“Only as Strong as the Weakest Link”: On the Security of Brokered Single Sign-On on the Web

Tommaso Innocenti
Northeastern University
Boston, MA, USA
innocenti.t@northeastern.edu
ORCID: 0000-0003-0247-806X

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

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

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

Engin Kirda
Northeastern University
Boston, MA, USA
ek@ccs.neu.edu
ORCID: 0000-0001-9988-6873

***Abstract*—Single Sign-On (SSO) is an authentication process that allows users to access multiple services with a single set of login credentials. Although SSO improves the user experience, it poses challenges to developers to implement complex authentication protocols securely. External services, called brokers, simplify the integration of SSO.**

In this paper, we shed light on the emerging brokered SSO ecosystem, focusing on the security of the newly introduced actor, the broker. We systematically evaluate the landscape of brokered SSO, uncovering significant blind spots in previous research. Our study reveals that 25% of the websites with SSO integrate brokers for authentication, an area that has not been covered by any previous research.

Through our comprehensive security evaluation, we identify three categories of threats associated with brokered SSO: (1) insufficient validation of redirect chains enabling injection attacks, (2) unauthorized data access enabling account takeovers, and (3) violations of security best current practices.

We expose vulnerabilities in over 50 brokers, compromising the security of more than 2k websites. These findings represent only a lower bound of a critical situation, underscoring the urgent need for improved security measures and protocols to safeguard the integrity of brokered SSO systems.

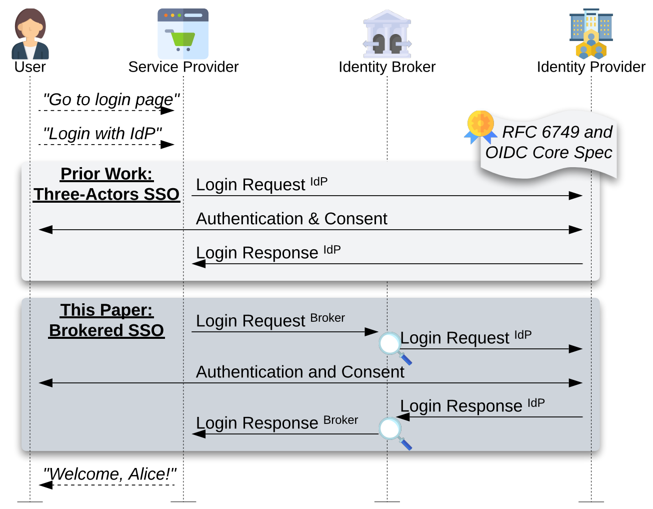


Figure 1: Three-Actors SSO vs. Brokered SSO. Three-actors SSO involves a *user* who wants to log in to an *SP*’s website by using their account at an *IdP*. Brokered SSO involves an additional actor called *broker* that mediates between the SPs and IdPs. This way, SPs implement only one broker while supporting SSO logins with multiple IdPs.

1. Introduction

Single Sign-On (SSO) is a widely used and extensively adopted user authentication technique for websites. Each year, more websites adopt SSO services to manage user identities, in order to reduce the significant burden of signing up and signing in on their sites. This trend is reflected in the forecast of the market share of identity management services, which is expected to reach 43 billion US dollars by 2029 [30]. Prominent providers of SSO services include social networks such as Facebook, Twitter, and Snapchat, as well as large enterprises such as Apple, Google, and Microsoft. Together, these SSO services are utilized by more than 45k websites within the Tranco top 1M sites [34].

The research community also recognizes the importance of SSO and is continually working to detect and mitigate security issues [24, 31, 33, 34, 48, 51, 64]. Additionally, the IETF regularly updates the Security Best Current Practices (SBCPs) to address the state-of-the-art research [39, 59].

Three-Actors SSO. Until now, the SSO security research community focused on specification [19, 20, 21, 40, 44, 47] and implementation [6, 24, 25, 31, 33, 34, 38, 41, 48, 51, 64, 67, 72] shortcomings in SSO login flows that involve *three* actors: the *user* who wants to log in on a *Service Provider (SP)'s* website by using their account at an *Identity Provider (IdP)*. We refer to this as three-actors SSO. Initially, the SP delegates the login request to the IdP, see Figure 1. Once the user authenticates at the IdP, it returns the user's authentication token in the login response back to the SP. The SP uses this token to authenticate the user.

Brokered SSO. This paper sheds light on new trends in which SPs use external services called Identity Brokers (brokers) to integrate SSO into their websites. Brokers offer a streamlined solution for SP developers by consolidating multiple IdPs into a single platform. By integrating a single broker, SPs can offer their users SSO logins with dozens of IdPs out of the box. This method significantly reduces the implementation overhead for SP developers. Additionally, SP developers lacking experience in SSO and its security can delegate the implementation to the more experienced broker developers. SPs also do not need to adjust their implementations when IdPs change theirs. Instead, they rely on the brokers to adapt to any changes. Prominent brokers are Keycloak [36], Amazon Cognito [5], and Google Firebase [26], which are adopted by more than 1k websites.

Figure 1 illustrates how this new actor integrates into the SSO ecosystem. Brokers effectively combine two separate SSO flows into a single SSO login. Therefore, the SP first delegates the user's authentication to the broker, which in turn delegates it to the IdP. In return, the IdP issues the user's authentication token to the broker, which finally generates a new token that it returns to the SP. SPs adopting brokered SSO should be aware that including a broker introduces a new single point of failure into their SSO flow.

Importance of the Brokered SSO Ecosystem. Considering the gap left by prior research, it is crucial to determine the extent to which brokered SSO is used on the web. To answer this, we developed a novel methodology and implemented a fully automated tool called IDB-DETECTOR that reliably detects the wide range of brokers on the web. We used IDB-DETECTOR to analyze the Tranco top 1M websites and identified 249 brokers used on 8,241 websites. In general, 25% of websites prefer to use brokered SSO over three-actors SSO to authenticate their users.

Blind Spots in SSO Security Research. While studying prior work [24, 31, 33, 34, 48, 51, 64], we discovered that the research community only analyzes half of the authentication flows when websites use brokered SSO. Their analyses focus primarily on direct communication with the IdP, including the login request$^{\text{IdP}}$ and / or login response$^{\text{IdP}}$ (see Figure 1). As a result, they miss analyzing all SSO messages that are exchanged between the SP and broker.

In this paper, we close this gap by analyzing the entire authentication flow. Through this comprehensive analysis, we discovered significant security vulnerabilities affecting multiple brokers and numerous websites integrating them. For instance, brokers are actively weakening or even breaking the security of the entire SSO authentication, even when the SP and the IdP implementations are secure. We classify our findings into three dimensions: (1) weak or broken redirection chains, (2) unauthorized data access, and (3) violations of the SBCPs.

Our investigation provides the first in-depth analysis of the brokered SSO ecosystem, revealing more than 50 brokers being susceptible to severe vulnerabilities. For example, 49 brokers do not sufficiently protect the inherently longer redirection chain, allowing attackers to receive confidential messages (see §6). Even worse, 34 brokers are leaking the user's authentication tokens, enabling attackers to take over user accounts on over 2,4k websites. We also discovered 15 brokers that allow attackers to gain unauthorized access to user data on more than 1k websites (§7). Finally, we reveal that on more than 1,5k websites, the broker downgrades the security of the otherwise protected flow (§8).

Extensive Impact of Vulnerabilities in Brokered SSO. We demonstrate that vulnerabilities affecting a single broker can compromise the security of hundreds of SPs. For example, the CivicPlus broker [8] leaks authentication tokens. This flaw allows attackers to compromise user accounts on over 500 US-governmental websites.

Contributions. We make the following contributions:

- ► *Systematization of Brokered SSO*: We systematize all brokers by introducing classification criteria that cover their integration and interaction with the traditional entities in three-actors SSO. Thus, we can fully understand and analyze this new emerging ecosystem. (§3)
- ► *Large-Scale Detection of Brokered SSO*: We implement IDB-DETECTOR, our three-fold system to reliably detect brokers on the web. By analyzing HTTP traffic recordings from the Tranco top 1M websites, we uncover 249 brokers. IDB-DETECTOR identifies 249 brokers and provides generic recognition patterns to re-identify these brokers in future work. (§4)
- ► *Breaking the Redirection Chains in Brokered SSO*: We identify critical vulnerabilities in the redirection chains. We classify them into three patterns: double chain, out-of-order chain, and open-end chain. Although double chains are secure, out-of-order chains and open-end chains are susceptible to CSRF attacks by design. Furthermore, we reveal that 20% of the brokers fail to validate the redirection chain resulting in unauthorized account access and Cross-Site Scripting (XSS). (§6)
- ► *Unauthorized Data Access in Brokered SSO*: For the first time, we reveal how brokers are using three different permission models to obtain access to user data. Although two of these models are secure, one fails to protect user data from unauthorized access by design. Even worse, attackers can escalate this issue to take over the user's account. In summary, 15 brokers expose user data on over 1k websites, and 6 brokers enable account takeovers on 799 websites. (§7)
- ► *The Weakest Link in Brokered SSO*: We estimate the weakest link in brokered SSO by comparing the security of the flows between the SP↔broker and the broker↔IdP. For this, we rely on the official Security Best Current Practices [39], which summarizes

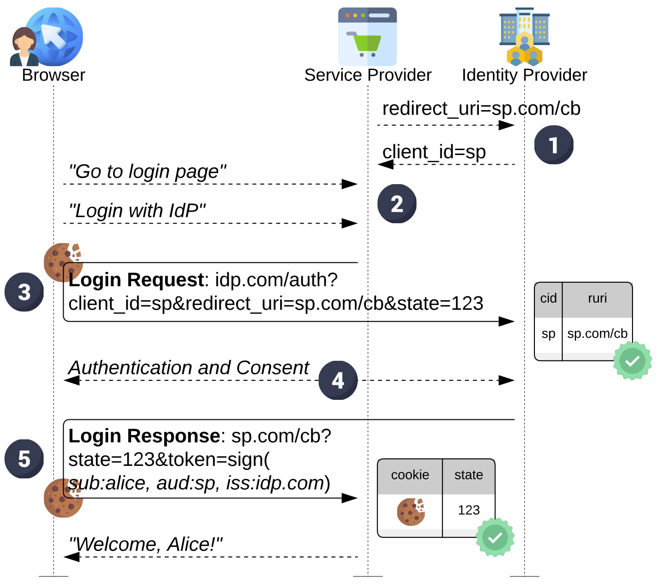


Figure 2: Three-Actors SSO Login Flow. ❶ The developer registers the SP once at the IdP. ❷ The user wants to log in to the SP's website by using their account at the IdP. ❸ The SP delegates the authentication to the IdP. ❹ The user authenticates and grants consent for the SP to access their data. ❺ The IdP returns the user's identity back to the SP.

all threats and implementation shortcomings. Overall, the flow between the SP↔broker turns out to be the weakest link with more than 7k SBCP violations. Even worse, we found over 1,5k violations in which the brokers are downgrading the security of the flow. (§8)

Open Science. We have made all artifacts publicly available as an open-source contribution.[1] These artifacts allow researchers to fully replicate our work. By providing the raw data, source code, and patterns for identifying brokered SSO, we aim to encourage further research on this emerging ecosystem. We strongly believe that brokered SSO will receive more attention from the research community.

Responsible Disclosure. We disclosed vulnerabilities to 55 brokers with insufficient redirection validation or flawed consent patterns, giving them at least six months to fix the issues. We used various contact points to reach out, with some brokers fixing or acknowledging vulnerabilities, while others remain unresponsive. We are contributing our findings to the OAuth SBCPs [39] and OAuth 2.1 draft [29]. More details are included in §B.

