgo/tracker-radar/blob/main/build-data/generated/entity_map.json). (Accessed on 04/21/2023).
- [96] Anna Vapen, Niklas Carlsson, Anirban Mahanti, and Nahid Shahmehri. “A Look at the Third-Party Identity Management Landscape”. In: *IEEE Internet Computing 20.2* (Mar. 2016), pp. 18–25. ISSN: 1089-7801. DOI: 10.1109/MIC.2016.38. URL: <http://ieeexplore.ieee.org/document/7420509/> (visited on 09/28/2021).
- [97] Anna Vapen, Niklas Carlsson, Anirban Mahanti, and Nahid Shahmehri. “Third-Party Identity Management Usage on the Web”. In: *Passive and Active Measurement*. Ed. by Michalis Faloutsos and Aleksandar Kuzmanovic. Red. by David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, and Gerhard Weikum. Vol. 8362. Cham: Springer International Publishing, 2014, pp. 151–162. DOI: 10.1007/978-3-319-04918-2\_15. URL: [http://link.springer.com/10.1007/978-3-319-04918-2\\_15](http://link.springer.com/10.1007/978-3-319-04918-2_15) (visited on 10/23/2023).
- [98] Hui Wang, Yuanyuan Zhang, Juanru Li, Hui Liu, Wenbo Yang, Bodong Li, and Dawu Gu. “Vulnerability Assessment of OAuth Implementations in Android Applications”. In: *Proceedings of the 31st Annual Computer Security Applications Conference*. ACSAC 2015: 2015 Annual Computer Security Applications Conference. Los Angeles CA USA: ACM, Dec. 7, 2015, pp. 61–70. ISBN: 978-1-4503-3682-6. DOI: 10.1145/2818000.2818024.

- URL: <https://dl.acm.org/doi/10.1145/2818000.2818024> (visited on 10/22/2023).
- [99] Rui Wang, Shuo Chen, and XiaoFeng Wang. “Signing me onto your accounts through facebook and google: A traffic-guided security study of commercially deployed single-sign-on web services”. In: *33th IEEE Symposium on Security and Privacy (S&P)*. IEEE. 2012, pp. 365–379.
- [100] Rui Wang, Yuchen Zhou, Shuo Chen, Shaz Qadeer, David Evans, and Yuri Gurevich. “Explicating SDKs: Uncovering Assumptions Underlying Secure Authentication and Authorization”. In: *22nd USENIX Security Symposium*. SEC’13. Washington, D.C.: USENIX Association, 2013. ISBN: 978-1-931971-03-4. URL: <http://dl.acm.org/citation.cfm?id=2534766.2534801>.
- [101] Wayback Machine. <https://web.archive.org/>. (Accessed on 04/24/2023).
- [102] WebAuthn.io. <https://webauthn.io/>. (Accessed on 06/29/2023).
- [103] Hanlin Wei, Behnaz Hassanshahi, Guangdong Bai, Padmanabhan Krishnan, and Kostyantyn Vorobyov. “MoScan: A Model-Based Vulnerability Scanner for Web Single Sign-On Services”. In: *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ISSTA ’21: 30th ACM SIGSOFT International Symposium on Software Testing and Analysis. Virtual Denmark: ACM, July 11, 2021, pp. 678–681. ISBN: 978-1-4503-8459-9. DOI: 10.1145/3460319.3469081. URL: <https://dl.acm.org/doi/10.1145/3460319.3469081> (visited on 03/17/2022).
- [104] WHATWG. *HTML Living Standard*. 2022. URL: <https://html.spec.whatwg.org/> (visited on 10/10/2022).
- [105] Luyi Xing, Yangyi Chen, X Wang, and Shuo Chen. “InteGuard: Toward Automatic Protection of Third-Party Web Service Integrations”. In: *Proceedings of 20th Annual Network & Distributed System Security Symposium*. 2013.
- [106] XPath — MDN. <https://developer.mozilla.org/en-US/docs/Web/XPath>. (Accessed on 10/30/2023).
- [107] Ronghai Yang, Wing Cheong Lau, Jiongyi Chen, and Kehuan Zhang. “Vetting Single Sign-On SDK Implementations via Symbolic Reasoning”. In: *27th USENIX Security Symposium (USENIX Security 18)*. Baltimore, MD: USENIX Association, Aug. 2018, pp. 1459–1474. ISBN: 978-1-939133-04-5. URL: <https://www.usenix.org/conference/usenixsecurity18/presentation/yang>.
- [108] Ronghai Yang, Wing Cheong Lau, and Shangcheng Shi. “Breaking and Fixing Mobile App Authentication with OAuth2.0-Based Protocols”. In: *Applied Cryptography and Network Security*. Ed. by Dieter Gollmann, Atsuko Miyaji, and Hiroaki Kikuchi. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017, pp. 313–335. ISBN: 978-3-319-61204-1. DOI: 10.1007/978-3-319-61204-1\_16.
- [109] Ronghai Yang, Guanchen Li, Wing Cheong Lau, Kehuan Zhang, and Pili Hu. “Model-based secu-