2. Background: Three-Actors SSO

Figure 2 provides an overview of the three-actors SSO login flow. This flow enables *users* to log in to multiple *Service Providers (SPs)* by using a single set of credentials associated with their account at an *Identity Provider (IdP)*. The OAuth 2.0 (OAuth) [28] and OpenID Connect 1.0 (OIDC) [53] protocols represent the de facto standard for this delegated authorization and authentication on the web. Their adoption has pervasively reached many websites to enable social logins, such as "Sign in with Facebook" [16].

❶ Pre-Registration. The SP developer must once pre-register their SP application at the IdP. Therefore, the developer provides the URL to which the login response should be sent (`redirect_uri`). The IdP creates an identifier for the SP (`client_id`), tightly associates it with the `redirect_uri`, and stores this pair in a database.

❷ Login Initiation. The user navigates their browser to the SP's login page and clicks on the Single Sign-On (SSO) button that starts the login with the chosen IdP. Then, the SP creates a session cookie, generates a random `state`, binds it to the session cookie, and stores this pair in a database.

❸ Login Request. The SP issues the login request to the chosen IdP by setting the session cookie in the browser and finally redirecting to the IdP. The login request includes the SP's URL to which the login response should be sent (`redirect_uri`), the SP's identifier (`client_id`), and the `state` bound to the session cookie.

❹ User Authentication and Consent. The IdP validates if the `redirect_uri` is associated with the `client_id`, and if successful, asks the user to authenticate and grant the SP access to their data. If the user is already authenticated on the IdP and has previously granted access, this step is entirely skipped without further user interaction.

❺ Login Response. If the user accepts the consent, the IdP redirects the user's authentication token in the login response back to the SP's `redirect_uri`. The IdP signs the token to certify its authenticity. It holds claims about the user's identity (`subject`), the intended recipient of the token (`audience`), and the issuer of the token (`issuer`). Upon receiving the login response, the SP verifies all claims. The login response must further reflect the `state` of the login request. Otherwise, the SP aborts the flow. Thus, the `state` binds both the login request and response to the user's session, which prevents injecting SSO messages.

3. Systematization of Brokered SSO

This section sheds light on the new actor introduced in the brokered SSO ecosystem, providing the first systematic analysis of Identity Brokers (brokers) on the web. This systematization provides a comprehensive understanding of the new interactions within this ecosystem. To establish a foundation for this study, we manually curated a ground truth (§3.1). Our analysis, grounded in this data, reveals various characteristics of brokers. We identify notable variations in their deployment (§3.2), the audience that has access to them (§3.3), and their standard compliance (§3.4). We conclude with a discussion of security implications (§3.5).

3.1. Ground Truth

Research Motivation. To date, the SSO research community has always assumed the SPs to redirect to the IdPs

1. https://doi.org/10.5281/zenodo.13918427

(cf. Step 3 in Figure 2). However, early experiments in this work proved that this belief is false. We observed that some SSO flows contain additional requests between the SPs and IdPs. Prompted by curiosity about their purpose and necessity, we decided to establish a ground truth and manually analyze this dataset.

Ground Truth Dataset. We use the public dataset from prior work that includes data of 45,532 websites with SSO [34, §1]. Since one website can integrate multiple IdPs, the data includes 88,994 recorded SSO login flows in total. Our ground truth encompasses 50 randomly sampled SSO flows from a diverse range of websites, including highly popular and less frequented ones. Each SSO flow is unique and associated with a different website. The sample represents SSO logins with 8 IdPs, distributed as follows: Google (17x), Facebook (16x), Apple (7x), LinkedIn (4x), Microsoft (3x), QQ (1x), Twitter (1x), and WeChat (1x).

Ground Truth Analysis. Driven by our observations, we posed three questions regarding the intermediary requests between SPs and IdPs: (1) *"How many are there?"* (2) *"To whom are they sent?"* (3) *"Do they share any common characteristics?"* Our analysis provides the answers:

(1) Only 18% of the SSO flows directly redirect to the IdP (9 of 50). In contrast, 72% include an additional request between the SP and IdP (36 of 50). The remaining 10% involve a total of two intermediary requests (5 of 50).

(2) We analyzed the domains of intermediary requests and discovered that these actors provide (free or paid) identity services, including SSO. This initial encounter with such intermediaries initiated our research on brokers.

(3) We thoroughly examined all intermediary requests in our ground truth and compared them. This analysis helped to identify key characteristics, enabling us to systematize brokers in three dimensions: (a) First- vs. Third-Parties (§3.2), (b) Audience (§3.3), and (c) Compliance (§3.4).

3.2. Systematization: First- vs. Third-Parties

We classify brokers by their relationship towards the SPs and IdPs. For example, brokers can be deployed as first parties or third parties. We define this relationship based on the eTLD+1 – also referred to as the site – on which the broker is running. As an example, co.uk is the eTLD and bar.co.uk is the eTLD+1 (called site) of foo.bar.co.uk. The site provides the lowest-level web browser security mechanism [56] to isolate websites from different parties.

Third-Party Brokers. We found brokers that deploy a single broker instance, which is shared by multiple SPs, on a central domain like idb.com. In this case, the broker has a cross-site relation to the SP and IdP. Alternatively, the broker could allocate multiple broker instances on separate subdomains for each SP, i.e., sp1.idb.com, sp2.idb.com, etc. In this case, the broker still has a cross-site relation to the SP and IdP but each SP has its exclusive instance.

First-Party Brokers. We also found brokers being deployed on the SP's domain sp.com such that the broker has a same-site relation to the SP. In this case, the broker is either integrated as a library into the SP application or the SP implements its own custom broker. Beyond that, we found SPs using CNAME DNS records such that idb.sp.com points and resolves to idb.com, a third-party broker. This CNAME cloaking evades the browser's security barrier [56] by sharing first-party cookies with third-party entities [4].

First-Parties vs. Third-Parties. For developers, third-party brokers are easier to integrate because SSO is operated by the third party and runs decoupled from the SPs. On the other side, first-party brokers running on the same site as the SPs are not affected by the third-party cookie phaseout [49, 55]. Blocking third-party cookies will cause certain SSO flows on third-party brokers to break [18]. For instance, the Auth0 broker can be integrated as a first party or third party. If used as a first party, it supports a "silent authentication flow" without user interaction in which the SP can fetch renewed tokens from the broker that is embedded in an iframe [10]. If used as a third party, the iframe has no access to third-party cookies and the token refresh will fail [11].

3.3. Systematization: Audience

Companies such as Apple, Facebook, and Google offer social login services to millions of websites. Anyone can create a free or paid developer account with these companies and set up their corresponding SPs (publicly accessible). This principle also applies to brokers. However, some of them only serve a restricted or somewhat related group of SPs (only internally accessible). Thus, we classify brokers based on the audience to which they provide their services.

Public Brokers. Publicly accessible brokers offer SSO to arbitrary SPs, whether as a paid or free service, or as an open-source option for self-hosted deployments. Prominent examples are Microsoft Entra ID (formerly known as Microsoft Azure Active Directory [43]) and plugins for the Shopify [7], WordPress [71], Magento [2], PrestaShop [50], and WooCommerce [70] platforms.

Internal Brokers. Internally accessible brokers offer SSO only to a restricted set of SPs. For example, Atlassian [68] runs an internal broker for their users to have one account across all its proprietary services, including Jira, Confluence, and Trello. Other internal brokers provide their services exclusively within specific ecosystems, such as those comprised of scholarly or US governmental websites.

Automatic Classification. If all SPs using a specific broker are related, we classify the broker as internally accessible. If at least one SP is unrelated to the others, we classify the broker as publicly accessible. For example, consider the brokers B1 and B2. B1 is used by the two SPs adidas.com and adidas.co.uk. B2 is used by at least two SPs, including nytimes.com and ebay.com. Since all SPs using B1 are related, we consider B1 to be internally accessible. Since B2 is used by at least two unrelated SPs, we consider B2 to be publicly accessible.

To automatically determine if two or more SPs are related, we proceed as follows: (1) We compare registrable domains, excluding the eTLDs. (2) We consult the Tracker Radar Entity List [65] that associates websites with companies and was already used for similar purposes in prior work [60]. (3) We examine if the TLS certificates share information like the digest, serial number, public key, subject organization and unit, common name, email, and alternative names. (4) We consult the Whois [22] records. (5) We compare the website titles and contents.

3.4. Systematization: Standard Compliance

By taking part in the SSO login flow, we further classify brokers by their compliance to standardized SSO protocols.

Compliant Brokers. Brokers are considered *compliant* if they are based on OAuth [28] or OIDC [53]. Both require the login request to contain a `response_type` parameter.

Custom brokers. We define brokers to be *custom* if they use proprietary parameters. Custom brokers use parameters having the same semantic meaning but a different syntactic spelling. As an example, `app_id` represents the `client_id` and `returnTo` refers to the `redirect_uri` parameter.

3.5. Security Considerations

Our systematization is crucial to determine the attacker model applicable to the different brokers, define the evaluation scope, and assess the impact of vulnerabilities.

Attack Surface. Introducing an additional actor into the SSO login flow elevates the attack surface. However, there are different security considerations depending on whether the broker is a first- or third party and its compliance with the specifications. First-party brokers may use same-origin communication techniques with the SP, which are inherently protected by the Same Origin Policy (SOP) [56]. For example, a broker can write a token in localStorage that the SP then uses for authenticating the user's requests. In contrast, third-party brokers must use cross-origin communication techniques (i.e., HTTP redirects, postMessage [69], etc.) that require critical validation mechanisms [33, 63]. In addition, compliant brokers can follow the Security Best Current Practices (SBCPs) [39], which addresses all known protocol weaknesses. For custom brokers, developers have to take extra care to resolve these issues and mitigate attacks.

Attack Impact. If a broker is vulnerable, the impact might vary depending on whether the broker is a first- or third party. For example, XSS on a third-party broker affects the broker and the user's authentication on all SPs. But even worse, XSS on a first-party broker that is deployed on the same origin as the SP compromises the entire SP app, elevating the impact beyond authentication.

Attack Prevalence. The prevalence of vulnerabilities in the wild depends on whether a broker is publicly or internally accessible. For example, flaws in publicly accessible brokers could potentially impact hundreds or even thousands of SPs that incorporate the identity services of the brokers. In contrast, weaknesses in internal brokers only affect a limited group of SPs managed by the same company.

4. Automatic Detection of Brokered SSO

This section introduces the first automated large-scale detection of brokered SSO on the web. To achieve this, we developed a tool called IDB-DETECTOR, designed to reliably identify brokers used by the Tranco top 1M websites.

4.1. System Overview

Step 1: Automated Detection. IDB-DETECTOR is our automated framework built to detect brokers in SSO login flows. It requires HTTP traffic recordings of SSO logins as input. Since these recordings contain many irrelevant HTTP messages, it first reconstructs the SSO login flow. It begins with identifying SSO messages initiated by the SP and ends at the IdP. Then, it utilizes two methods to determine the presence of a broker: (1) *backtracking* identifies third-party and compliant brokers and (2) *clustering* detects first-party and custom brokers. We applied IDB-DETECTOR on more than 55k SSO login flows and detected 249 brokers. For each identified broker, IDB-DETECTOR outputs a unique regular expression pattern. These patterns will allow future research to locate brokers in arbitrary HTTP traffic recordings.

Step 2: Automated Classification. After detecting the brokers, we systematically categorize them according to our established criteria from §3. For each broker, we assess: (1) its classification as either a first or third party (§3.2), (2) its target audience, whether public or internal (§3.3), and (3) its protocol usage, whether compliant or custom (§3.4).