urity testing: An empirical study on oauth 2.0 implementations”. In: *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. 2016, pp. 651–662.

- [110] Yuchen Zhou and David Evans. “Automated Testing of Web Applications for Single Sign-On Vulnerabilities”. In: *23rd USENIX Security Symposium*. San Diego, CA: USENIX Association, Aug. 2014. URL: <https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/zhou>.
- [111] Chaoshun Zuo, Qingchuan Zhao, and Zhiqiang Lin. “AUTHSCOPE: Towards Automatic Discovery of Vulnerable Authorizations in Online Services”. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. New York, NY, USA: ACM, Oct. 2017, pp. 799–813. ISBN: 9781450349468. DOI: 10.1145/3133956.3134089. URL: <https://dl.acm.org/doi/10.1145/3133956.3134089>.

## A. Appendix

Table 8 and Table 9 present our complete data for the ground truth (cf. §4.1) and landscape evaluation (cf. §6.1).

	Web-sites	SSO Buttons Window								Integration	
		All	▲	▶	⚙	☐	🔊	🔊	🔊	API	SDK
SSO	GOOGLE	276	331	2	17	312	141	133	56	206	124
	FACEBOOK	213	213	9	15	189	103	106	0	158	51
	APPLE	155	155	0	6	149	96	59	0	112	43
	TWITTER 1.0	51	51	13	9	29	23	15	0	38	0
	QQ	28	28	0	12	16	14	14	0	28	0
	GITHUB	27	27	0	14	13	25	2	0	27	0
	MICROSOFT	25	25	0	3	22	20	5	0	25	0
	LINKEDIN	23	23	2	3	18	10	12	0	22	0
	WECHAT	21	21	0	8	13	12	9	0	21	0
	SINA WEIBO	21	21	0	8	13	10	11	0	21	0
	VK	11	11	0	3	8	3	8	0	11	0
	YAHOO	10	10	1	3	6	5	4	0	9	0
	AMAZON	8	8	0	0	8	5	3	0	8	0
	OK	6	6	0	3	3	1	5	0	6	0
	BAIDU	5	5	0	3	2	3	2	0	5	0
	MAIL RU	5	5	0	1	4	1	4	0	5	0
	ALIPAY	4	4	0	1	3	1	3	0	4	0
	LINE	4	4	0	0	4	4	0	0	4	0
	SLACK	4	4	0	0	4	4	0	0	4	0
	MYVALUE	3	3	0	0	3	3	0	0	3	0
	YANDEX	2	2	0	1	1	1	1	0	2	0
	TAobao	2	2	0	0	2	0	2	0	2	0
	CLEVER	2	2	0	1	1	1	1	0	2	0
	TWITCH	2	2	0	0	2	2	0	0	2	0
	BITBUCKET	2	2	0	0	2	2	0	0	2	0
	ORCID ID	2	2	0	0	2	2	0	0	2	0
	KAKAO	1	1	0	0	1	1	0	0	1	0
	SBERBANK	1	1	0	0	1	0	1	0	1	0
	WIKIPEDIA 1.0	1	1	0	0	1	1	0	0	1	0
	MOS RU	1	1	0	0	1	0	1	0	1	0
	LIVEJOURNAL	1	1	0	0	1	0	1	0	1	0
	PEOPLE	1	1	0	0	1	1	0	0	1	0
	BINANCE	1	1	0	1	0	1	0	0	1	0
	RAMBLER	1	1	0	0	1	1	0	0	1	0
	TAIWANMOBILE	1	1	0	0	1	1	0	0	1	0
	STEAM	1	1	0	0	1	0	1	0	1	0
	GOSUSLUGI	1	1	0	1	0	0	1	0	1	0
	EVERNOTE	1	1	0	1	0	1	0	0	1	0
	AUTODESK	1	1	0	0	1	0	1	0	1	0
	PLAYSTATION	1	1	0	0	1	0	1	0	1	0
	ID ME	1	1	0	0	1	1	0	0	1	0
	DOCOMO	1	1	0	0	1	1	0	0	1	0
	XIAOMI	1	1	0	1	0	0	1	0	1	0
	NINTENDO	1	1	0	0	1	1	0	0	1	0
	GITLAB	1	1	0	0	1	1	0	0	1	0
	DISCORD	1	1	0	0	1	1	0	0	1	0
	FIREFOX	1	1	0	0	1	1	0	0	1	0
All		351	988	27	115	846	505	407	56	751	218
SSO or Password		750	No Auth.	132	Not Reachable	118	Total	1k			