Step 3: Contact Address (Optional). The first two steps have yielded all the necessary data for the evaluations in this paper. However, identifying the companies or institutions that provide the brokers remains essential for responsible disclosure. To address this, we manually searched for the broker patterns using Google Search, Whois Lookup [22], and GitHub Code Search [57].

4.2. IDB-DETECTOR

Figure 3 shows how IDB-DETECTOR can automatically detect brokers in brokered SSO login flows at scale.

Input: HAR Files. We use a public dataset from prior work [34]. These recordings include 88,983 SSO flows across the Tranco[2] 1M websites. They contain the HTTP traffic recorded while running the SSO logins. Each SSO flow is captured in the JSON-based HAR data format [27].

❶ Extract Login Redirection Chains (LRCs). We start by filtering out non-essential requests from the HAR files, such as images, scripts, and fonts, to focus solely on the SSO flow. We include all top-level requests triggered after loading the SP's login page and clicking the SSO button into the LRC. The process ends once we reach the IdP. We capture both server-initiated requests (by analyzing the `Location` header in `3xx` redirects) and JavaScript-initiated requests (by analyzing the `Sec-Fetch-Dest` header). We also consider multiple frames, such as iframes and popups.

❷ Backtracking. We use a backtracking technique to identify all third-party and compliant brokers. The process begins at the last entry in the LRC, the IdP. We examine its `redirect_uri` parameter to which it will redirect back.

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

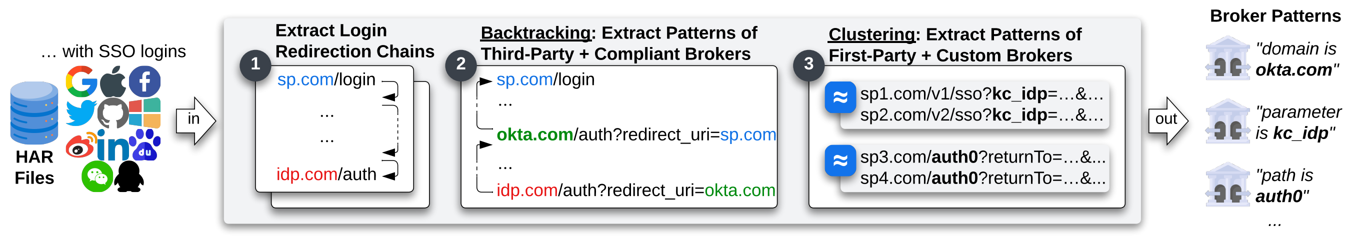

... with SSO logins
HAR Files
in
1 Extract Login Redirection Chains
sp.com/login
...
...
idp.com/auth
2 Backtracking: Extract Patterns of Third-Party + Compliant Brokers
sp.com/login
...
okta.com/auth?redirect_uri=sp.com
...
idp.com/auth?redirect_uri=okta.com
3 Clustering: Extract Patterns of First-Party + Custom Brokers
sp1.com/v1/sso?kc_idp=...&...
sp2.com/v2/sso?kc_idp=...&...
sp3.com/auth0?returnTo=...&...
sp4.com/auth0?returnTo=...&...
out
Broker Patterns
"domain is okta.com"
"parameter is kc_idp"
"path is auth0"
...

Figure 3: Overview of IDB-DETECTOR. We introduce two novel approaches, backtracking and clustering, to uncover a diverse set of brokers. By following this methodology, we discovered a total of 249 brokers.

This URL must differ from the SP and IdP, indicating a third party. Next, we move upwards in the LRC until we find a request sent to this third party. This request must also include a `redirect_uri`, which now points back to the SP. If so, IDB-DETECTOR returns a pattern for this request.

❸ **Clustering**. We use a clustering technique to identify first-party and custom brokers. Our hypothesis is that all requests to the same brokers share similar, if not identical paths and parameters, regardless of the SPs issuing them.

The clustering technique works as follows: We correlate *all* intermediary requests from *all* LRCs of *all* SSO flows. We compare and group similar requests into clusters based on the domain, path, and parameters. Two requests are similar if all but one of their parameters or path components match. We repeat this process iteratively.

For example, consider three requests: R1, R2, and R3. R1 and R2 differ by one path component but are otherwise identical. R2 and R3 differ by one parameter but are otherwise identical. Since R1 and R2 form a cluster, and R2 and R3 forms a cluster, and all three requests join a single cluster. If a cluster contains requests from at least two different SPs, we mark it as an broker cluster. We choose a unique parameter or path component that only appears in this cluster to generate a pattern for this cluster.

Output: Broker Patterns. IDB-DETECTOR creates patterns to identify broker requests in HTTP traffic. Each pattern is a regular expression defining the domain, path, or parameters of a request. These patterns should lay the foundation for future work investigating the brokered SSO.

Accuracy of the Broker Patterns. We are able to detect *all* compliant brokers in the dataset [34], as they consistently use well-defined parameters, such as `client_id`. However, we may miss custom brokers that employ unpredictable parameter names and values. In our ground truth (see §3.1), we successfully identified all 10 manually found brokers, without false negatives. Furthermore, we manually verified and annotated each of the 249 automatically identified brokers, confirming that there were no false positives.

4.3. The Brokered SSO Ecosystem

We used IDB-DETECTOR to capture the first snapshot of the brokered SSO ecosystem, demonstrating its practical relevance. Therefore, we enriched the publicly available data from recent work [34], which already identified 88,983 SSO flows across the Tranco 1M websites, see Table 1.

		# Brokers		# Flows		# Websites	
Three-Actors		–	–	39,870	72%	24,880	75%
Brokered	Public	64	26%	10,911	71%	5,944	72%
	Internal	117	47%	3,442	22%	1,635	20%
	Unknown	68	27%	1,139	7%	686	8%
		249	100%	15,213	28%	8,241	25%
Three-Actors or Brokered				55,083	100%	33,121	100%

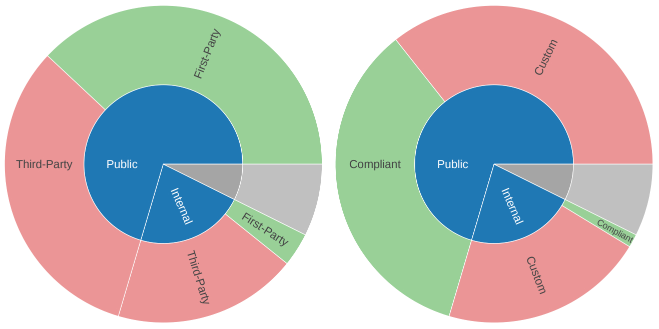
Table 1: Brokered SSO on the Tranco Top 1M Websites. In total, 25% of the websites use brokered over three-actors SSO. Interestingly, 47% of the brokers are internally accessible while 72% of the websites use public brokers.

Excluded SSO Flows. Nearly 30% of the SSO flows (26,399) use social SDKs from IdPs, like “Sign in with Apple JS” [62]. Social SDKs are strictly tied to IdPs and cannot involve any brokers. Therefore, we excluded them from our analysis. We further excluded 7,501 SSO flows due to multiple issues preventing their analysis (see Table 5).

Three-Actors vs. Brokered SSO. Our analysis of brokered SSO is based on the remaining 55,083 SSO flows associated with 33,121 websites. IDB-DETECTOR identified 249 brokers used in 15,213 SSO flows. Websites with three-actors SSO integrate 1.6 SSO flows on average. In contrast, websites with brokered SSO integrate an average of 1.85 SSO flows. This slight increase suggests that brokers make it easier for SPs to adopt multiple IdPs.

Takeaway 1: Every fourth website with SSO, which is not based on social SDKs, uses the brokered instead of the three-actors flow. This observation underscores the importance of research into this emerging ecosystem.

Internal vs. Public Brokers. We discovered that nearly twice as many brokers are internally accessible (117) compared to publicly accessible ones (64). However, the majority of websites (72%) use a publicly accessible broker. This makes sense, as public brokers are available to everyone, while internal brokers are limited to a few websites. On average, each public broker is used by 93 websites (5,944 / 64), whereas internal brokers serve only 14 websites on average (1,635 / 117). Thus, public brokers affect a wider audience if found to be vulnerable. We could not classify 68 brokers due to missing entries in the Tracker Radar Entity List / Whois records / certificates, timeouts, etc. (see §3.3).



(a) First- vs. Third Parties. (b) Compliant vs. Custom.

Figure 4: Comparison of Public and Internal Brokers. Internal brokers are mostly third parties and use custom protocols. Public brokers show a balanced distribution.

Figure 4 further illustrates the differences between public and internal brokers. Internal brokers are mostly third parties, while public brokers are equally likely to be a first- or third party. Furthermore, internal brokers use primarily custom protocols, while public brokers use both compliant and custom protocols equally. These findings highlight the practical importance of our systematization of brokers.

5. Threat Model

Web Attacker. In this paper, we consider the regular *web attacker* [3] who hosts a malicious website, for instance, attacker.com, and can lure the victim to visit it. This is in line with recent research on the security of SSO on the web [24, 31, 33, 34, 48, 51, 64]. We exclude any software or hardware deficiencies, and we assume standard-compliant, non-compromised web browsers and secure TLS. In the following, we reveal an abstract overview of the victim's behavior and the attacker's approach and goals.

Victim. The victim utilizes SSO to authenticate on a trustworthy SP using their account at a trusted IdP. We assume that the victim is already logged in at the IdP and has given consent for the IdP to share personal data with the SP. Therefore, the SSO authentication is seamless and does not require user interaction. As a result, the attacks discussed in this paper work stealthily and do not require the victim to interact with trustworthy parties. Instead, the victim visits the malicious website and clicks on one button.

Attacker Approach and Goals. From the malicious website, the attacker navigates to a vulnerable endpoint of the broker, either by opening a popup or redirection. The victim loads the malicious website, which exploits missing or insufficient validation mechanisms to (1) inject messages, breaking the message authenticity, or (2) receive messages, breaking the message confidentiality. For example, attacker.com could inject a maliciously crafted login request or receive the login response containing the victim's authentication token to take over the account. The attack is successful if the malicious website can receive a security-relevant SSO message from the vulnerable broker. Similarly, the attack succeeds if the malicious site can inject an SSO message that is accepted by the vulnerable broker.

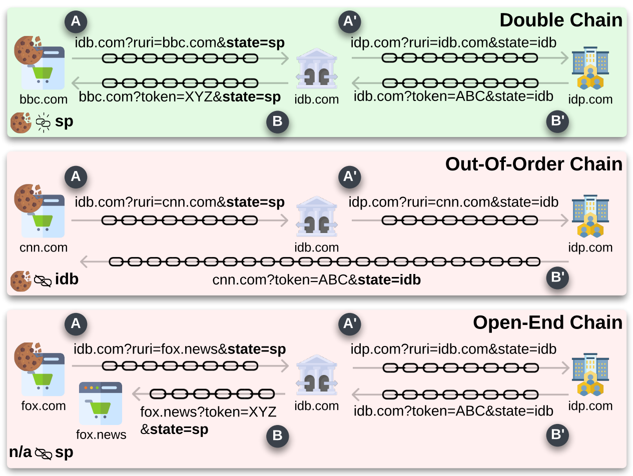

Double Chain
A
A'
idb.com?ruri=bbc.com&state=sp
idp.com?ruri=idb.com&state=idb
bbc.com
bbc.com?token=XYZ&state=sp
idb.com
idb.com?token=ABC&state=idb
idp.com
sp
B
B'
Out-Of-Order Chain
A
A'
idb.com?ruri=cnn.com&state=sp
idp.com?ruri=cnn.com&state=idb
cnn.com
idb.com
idp.com
idb
cnn.com?token=ABC&state=idb
B'
Open-End Chain
A
A'
idb.com?ruri=fox.news&state=sp
idp.com?ruri=idb.com&state=idb
fox.com
fox.news?token=XYZ &state=sp
idb.com
idb.com?token=ABC&state=idb
idp.com
fox.news
B
B'
n/a sp

Figure 5: Redirection Chains in Brokered SSO. The double chain encapsulates two three-actors SSO flows into a single brokered SSO flow and is considered secure. The out-of-order chain is misdirected by the broker, which instructed the IdP to return the login response directly to the SP. The open-end chain is again misdirected by the broker, which returns the login response to the SP operating on a different site. Both, the out-of-order chain and the open-end chain are deemed insecure, as they fail to protect the integrity of the message channel between the SP and broker.

6. Redirection Chains in Brokered SSO

In this section, we are the first to investigate the redirection chains in brokered SSO. For three-actors SSO, we expect the SP to redirect to the IdP with the login request, and the IdP to redirect back to the SP with the login response (cf. Figure 2). However, brokered SSO extends the redirection chain and inherently increases its attack surface. In this scenario, the SP redirects to the broker, which redirects to the IdP. The process reverses with the IdP redirecting back to the broker, and the broker redirecting back to the SP.

In §6.1, we demonstrate that this sequence is misdirected in certain implementations of brokered SSO, allowing attackers to inject arbitrary SSO messages. In §6.2, we show that some redirection chains are not properly validated even if the sequence is aligned, enabling attackers to receive confidential SSO messages.

6.1. Misdirected Redirection Chains

We used our ground truth (see §3.1) as a basis to investigate how the inherently longer redirection chains are shaped in brokered SSO. From this investigation, we identified three common patterns, as shown in Figure 5 and described below.

Methodology. Our automated analysis is based on the dataset obtained from prior work [34]. Since the prior work did not execute the full SSO login flow, the dataset does not contain the login responses. However, we have access to the login requests, which include the `redirect_uri` parameter indicating the URL to which the the login response is sent. We use this information to reconstruct the entire redirection chain, including both login requests and login responses.

Starting with 15,213 brokered SSO login flows (see Table 1), we filtered out flows that are not compliant (leaving 5,668), do not use a `redirect_uri`, or use an unparsable `redirect_uri`. In total, we could fully reconstruct the redirection chains for 3,926 brokered SSO login flows.

Double Chain. The double chain integrates two three-actors SSO flows into a single brokered SSO flow, as shown in Figure 5. The SP generates a random `state`, binds this `state` to the user's session cookie, and issues the login request to the broker (Ⓐ). The broker proceeds similarly and issues the login request with its own `state` to the IdP (Ⓐ'). The IdP generates the user's authentication token and returns it with the login response to the broker, also reflecting the broker's `state` (Ⓑ'). The broker, in turn, returns the login response reflecting the SP's `state` to the SP (Ⓑ). Since all `states` are bound to the user's session cookie, the entire flow is protected. Attackers cannot inject SSO messages because they neither know the `state` nor the session cookie. We found that 3,800 brokered SSO login flows implement the double chain.

Out-Of-Order Chain. The out-of-order chain misdirects the brokered SSO flow, as illustrated in Figure 5. Similar to the double chain, the SP issues the login request to the broker (Ⓐ), which then sends a new login request to the IdP (Ⓐ'). However, the IdP is misdirected by the *redirect_uri* received from the broker in the login request and returns the login response directly to the SP. This causes a mismatch between the `state` returned in Ⓑ' and the `state` bound to the session cookie in Ⓐ. Although the SP uses the token to authenticate the user, it cannot properly validate the `state` by design [28, 39]. Consequently, the SP becomes vulnerable to CSRF attacks, being unable to distinguish between legitimate login responses in Ⓑ' and forged ones. Attackers can exploit this vulnerability by injecting their tokens into the victim's browser, effectively logging the victim into the attacker's account. We identified 69 brokered SSO login flows with the out-of-order chain.

Open-End Chain. As shown in Figure 5, the broker misdirects its login response in the open-end chain. In this scenario, the broker returns the login response to the SP as expected. However, the SP receiving the login response in Ⓑ operates on a different site than the SP that initiated the login request in Ⓐ. Since the `state` in Ⓐ is bound to the session cookie scoped to fox.com, the SP running on fox.news cannot access this cookie. Consequently, the SP cannot validate whether the `state` received in the login response in Ⓑ matches the `state` bound to the session cookie in Ⓐ. Similar to the out-of-order chain, attackers can exploit this CSRF vulnerability by injecting forged login responses to fox.news. We identified 57 brokered SSO login flows implementing the open-end chain.

Takeaway 2: Out-of-order chains and open-end chains misdirect the redirection chains in brokered SSO, enabling CSRF attacks by design. We found 126 brokered SSO login flows affected by this issue. In contrast, double chains, which encapsulate two three-actors SSO flows into a single brokered SSO flow, are secure by design.

6.2. Open Redirection Chains

Both IdPs and brokers must validate the redirection chain to ensure that only authorized entities receive access to the user's token. This validation involves checking the URL to which they issue the token (`redirect_uri`) [28, 39]. Failure to do so allows attackers to manipulate the redirection chain, gaining access to the user's token and potentially taking over the account. Prior research has proven that the majority of IdPs only *partially* validate the `redirect_uri` [31, 48]. In this paper, we are the first to reveal that 20% of the brokers do not validate the `redirect_uri` *at all*, which poses an even greater risk.

Manual Analysis. We opted for a manual approach to empirically evaluate if brokers correctly validate the redirection chain for the following reasons: (1) Executing the full SSO login is challenging to automate and prone to failures [15, 24, 31, 35, 61, 72]. Previous studies that automated the full SSO login are error-proned and difficult to replicate [34]. We aim to avoid these automation issues (i.e., false positives / negatives) and since we only need to test 249 brokers rather than millions of SPs, manual analysis seems suitable. (2) We need to manipulate the URL to which the token is issued. Beyond the standardized `redirect_uri` parameter [28], many custom ones exist, such as `callback`, `returnTo`, and `backURL`. Detecting all these parameters automatically is difficult and remains a problem for future research. (3) We must interpret the leaked data to determine if it contains sensitive information, such as tokens or user data. Similarly, we must test if a leaked token can be redeemed by an attacker to take over the victim's account at the SP. Automating this process is challenging [31, 33] and remains an open problem.

Methodology. For each of the 249 brokers, we followed these steps: First, we visited the login page of a randomly sampled SP that integrates the broker. We logged in using SSO with one of our IdP testing accounts. If the broker performs a server-side redirect, we set the `redirect_uri` (or custom parameters like `callback`) to the malicious website. If the broker performs a client-side redirect, we additionally applied the `javascript` pseudo-protocol handler [12] to test for XSS. Since we aimed to measure a lower bound of vulnerabilities, we did not test further URL validation bypasses. We considered the attack successful if the broker accepted the manipulated `redirect_uri` and redirected to the malicious website or triggered the XSS payload. Upon a successful redirect to the malicious website, we inspected the data received by the attacker. We identified three patterns: (1) the broker leaks tokens (🪪), (2) the broker leaks user data such as names and emails (👤), (3) the

# Vulnerable		Vulnerabilities		Impact	
Brokers	SPs	Open Redirect	XSS	Leak	Account Takeover
20	1,093	●	–	🪪	✓
9	610	●	○	🪪	✓
8	337	●	–	⊟	✗
5	247	●	●	🪪	✓
2	94	●	○	⊟	✗
1	17	●	●	👤	✗
1	11	○	●	⊟	✗
1	5	●	–	👤	✗
1	4	●	○	🪪	✗
1	3	●	–	🪪	✗
49	2,421	48	7	36x 🪪, 2x 👤, 11x ⊟	34x ✓, 15x ✗

● Vulnerable, ○ Secure, – Not Applicable
🪪 Token, 👤 User Data, ⊟ No Data
✓ Confirmed Account Takeover, ✗ No Account Takeover

Table 2: Insufficient Validation of Redirection Chains in Brokered SSO. We identified 49 brokers used by 2,421 SPs that fail to validate the redirection chain, leading to XSS, user data leakage, and account takeovers. *Details in Table 6.*

broker leaks no data but still redirects to the malicious site (⊟). If tokens were leaked, we tried to redeem them to determine if attackers can take over the victim's account.

Results. Table 2 summarizes the results of our empirical evaluation. Surprisingly, 49 brokers did not validate the redirection chain, which is alarming. These vulnerabilities enable Open Redirects (ORs) on 48 brokers and XSS on 7 brokers. At least 2,421 different SPs are affected by these vulnerabilities, and the actual number may be even higher beyond our dataset. We manually confirmed *token leaks* for 34 of 36 brokers, which allow account takeover attacks on 1,950 SPs (cf. gray-shaded rows). Only two brokers leaked tokens that we could not redeem. Another two brokers leaked user data, including identifiers, emails, and profile pictures, while 11 brokers did not leak any data.

Takeaway 3: The redirection chain is not validated by 20% of the brokers. Attackers can exploit this vulnerability to hijack authentication tokens, enabling them to take over the victim's account on a total of 1,950 SPs.

7. Unauthorized Data Access in Brokered SSO

In this section, we explore how attackers can gain access to the resources of users without having to obtain their consent. This threat comes from a flawed consent mechanism that is inherent to brokers and brokered SSO. Even worse, attackers may take over and impersonate the victim's account on *all* SPs which integrate the vulnerable broker. To our knowledge, we are the first to quantify this issue.

First, §7.1 briefly explains the process and reasons for obtaining the user's consent in three-actors SSO. Next, §7.2 delves into brokered SSO and its three different patterns of acquiring user consent. Each of them provides varying levels of data access transparency to the user and one fails to safeguard user data, enabling unauthorized data access. Finally, §7.3 presents our large-scale measurement of unauthorized data access in the brokered SSO ecosystem.

7.1. Consent in Three-Actors SSO

Consent. Consent is an essential mechanism that allows users to control which SPs are allowed to receive their data from the IdP. Consider a user who logs in to an SP such as sp.com for the first time using their IdP account. Consequently, the user must grant consent to the IdP so that sp.com is allowed to access their personal data, including email, birthday, and address. Without such consent, any site, including harmful ones, could access the user's data without permission, severely jeopardizing the SSO system.

Consent Page. To protect and inform users, IdPs show them a dedicated consent page for each SP upon their initial login. This consent page serves as a critical checkpoint. Popular IdPs, such as Apple, Facebook, and Google, display an interface that includes details like the SP's name (e.g., "Shop XYZ"), site (e.g., sp.com), and requested resources (e.g., id, email, address). Users can then approve access.

Client Identifier. Upon account creation, developers manually pre-register their SPs. During pre-registration, the IdP assigns the `client_id` (CID), a unique identifier, to the SP. The IdP stores SP-related configuration, such as the CID and allowed `redirect_uris` (RURIs). It is important to note that users give their consent based on the CID, rather than on a per-site basis. Thus, users only have to provide consent a single time, allowing various sites (such as those with country-specific TLDs) to access their resources.

One Consent for One Third Party. According to the General Data Protection Regulation (GDPR), third parties are processors who, under the authority of a controller, are authorized to process personal data [23, §4.10]. Controllers must have a legal reason for sharing personal data with third parties, such as obtaining consent from the data subjects [23, §6.1]. In three-actors SSO, this sentence holds as the IdP (controller) obtains consent to share user data (subject) with the SP (processor).