▲ Broken ▶ User Interaction ⚙ Automatable ☐ Topmost 🔊 Popup 🗑 IFrame

TABLE 8: Ground Truth of the Tranco Top 1k Websites. We found 988 SSO buttons on 351 websites (35%). Due to broken SSO buttons or required user interaction, we can automatically analyze 86% of them. *Summary in Table 3.*

		Login Pages						SSO Buttons							
		Login Page Detection Techniques						SSO Detection Techniques				Window			
		All						All							
API		21,473	6,244	9,157	10,687	385	498	24,576	22,827	1,738	0	11	18,776	5,789	0
		18,560	5,460	7,923	9,249	345	373	20,550	19,001	1,549	0	0	15,402	5,148	0
		3,324	916	1,428	1,615	26	62	4,459	4,204	255	0	0	3,652	807	0
		3,360	1,486	1,132	1,110	148	72	3,972	3,636	336	0	0	3,080	892	0
		2,700	1,112	1,111	1,037	47	72	3,037	2,832	205	0	0	2,431	606	0
		1,622	478	678	847	28	42	1,972	1,947	25	0	0	1,718	254	0
		1,533	441	560	860	54	22	1,750	1,581	169	0	0	1,137	613	0
		980	338	483	230	77	15	1,069	1,041	28	0	0	869	200	0
SDK		647	207	325	180	23	11	691	574	117	0	0	574	117	0
		487	164	251	142	15	7	519	487	32	0	0	420	99	0
	GOT <sub>G</sub>	10,607	6,446	2,101	1,437	2,960	325	12,085	0	0	12,085	0	0	0	12,085
	GSI <sub>G</sub>	4,682	1,227	1,658	2,592	80	128	5,213	4,866	347	0	0	22	5,191	0
	SiwA <sub>A</sub>	3,021	465	774	2,135	23	52	3,457	3,330	127	0	0	1,932	1,525	0
API & SDK		47,730	18,423	17,803	18,426	3,949	1,212	88,994	71,676	5,222	12,085	11	50,164	26,734	12,085

Topmost Popup IFrame

Search Engines Crawling Paths and Subdomains Homepage Sitemap

Keywords in DOM Tree Logos of IdPs in Screenshot Patterns in InBC API Calls in Browser

GOT<sub>G</sub> Google One Tap SDK GSI<sub>G</sub> Google Sign-In SDK SiwA<sub>A</sub> Sign in with Apple SDK SiwG<sub>G</sub> Sign in with Google SDK FL<sub>F</sub> Facebook Login SDK

TABLE 9: The Top 1M SSO Landscape. The combination of multiple login page and SSO detection techniques expands the overall coverage. Certain techniques are particularly effective for specific API or SDK integrations. For instance, homepages () and patterns in InBC () are well-suited for detecting the GOT<sub>G</sub> SDK integration. *Summary in Table 6.*