7.2. Consent in Brokered SSO

Theory: Provide Consent to Every Third Party. In brokered SSO, an additional actor, which is the broker, obtains access to the user's data. To align with GDPR regulations, users must consent to each third party receiving access to their data (both SP and broker). Requiring users to give consent twice during the SSO login process leads to a cumbersome login experience. Moreover, since users are generally unaware of brokers, they may find it confusing and unsettling to authorize these lesser-known third parties.

Real-World: Implicit Trust in the Broker. In practice, explicit consent is not provided. By incorporating brokers into the login process, SPs place trust into them. SPs allow brokers to access their users' data. However, IdPs show only a single consent page to the users, thus effectively hiding the presence and the active role of the brokers. Figure 8 shows an example in which Google (IdP) asks the user to

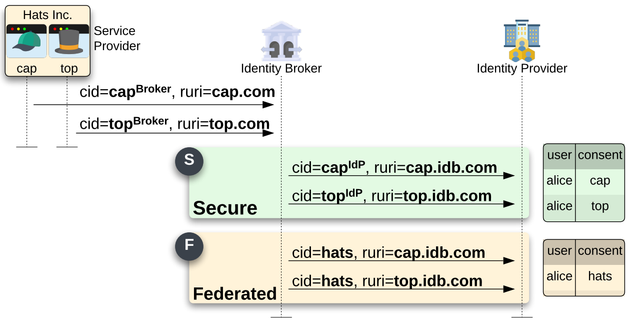

Hats Inc.
Service Provider
cap
top
Identity Broker
Identity Provider
cid=**cap**^Broker, ruri=**cap.com**
cid=**top**^Broker, ruri=**top.com**
S
Secure
cid=**cap**^IdP, ruri=**cap.idb.com**
cid=**top**^IdP, ruri=**top.idb.com**
user | consent
alice | cap
alice | top
F
Federated
cid=**hats**, ruri=**cap.idb.com**
cid=**hats**, ruri=**top.idb.com**
user | consent
alice | hats

Figure 6: Secure and Federated Data Access. **S** The user grants consent to each SP. There are distinct CIDs between SP↔broker and broker↔IdP. **F** The user grants consent to a joint federation of multiple SPs. Multiple SP↔broker CIDs are mapped to the same federated broker↔IdP CID.

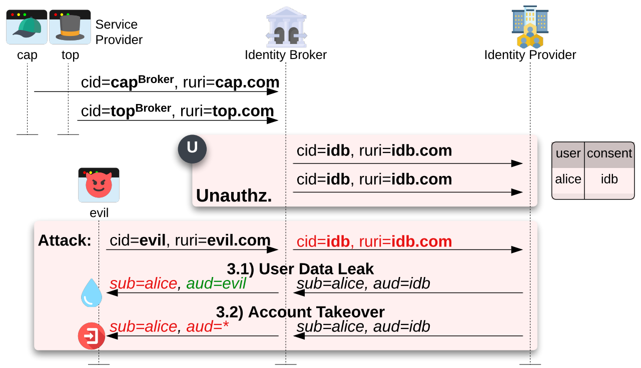


Figure 7: Unauthorized Data Access. **U** The user grants access to the broker and all its SPs, including malicious SPs that the user has never visited or given consent to. All SP↔broker CIDs are mapped to the same broker↔IdP CID.

provide consent to Skyscanner (SP) although it returns the token to Auth0 (broker). In this paper, we operate under the premise that SPs make well-informed choices to integrate only trusted brokers.

Figure 6: Running Example. We consider two honest SPs operating on cap.com and top.com, and one malicious SP on evil.com. All of them implement brokered SSO with the same broker and IdP. The SPs cap.com and top.com are trustworthy and the user has consented to share data with them. Both are operated by the same company, called "hats". The SP on evil.com is under the attacker's control. We assume that the user has *not* provided consent to the malicious SP.

Client Identifier. Initially, all three SP developers need to pre-register their SP at the broker. The broker assigns them a unique CID (e.g., `cap`$^{\text{Broker}}$, `top`$^{\text{Broker}}$, `evil`). For each CID, SP developers store the respective RURI on the allowlist (e.g., cap.com, top.com, evil.com). Finally, the integration between the broker and IdP must be configured.

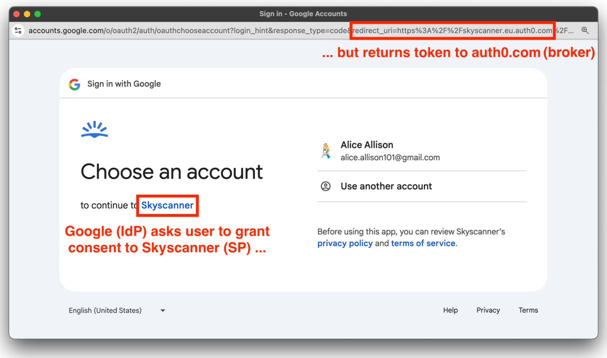
By analyzing our ground truth (cf. §3.1), we found three integration patterns: (1) Secure Data Access, (2) Federated Data Access, (3) Unauthorized Data Access.

S Secure Data Access. The broker provides a distinct tuple (CID, RURI) for each SP. The broker also uses distinct tuples for the communication with the IdP. As a result, the user gives consent only for the SP starting the authentication.

When the user initiates the SSO login on cap.com, the SP provides its CID and RURI to the broker (`cap`$^{\text{Broker}}$, cap.com). The broker then maps them to the corresponding CID and RURI at the IdP (`cap`$^{\text{IdP}}$, cap.idb.com). Thereafter, it redirects to the IdP. If the user has previously granted consent to `cap`, the IdP issues a signed token, which includes the user's identity (`subject` is Alice). Finally, the broker issues a new token that encapsulates the user's identity.

Takeaway 4: Secure Data Access. The most secure and privacy-friendly way of implementing brokered SSO is to let users grant consent to each SP individually.

F Federated Data Access. In the federated integration


Sign in - Google Accounts

accounts.google.com/o/oauth2/auth/oauthchooseaccount?login_hint&response_type=code&redirect_uri=https%3A%2F%2Fskyscanner.eu.auth0.com%2F...

... but returns token to auth0.com (broker)

Sign in with Google

Choose an account

to continue to Skyscanner

Google (IdP) asks user to grant consent to Skyscanner (SP) ...

Alice Allison
alice.allison101@gmail.com

Use another account

Before using this app, you can review Skyscanner's privacy policy and terms of service.

English (United States) Help Privacy Terms

Figure 8: Exemplary Consent Screen in Brokered SSO. Google (IdP) asks the user to provide consent to Skyscanner (SP) although it returns the token to Auth0 (broker).

pattern, the SPs `cap` and `top` belong to the same "hats" federation. The broker registers one CID at the IdP per federation. Since the SPs share a common CID at the IdP (`hats`), users who logged in to cap.com are not asked to grant consent when logging in to top.com. The federation model ensures security by blocking unauthorized access from SPs that are not part of the federation "hats". Real-world federations include Atlassian [9], which operates the SPs trello.com and bitbucket.com.

Takeaway 5: Shared Federated Data Access. All SPs within the federation receive the same data access rights.

U Unauthorized Data Access. In this pattern (cf. Figure 7), the *broker* must pre-register *itself* at the IdP by providing its single RURI (e.g., idb.com). The IdP assigns one CID to the broker (e.g., `idb`). Despite each SP integrating the broker independently, they *all* share the same CID at the IdP. Similarly to the federation pattern, users who have logged into cap.com are not required to provide their consent again when accessing top.com. However, malicious

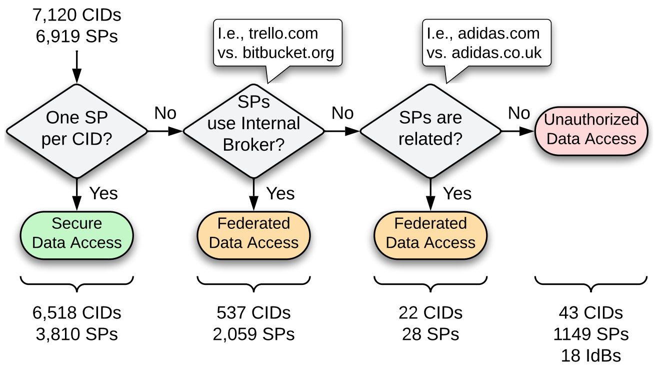


Figure 9: Our Automated Methodology to Measure Integration Patterns for Obtaining Consent in Brokered SSO.

SPs such as evil.com can now misuse the consent that the victim granted to trustworthy SPs.

User Data Leak (💧): Initially, the malicious SP must pre-register at the broker. The practical feasibility of this pre-registration is discussed in §7.3. We also assume the victim visits the malicious SP and holds an account with one of the trustworthy SP (`cap` or `top`). Upon visiting the malicious SP, the SSO login is initiated. With the prior consent given to the broker, the malicious SP promptly obtains a token scoped to their site. This token grants access to the victim's personal data, such as name, email, and address.

Escalation to Account Takeover (➔): In the prior case, the malicious SP receives a token scoped only to its site (`audience` is `evil`). This token allows the malicious SP to access the victim's data, but still protects the accounts at the trustworthy SPs. However, some brokers issue tokens without specific scoping (`audience` is anyone). When the malicious SP obtains such tokens, the attacker can compromise the victim's accounts at trustworthy SPs. It is important to note that in the secure data access pattern, even with an unrestricted token, the malicious SP must still obtain the user's consent, which is practically unlikely. Conversely, in the unauthorized data access pattern, the attacker has pre-existing consent, enabling immediate account takeover.

Takeaway 6: Unauthorized Data Access. This design flaw arises if the broker reuses the same CID for all its SPs. If the set of SPs includes a malicious one, it gains access to the victim's data. Some brokers expand the severity of this issue by issuing unscoped tokens.

7.3. Evaluation: Consent in Brokered SSO

Figure 9 presents our automated method for evaluating secure, federated, and unauthorized data access in brokered SSO. We began our analysis with 15,213 SSO buttons implementing a brokered SSO login (see Table 1). We excluded 319 SSO buttons for which we could not extract the CID. The analysis continued with the remaining 14,894 SSO buttons, linked to 6,919 different SPs. These SPs utilized a total of 7,120 unique CIDs across 10 IdPs, including the top three: Google (3,240 CIDs), Facebook (2,346 CIDs), and Apple (576 CIDs). We employed a three-step process to discern secure, federated, and unauthorized data access.

Step 1: "One SP per CID?" CIDs linked to a single SP are considered secure. We found 3,810 SPs linked to 6,518 CIDs. On average, each SP supports SSO with 1.7 IdPs.

Step 2: "Do SPs use an Internal Broker?" Internal brokers are part of the federated data access model because only a restricted set of SPs can integrate them. For instance, both trello.com and bitbucket.org utilize Atlassian's internal broker for SSO, creating a closed federation that prevents access from malicious actors. We identified 2,059 SPs linked to 537 CIDs, averaging to 3.8 SPs per federation.

Step 3: "Are SPs related?" SPs can still join a federation on a publicly accessible broker. We use the automated approach described in §3.3 to determine whether SPs are related. This method filters out 28 related SPs linked to 22 CIDs. For instance, thesalinepost.com and thesuntimesnews.com display different news articles but share the same TLS certificate from the same parent company.

Unauthorized Data Access. Our automated approach identified 18 brokers susceptible to unauthorized data access. These brokers connect 1,149 SPs to 43 CIDs. On average, 26.7 unrelated SPs share the same consent at the IdP.

Manual Verification: Methodology. Table 3 presents our manual verification of the vulnerabilities identified by our automated system. For freely accessible brokers, we pre-registered our own malicious SP. For brokers requiring a registration fee, we randomly selected two unrelated SPs for each broker, one deemed trustworthy and the other malicious. We assumed that the user would consent only to the trustworthy SP. brokers were considered to leak user data (💧) if the malicious SP gains access to it. brokers were deemed vulnerable to account takeover (➔) if the malicious SP obtains a token that is valid for the trustworthy SPs.

Manual Verification: Results. We manually verified that 15 of 18 brokers are leaking user data. Two brokers implemented an additional consent layer at the broker. We were unable to test one broker due to bugs in the SP implementation that prevented us from successfully logging in. On 6 brokers, we could escalate the issue further as they returned unscoped tokens to the malicious SP, allowing us to take over the victim's accounts at the trusted SPs.

Pre-Registration of Malicious SPs. Our manual investigation reveals that the majority of brokers present minimal obstacles for attackers trying to register malicious SPs. Specifically, 9 brokers are accessible without any cost, whereas 6 necessitate a registration fee that varies between $1.50 and $400. Only 3 brokers mandate engagement with their sales departments, potentially reducing, but not necessarily eliminating, the likelihood of malicious registrations.

		Vuln.		# Vuln. SPs per (Broker, IdP)				
Brokers	**$**	💧	➔]	f	G		in	⊞
CivicPlus	💬	●	●	531				
uLogin	🤲	●	○	188	130			16
Growave	🤲	●	●	102	107			
Hiko	🤲	●	●	46	42	16	8	7
Oxi	💵	●	●	43	48		10	
Okas	💵	●	○	35	38		14	
CONNECT	🤲	○	○		~~21~~	~~21~~		~~12~~
Login4Play	🤲	●	●	17	21			
miniOrange	💵	●	○		16		12	
Mylo.id	🤲	○	○	~~23~~	~~22~~	~~21~~		
NexusMedia EA	💵	●	○	9	7			
Salesforce CI	💬	●	○	6	7			
Zifyapp	🤲	●	○	5	6			
Froonze	🤲	●	○	4	2			
miniOrange SL	💵	●	○		2		2	
LiveRe	🤲	●	●	2				
Xsolla	💬	◐	◐		~~2~~			
Hexgator	💵	●	○	2				
18		15	6	→ 1,103 SPs with 💧 → 799 SPs with ➔]				

$ Registration Fees, 💧 User Data Leak, ➔] Account Takeover
🤲 Freely Available, 💵 Paid Plan Only, 💬 Price Upon Request
○ Secure, ● Vulnerable, ◐ Manual Verification Failed (Buggy SPs)

Table 3: Unauthorized Data Access in Brokered SSO. In total, 15 brokers fail to provide adequate protection for user data against unauthorized access. Attackers can access the victim's data if the victim holds an account with any of the 1,103 SPs. More alarmingly, attackers can escalate this issue on 6 brokers to take over the victim's account on 799 SPs.

Takeaway 7: We found 15 **brokers that fail to protect the user's data.** If a user logs in to one of 1,103 SPs integrating such a vulnerable broker, their data is at risk of being compromised. Additionally, attackers can escalate this issue on 6 brokers that issue unscoped tokens, allowing them to hijack the user's account on 799 SPs.

8. The Weakest Link in Brokered SSO

Over the years, extensive research has identified attacks in OAuth and OIDC, culminating in the publication of the official OAuth SBCPs [39]. This document provides a current framework for SSO implementors by outlining all known security threats and mitigation strategies. Our analysis is based on previously recorded data and can thus focus only on vulnerabilities detectable through passive frontend communication without requiring authentication [34, §6.2].

We examine five main threats, as shown in §8.1: (1) insecure flows, (2) secret leaks, (3) PKCE, (4) `state`, and (5) `nonce`. Note that not following these recommendations does not necessarily indicate a vulnerability but increases the risk of exploits. In §8.2, we reveal 3,367 violations in 5,668 SSO flows, in which the brokers actively downgrade the security by removing security-relevant parameters. In §8.3, we discover 7,415 violations, in which the brokers tolerate less secure SSO flows instead of enforcing secure ones.

		# Flows (Total: 5,668)			# Websites (Total: 2,780)		
		§8.2	§8.3	Secure	§8.2	§8.3	Secure
❶	Insecure Flows	0	1,034	4,634	0	545	2,235
❷	Secret Leaks	0	0	5,668	0	0	2,780
❸	Missing PKCE	2,954	2,698	16	1,505	1,273	2
	Invalid PKCE	0	89	5,579	0	41	2,739
	→ no `code`	0	17	5,651	0	9	2,771
	→ reuse	0	84	5,584	0	36	2,744
	→ entropy < 128 bit	0	35	5,633	0	13	2,767
	→ invalid method	0	39	5,629	0	17	2,763
❹	Missing `state`	7	1,021	4,640	7	543	2,230
	Invalid `state`	8	704	4,956	4	306	2,470
	→ reuse	8	697	4,963	4	300	2,476
	→ entropy < 128 bit	0	141	5,527	0	78	2,702
❺	Missing `nonce`	93	2	5,573	93	1	2,686
	Invalid `nonce`	305	1,866	3,557	263	858	1,659
	→ no `id_token`	304	1,786	3,578	262	846	1,672
	→ reuse	1	138	5,529	1	67	2,712
	→ entropy < 128 bit	0	25	5,643	0	18	2,762
	Σ **Total Violations**	3,367	7,415	–	1,872	3,568	–

Table 4: Violations of SBCPs. Brokers downgrade the adoption of PKCE and `nonce` protections, and they fall short of disallowing insecure flows and missing `states`.

8.1. Security Best Current Practices

❶ **Insecure Flows**. OAuth / OIDC defines three tokens:
(1) The `code` serves as a token reference and is redeemed in exchange for other tokens. A secret secures this exchange and prevents illegitimate misuse of leaked `codes`.
(2) The `id_token` includes digitally signed claims about the user's identity. Should it be compromised, the `nonce` protection prevents its misuse (see below).
(3) The `access_token` grants direct access to the user's resources. If leaked, no further security mechanisms exist to prevent its illegitimate misuse. Since 2018, the SBCPs [39, §2.1.2] have banned exposing `access_tokens` to browsers, recognizing their high risk of leakage [17, 72]. Today, `codes` are uniformly recommended [13, 14, 39].

❷ **Secret Leaks**. The secret is established during the pre-registration, see Step 1 in Figure 2. The secret protects leaked `codes` from illegitimate misuse by attackers and thus should not be revealed to untrusted parties.

❸ **Proof Key for Code Exchange**. PKCE protects against the unauthorized use of exposed `codes` by malicious actors. It ties the `code` to a single-use, random identifier with over 128-bit entropy [54, §7.1], which undergoes hashing.

❹ **`state`**. The `state` protects against CSRF attacks (see §2). Like PKCE, it is designed to be a single-use, high-entropy [32], and random identifier.

❺ **`nonce`**. The `nonce` safeguards against CSRF and prevents illegitimate use of `id_tokens` upon their exposure. Similar to the `state`, it serves as a single-use, high-entropy [32], and random identifier that is also incorporated within the signed `id_token`.

8.2. Security Downgrades

Brokers are used to facilitate the integration of SSO services. However, a broker should not decrease the security compared to three-actors SSO. Unfortunately, we discovered 1,872 violations on 2,780 websites where brokers are actively downgrading security, see Table 4. To our surprise, brokers commonly downgrade PKCE, which is a strong security mechanism preventing many attacks, including CSRF and token leakage. We found that in 2,954 flows, the SP initiates PKCE, but the broker removes it when communicating with the IdP, thereby actively reducing the security level. We discovered similar, although less prevalent issues with the `state` and `nonce` parameters. In summary, we observed 3,367 violations downgrading the flows.

8.3. Tolerating SBCPs Violations

❶ **Insecure Flows**. In Table 4, we show that none of the brokers are using insecure flows. But all are tolerating insecure flows started by the SP instead of preventing them.

❸ **PKCE**. Besides deploying PKCE, we observed its invalid use in scenarios such as flows not using any `code`, reuse of the same identifiers, insufficient entropy, or the use of invalid hashing algorithms. Overall, the adoption of PKCE remains low where either the broker or the SP do not support it. Only in 16 flows, PKCE is used during the entire authentication. This demonstrates how the hope of SPs to delegate this complex implementation to more familiar developers is misplaced endangering all unaware users.

❹ **`state`**. Although CSRF attacks at the IdP are largely mitigated, they remain a considerable challenge at the broker. For instance, the three most frequently ineffective `state` identifiers accepted or generated by brokers include: (1) 52x empty string, (2) 37x `STATE`, and (3) 31x `OD0w`.

❺ **`nonce`**. Similar issues are observed with respect to the `nonce` parameter. For instance, the three most frequent values include: (1) 11x empty string, (2) 8x `defaultNonce`, and (3) 6x `rA3H[...]HBM=` (static base64-encoded string).

Takeaway 8: Analyzing the adoption of the SBCPs [39], we observe a shift of responsibility from the IdP to the new entity of the SSO flow, the broker. Regarding the adoption of SBCPs, is it possible to see that the brokers keep the status quo regarding existing security risks [34, 48]. Even more alarming, brokers actively decrease security by removing security-relevant parameters.

9. Related Work

In this section, we review prior work on the security of SSO and highlight how this paper differs / advances them.

Three-Actors vs. Brokered SSO. In recent years, it has become increasingly clear that users heavily rely on SSO services to authenticate on websites. This reliance has stimulated the research community to conduct numerous studies on the security of SSO, including analyses of the protocols [19, 20, 21, 47] and implementations [6, 24, 25, 31, 33, 34, 38, 40, 41, 44, 48, 51, 64, 67, 72].

However, prior work has focused only on the three-actors SSO flow, involving users, SPs, and IdPs. Interestingly, in 2022, Jannett et al. [33] and Ghasemisharif et al. [24] observed that multiple websites are susceptible to the same SSO vulnerabilities. For instance, Jannett et al. identified 11 different websites susceptible to the same In-Browser Communication (InBC) vulnerability [33, Table 3]. Ghasemisharif et al. found that websites are reusing the same CID at the IdP [24, Figure 9]. However, they did not elaborate on the root causes or implications of this behavior.

In this paper, we fill this gap by uncovering the reasons behind their observations, presenting a new ecosystem of brokered SSO that has not yet been studied in previous research. We suggest that in brokered SSO, the same vulnerability at the broker affects multiple different websites.

Security Best Current Practice. The IETF and research community [31, 33, 39, 40, 58, 59] strive to continually update the SBCPs with mitigation strategies to safeguard SPs and IdPs from known threats, thus protecting users from emerging attack vectors. Recent studies have shown that many SPs and IdPs in three-actors SSO do *not* comply with these SBCPs [34, 48], which is alarming.

In this paper, we extend prior work by including brokers into the assessment of SBCP compliance. In specific, we compare the compliance of flows between SPs↔brokers and brokers↔IdPs. Our findings suggest that flows between SPs↔brokers are weaker compared to brokers↔IdPs. Even worse, brokers actively downgrade the security of SSO flows that would have been classified as protected by prior work.

Open Redirects. ORs [66] are a well-known web security threat that has for years been considered a low-risk vulnerability and out-of-scope in bug bounty programs [37]. Websites with ORs redirect users to the attacker's websites, with the impact typically limited to phishing. However, ORs have proven to be the vulnerable heel of SSO, as they can leak authentication tokens to the attacker's website. The SBCPs[39] and research community[31, 42, 45, 52] have addressed this emerging attack surface, finding that IdPs only *partly* prevent them [1, 31, 42, 45, 52]. We extend prior work and show that over 20% of brokers do not prevent ORs *at all*, which is even more alarming.

10. Concluding Remarks

This paper is the first to shed light on the brokered SSO ecosystem. Brokered SSO introduces an additional entity into the SSO login flow, called broker. Brokers provide identity services and significantly offload the burden of implementing SSO. However, we empirically show that using brokers is not a panacea without consequences, and their adoption should be considered carefully. We hope that our research will start a broader discussion in the IETF and research community regarding the security and privacy implications of including a new actor in the SSO flow.

Surprising Results. In this paper, we demonstrate that blind spots in SSO research still exist. Even recent work by Jannett et al. [34], which systemizes all scanning techniques and provides a comprehensive evaluation, fails to fully analyze 25% of the SSO landscape. The significance of these gaps in security is evident as we reveal vulnerabilities in over 50 brokers, affecting thousands of websites. Regarding brokers, we are also surprised to find that security is often weakened by implementing new, unknown bad patterns, allowing malicious actors to access user data without consent, or by violating SBCPs.

Security Best Current Practices. As a direct consequence of our findings, we will initiate contact with the IETF and request updates to the SBCPs, focusing on brokered SSO flows. Among other issues, the identified flaws in designing the redirection chain (see §7) should be addressed by requiring users to grant consent to each SP individually.

Non-Compliant SSO. We discovered two promising research areas that should be further explored. Non-compliant brokered SSO flows present a significant blind spot in current security evaluations. While some of these blind spots have been addressed in [33] and this research, our findings reveal new gaps that warrant further investigation. Specifically, we were only able to successfully recover 3,926 out of 15,213 flows (see §6). This indicates that a substantial portion of SSO traffic remains unaccounted, likely due to custom parameters and non-compliant behaviors that standard evaluation methods fail to address.

Future research should focus on developing robust strategies to handle these custom parameters and non-compliant behaviors more effectively. This could involve the creation of adaptive algorithms that can dynamically recognize and adjust to varying SSO implementations. Additionally, machine learning techniques could be leveraged to identify patterns and anomalies within non-compliant flows, enhancing the detection and mitigation of potential security risks.

Clustering Approach. Our clustering approach has shown great promise in filtering unknown traffic and highlighting similarities, which is crucial for analyzing less-documented protocols such as brokered payment services. Prominent examples of such services include Arvato, Stripe, Payone, GiroSolution, and Klarna. By applying our clustering methodology, we can systematically organize and analyze traffic by focusing on the most relevant messages, thereby improving our understanding of these protocols.

Future work should aim to refine and expand this clustering approach to cover a broader range of protocols and services. This could involve integrating more sophisticated clustering algorithms and enhancing the granularity of the analysis to capture subtle differences in traffic patterns. Additionally, applying this method to real-world datasets will uncover new insights into the behavior of brokered payment services and other similar protocols.

Acknowledgments

The authors would like to 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. This work was partially-funded by the NSF project number 2031390 and NSF project number 2329540.

References

[1] *(Web-)Insecurity Blog — Insufficient Redirect URI validation: The risk of allowing to dynamically add arbitrary query parameters and fragments to the redirect_uri*. https://security.lauritz-holtmann.de/post/sso-security-redirect-uri-ii/. (Accessed on 06/07/2024).

[2] *Adobe Commerce Extensions & Themes for Your Online Store — Marketplace*. https://commercemarketplace.adobe.com/. (Accessed on 04/11/2024).

[3] Devdatta Akhawe et al. "Towards a Formal Foundation of Web Security". In: *2010 23rd IEEE Computer Security Foundations Symposium*. 2010 IEEE 23rd Computer Security Foundations Symposium (CSF). Edinburgh, United Kingdom: IEEE, July 2010, pp. 290–304. ISBN: 978-1-4244-7510-0. DOI: 10.1109/CSF.2010.27. URL: http://ieeexplore.ieee.org/document/5552637/ (visited on 09/28/2021).

[4] Assel Aliyeva and Manuel Egele. "Oversharing Is Not Caring: How CNAME Cloaking Can Expose Your Session Cookies". In: *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*. ASIA CCS '21. Virtual Event, Hong Kong: Association for Computing Machinery, 2021, pp. 123–134. ISBN: 9781450382878. DOI: 10.1145/3433210.3437524. URL: https://doi.org/10.1145/3433210.3437524.

[5] Amazon AWS. *Amazon Cognito*. URL: https://aws.amazon.com/en/cognito/.

[6] Michele Benolli et al. "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 et al. Vol. 12756. Series Title: Lecture Notes in Computer Science. 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 (visited on 10/21/2023).

[7] *Build Shopify Apps*. https://shopify.dev/docs/apps. (Accessed on 04/11/2024).

[8] *CivicPlus — THE Modern Civic Experience Platform*. https://www.civicplus.com/. (Accessed on 04/11/2024).

[9] *Collaboration software for software, IT and business teams*. https://www.atlassian.com/. (Accessed on 05/13/2024).

[10] *Configure Silent Authentication*. https://auth0.com/docs/authenticate/login/configure-silent-authentication#renew-expired-tokens. (Accessed on 04/08/2024).

[11] *Cross-Origin Authentication*. https://auth0.com/docs/authenticate/login/cross-origin-authentication#limitations. (Accessed on 04/08/2024).

[12] *Dangerous javascript: pseudo protocol — Beyond XSS*. https://aszx87410.github.io/beyond-xss/en/ch1/javascript-protocol/. (Accessed on 06/05/2024).

[13] *draft-ietf-oauth-browser-based-apps-18*. https://datatracker.ietf.org/doc/html/draft-ietf-oauth-browser-based-apps. (Accessed on 05/20/2024).

[14] *draft-ietf-oauth-v2-1-10*. https://datatracker.ietf.org/doc/html/draft-ietf-oauth-v2-1-10. (Accessed on 05/20/2024).

[15] 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*. CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security. Virtual Event USA: ACM, Oct. 30, 2020, pp. 1953–1970. ISBN: 978-1-4503-7089-9. DOI: 10.1145/3372297.3417869. URL: https://dl.acm.org/doi/10.1145/3372297.3417869 (visited on 07/21/2021).

[16] *Facebook Login*. https://developers.facebook.com/docs/facebook-login/. (Accessed on 03/20/2024).

[17] Shehroze Farooqi et al. "Measuring and Mitigating OAuth Access Token Abuse by Collusion Networks". In: *Proceedings of the 2017 Internet Measurement Conference*. IMC '17. New York, NY, USA: Association for Computing Machinery, Nov. 1, 2017, pp. 355–368. ISBN: 978-1-4503-5118-8. DOI: 10.1145/3131365.3131404. URL: https://doi.org/10.1145/3131365.3131404 (visited on 06/15/2021).

[18] *FedCM/explainer.md at main · fedidcg/FedCM*. https://github.com/fedidcg/FedCM/blob/main/explainer.md#what-will-break. (Accessed on 04/08/2024).

[19] Daniel Fett, Pedram Hosseyni, and Ralf Küsters. "An Extensive Formal Security Analysis of the OpenID Financial-Grade API". In: *2019 IEEE Symposium on Security and Privacy (SP)*. 2019 IEEE Symposium on Security and Privacy (SP). ISSN: 2375-1207. May 2019, pp. 453–471. DOI: 10.1109/SP.2019.00067. URL: https://ieeexplore.ieee.org/abstract/document/8835218?casa_token=51CrF3vT03UAAAAA:aHO9CZL8Qdk_rvzoKYuRE3_2rphOv6X4pQz-Q_VPFE4rivkxr2NN65Up37dIoeNuhyDWqWAyXg (visited on 10/22/2023).

[20] Daniel Fett, Ralf Küsters, and Guido Schmitz. "A Comprehensive Formal Security Analysis of OAuth 2.0". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 2016. URL: https://dl.acm.org/doi/pdf/10.1145/2976749.2978385 (visited on 09/03/2020).

[21] 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.

[22] *Free Whois Lookup - Whois IP Search & Whois Domain Lookup — Whois.com*. https://www.whois.com/whois/. (Accessed on 05/23/2024).

[23] *General Data Protection Regulation (GDPR) – Legal Text*. https://gdpr-info.eu/. (Accessed on 05/12/2024).

[24] 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)*. ISSN: 2375-1207. Los Alamitos, CA, USA: IEEE Computer Society, May 2022, pp. 1524–1524. DOI: 10.1109/SP46214.2022.00095. URL: https://doi.ieeecomputersociety.org/10.1109/SP46214.2022.00095.

[25] Mohammad Ghasemisharif et al. "O Single Sign-Off, Where Art Thou? An Empirical Analysis of Single Sign-On Account Hijacking and Session Management on the Web". In: *27th USENIX security symposium (USENIX security 18)*. Baltimore, MD: USENIX Association, Aug. 2018, pp. 1475–1492. ISBN: 978-1-939133-04-5. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/ghasemisharif.

[26] Google. *Google Firebase*. URL: https://firebase.google.com/.

[27] *HAR (file format) - Wikipedia*. https://en.wikipedia.org/wiki/HAR_(file_format). (Accessed on 05/23/2024).

[28] Dick Hardt. *The OAuth 2.0 Authorization Framework*. RFC 6749. Oct. 2012. DOI: 10.17487/RFC6749. URL: https://www.rfc-editor.org/info/rfc6749.

[29] Dick Hardt, Aaron Parecki, and Torsten Lodderstedt. *The OAuth 2.1 Authorization Framework*. Internet-Draft draft-ietf-oauth-v2-1-11. Work in Progress. Internet Engineering Task Force, May 2024. 96 pp. URL: https://datatracker.ietf.org/doc/draft-ietf-oauth-v2-1/11/.

[30] *Identity and Access Management - statistics*. https://www.statista.com/topics/10552/identity-and-access-management. (Accessed on 05/3/2024).

[31] Tommaso Innocenti et al. "OAuth 2.0 Redirect URI Validation Falls Short, Literally". In: *Annual Computer Security Applications Conference (ACSAC)*. Dec. 2023. DOI: https://doi.org/10.1145/3627106.3627140.

[32] *Insufficient Session-ID Length — OWASP Foundation*. https://owasp.org/www-community/vulnerabilities/Insufficient_Session-ID_Length. (Accessed on 05/21/2024).

[33] Louis Jannett et al. "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.

[34] Louis Jannett et al. "SoK: SSO-MONITOR — The Current State and Future Research Directions in Single Sign-On Security Measurements". In: *Proceedings of the 2024 European Symposium on Security

and Privacy (EuroSP). 2024. URL: https://sso-monitor.me/paper.pdf.

[35] Oscar Jarpehult et al. “A Longitudinal Characterization of the Third-Party Authentication Landscape”. In: *2022 IFIP Networking Conference (IFIP Networking)*. 2022 IFIP Networking Conference (IFIP Networking). Catania, Italy: IEEE, June 13, 2022, pp. 1–9. ISBN: 978-3-903176-48-5. DOI: 10.23919/IFIPNetworking55013.2022.9829804. URL: https://ieeexplore.ieee.org/document/9829804/ (visited on 07/10/2023).

[36] *Keycloak*. https://www.keycloak.org/. (Accessed on 04/08/2024).

[37] *Learn about Open Url Redirects — BugBountyHunter.com*. https://www.bugbountyhunter.com/vulnerability/?type=open_redirect. (Accessed on 06/07/2024).

[38] Guannan Liu, Xing Gao, and Haining Wang. “An Investigation of Identity-Account Inconsistency in Single Sign-On”. In: *Proceedings of the Web Conference 2021*. WWW ’21: The Web Conference 2021. Ljubljana Slovenia: ACM, Apr. 19, 2021, pp. 105–117. ISBN: 978-1-4503-8312-7. DOI: 10.1145/3442381.3450085. URL: https://dl.acm.org/doi/10.1145/3442381.3450085 (visited on 09/20/2021).

[39] Torsten Lodderstedt et al. *OAuth 2.0 Security Best Current Practice*. Internet-Draft draft-ietf-oauth-security-topics-25. Work in Progress. Internet Engineering Task Force, Feb. 2024. 59 pp. URL: https://datatracker.ietf.org/doc/draft-ietf-oauth-security-topics/25/.

[40] Christian Mainka, Vladislav Mladenov, and Jörg Schwenk. “Do not trust me: Using malicious IdPs for analyzing and attacking Single Sign-On”. In: *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 321–336.

[41] Christian Mainka et al. “SoK: Single Sign-On Security – An Evaluation of OpenID Connect”. In: *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2017, pp. 251–266.

[42] *Make Redirection Evil Again: URL Parser Issues in OAuth*. https://i.blackhat.com/asia-19/Fri-March-29/bh-asia-Wang-Make-Redirection-Evil-Again-wp.pdf. (Accessed on 05/20/2024).

[43] *Microsoft Entra ID (formerly Azure Active Directory) — Microsoft Security*. https://www.microsoft.com/en-us/security/business/identity-access/microsoft-entra-id. (Accessed on 04/08/2024).

[44] 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). (Visited on 09/01/2020).

[45] OAuth 2.0. *OAuth Security Advisory: 2014.1 ”Covert Redirect”*. https://oauth.net/advisories/2014-1-covert-redirect/. 2014.

[46] *OAuth Security Workshop - OSW 2025*. https://oauth.secworkshop.events/osw2025. (Accessed on 10/01/2024).

[47] Suhas Pai et al. “Formal Verification of OAuth 2.0 Using Alloy Framework”. In: *2011 International Conference on Communication Systems and Network Technologies*. 2011 International Conference on Communication Systems and Network Technologies (CSNT). Katra, Jammu, India: IEEE, June 2011, pp. 655–659. ISBN: 978-1-4577-0543-4. DOI: 10.1109/CSNT.2011.141. URL: http://ieeexplore.ieee.org/document/5966531/ (visited on 10/22/2023).

[48] Pieter Philippaerts, Davy Preuveneers, and Wouter Joosen. “OAuch: Exploring Security Compliance in the OAuth 2.0 Ecosystem”. In: *25th International Symposium on Research in Attacks, Intrusions and Defenses*. RAID 2022: 25th International Symposium on Research in Attacks, Intrusions and Defenses. Limassol Cyprus: ACM, Oct. 26, 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).

[49] *Prepare for phasing out third-party cookies — Privacy Sandbox — Google for Developers*. https://developers.google.com/privacy-sandbox/3pcd. (Accessed on 04/05/2024).

[50] *PrestaShop Addons Marketplace - Modules, Themes & Support*. https://addons.prestashop.com/en/. (Accessed on 04/11/2024).

[51] Tamjid Al Rahat, Yu Feng, and Yuan Tian. “Cerberus: Query-driven Scalable Vulnerability Detection in OAuth Service Provider Implementations”. 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. 7, 2022, pp. 2459–2473. ISBN: 978-1-4503-9450-5. DOI: 10.1145/3548606.3559381. URL: https://dl.acm.org/doi/10.1145/3548606.3559381 (visited on 10/21/2023).

[52] Frans Rosén. *Account hijacking using ”dirty dancing” in sign-in OAuth-flows*. https://labs.detectify.com/2022/07/06/account-hijacking-using-dirty-dancing-in-sign-in-oauth-flows/. 2022.

[53] N. Sakimura et al. *OpenID Connect Core 1.0 incorporating errata set 2*. Dec. 15, 2023. URL: https://openid.net/specs/openid-connect-core-1_0.html (visited on 03/20/2024).

[54] Nat Sakimura, John Bradley, and Naveen Agarwal. *Proof Key for Code Exchange by OAuth Public Clients*. RFC 7636. Sept. 2015. DOI: 10.17487/RFC7636. URL: https://www.rfc-editor.org/info/rfc7636.

[55] *Saying goodbye to third-party cookies in 2024 — MDN Blog*. https://developer.mozilla.org/en-US/blog/goodbye-third-party-cookies/#how_browsers_have_responded_to_this. (Accessed on 04/08/2024).

[56] Jörg Schwenk, Marcus Niemietz, and Christian Mainka. “Same-Origin Policy: Evaluation in Modern Browsers”. In: *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX

Association, Aug. 2017, pp. 713–727. ISBN: 978-1-931971-40-9. URL: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/schwenk.

[57] *Search*. https://github.com/search. (Accessed on 05/23/2024).

[58] Karsten Meyer zu Selhausen and Daniel Fett. *OAuth 2.0 Authorization Server Issuer Identification*. RFC 9207. Mar. 2022. DOI: 10.17487/RFC9207. URL: https://www.rfc-editor.org/info/rfc9207.

[59] Karsten Meyer zu Selhausen, Louis Jannett, and Christian Mainka. *OAuth 2.0 Web Message Response Mode for Popup- and Iframe-based Authorization Flows*. Internet-Draft draft-meyerzuselha-oauth-web-message-response-mode-00. Work in Progress. Internet Engineering Task Force, Nov. 2023. 12 pp. URL: https://datatracker.ietf.org/doc/draft-meyerzuselha-oauth-web-message-response-mode/00/.

[60] Asuman Senol et al. "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.

[61] Ethan Shernan et al. "More Guidelines Than Rules: CSRF Vulnerabilities from Noncompliant OAuth 2.0 Implementations". In: *Detection of Intrusions and Malware, and Vulnerability Assessment*. Ed. by Magnus Almgren, Vincenzo Gulisano, and Federico Maggi. Vol. 9148. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 239–260. DOI: 10.1007/978-3-319-20550-2_13. URL: http://link.springer.com/10.1007/978-3-319-20550-2_13 (visited on 09/03/2020).

[62] *Sign in with Apple JS — Apple Developer Documentation*. https://developer.apple.com/documentation/sign_in_with_apple/sign_in_with_apple_js. (Accessed on 03/24/2024).

[63] Marius Steffens and Ben Stock. "PMForce: Systematically Analyzing postMessage Handlers at Scale". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. CCS '20. Number of pages: 13 Place: Virtual Event, USA. New York, NY, USA: Association for Computing Machinery, 2020, pp. 493–505. ISBN: 978-1-4503-7089-9. DOI: 10.1145/3372297.3417267. URL: https://doi.org/10.1145/3372297.3417267.

[64] Avinash Sudhodanan and Andrew Paverd. "Pre-hijacked accounts: An Empirical Study of Security Failures in User Account Creation on the Web". In: *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 1795–1812. ISBN: 978-1-939133-31-1. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/sudhodanan.

[65] *tracker-radar-detector/src/entities/README.md at main · duckduckgo/tracker-radar-detector*. https://github.com/duckduckgo/tracker-radar-detector/blob/main/src/entities/README.md. (Accessed on 05/19/2024).

[66] *Unvalidated Redirects and Forwards - OWASP Cheat Sheet Series*. https://cheatsheetseries.owasp.org/cheatsheets/Unvalidated_Redirects_and_Forwards_Cheat_Sheet.html. (Accessed on 05/20/2024).

[67] 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: *2012 IEEE Symposium on Security and Privacy*. 2012 IEEE Symposium on Security and Privacy (SP) Conference dates subject to change. San Francisco, CA, USA: IEEE, May 2012, pp. 365–379. DOI: 10.1109/SP.2012.30. URL: http://ieeexplore.ieee.org/document/6234424/ (visited on 11/11/2021).

[68] *What is an Atlassian account? — Atlassian Support*. https://support.atlassian.com/atlassian-account/docs/what-is-an-atlassian-account/. (Accessed on 04/09/2024).

[69] *Window: postMessage() method - Web APIs — MDN*. https://developer.mozilla.org/en-US/docs/Web/API/Window/postMessage. (Accessed on 03/22/2024).

[70] *WooCommerce Mobile App – Run your store from anywhere*. https://woocommerce.com/de/mobile/. (Accessed on 04/11/2024).

[71] *WordPress Plugins — WordPress.org*. https://wordpress.org/plugins/. (Accessed on 04/11/2024).

[72] Yuchen Zhou and David Evans. "SSOScan: Automated Testing of Web Applications for Single Sign-On Vulnerabilities". In: *Proceedings of the 23rd USENIX Security Symposium*. San Diego, CA, Aug. 2014. ISBN: 978-1-931971-15-7.

Appendix A. Erroneous SSO Flows

Exclusion Reason	# Flows
Missing HAR file	5,225
Missing `redirect_uri` in login request of IdP	1,751
Multiple IdP login requests in single SSO flow	289
Login page not associated with Tranco domain	217
Invalid `redirect_uri` in login request of IdP	19
Σ	7,501

Table 5: Erroneous SSO Flows. In total, we had to exclude 7,501 erroneous SSO flows from the dataset provided by SSO-MONITOR [34]. We identified five reasons that prevented us from further analyzing the SSO flows.

Appendix B. Responsible Disclosure

We have responsibly disclosed the vulnerabilities to all affected brokers. In total, we have contacted all 55 brokers with insufficient validation of redirect chains (see §6, Table 2, and Table 6) or flawed consent patterns (see §7 and Table 3). At the time of publication, all brokers will have had at least 6 months to fix them.

Contact Points. For each of the 55 vulnerable brokers, we used a search engine and the broker's website to manually determine the following contact points (sorted based on priority): (1) security.txt file, (2) bug bounty programs (HackerOne, Bugcrowd, Intigriti), (3) vulnerability disclosure instructions, (4) email of data protection officer, (5) email of webmaster / administrator, and (6) other general contact points (i.e., support emails, online forms, etc.). For each broker, we manually identified one of the following contact points: 28x general email addresses (i.e., support@broker.com), 14x general online forms (i.e., support online forms), 8x dedicated emails for vulnerability disclosure (i.e., security@broker.com), 3x dedicated online forms for vulnerability disclosure, and 2x HackerOne.

Notifications. For all general-purpose contact points, we sent initial contact messages asking brokers to provide a secure communication channel for our vulnerability disclosure. We included details about our institutions and sent messages from our institutional email address to verify our credibility. If we received a positive response, we disclosed the full details of the vulnerability (including steps to reproduce) to the appropriate contact point. For all contact points dedicated to vulnerability disclosure, we directly disclosed the full details, skipping the initial contact message. We sent a second reminder if we did not receive a response after a month. We provide the full text of our initial and disclosure messages in our artifacts.

Unresponsive Brokers. If we do not receive a response to the second reminder, we will contact our national CERT and ask them to establish a communication channel with the brokers. We will discuss with them how we should proceed.

Current Status (as of October 1st, 2024). 2 brokers have already fixed the vulnerability. 4 brokers have confirmed the vulnerability and are working on a fix. 1 HackerOne report was closed as a duplicate. 11 brokers are currently investigating the report. 30 brokers have not yet responded to our initial contact attempt conducted in mid-July. We started a second contact attempt on August 21, 2024. 7 brokers were not reachable (i.e., "Undelivered Mail Returned to Sender"). Interestingly, all brokers that possess designated contact points for the communication of vulnerability disclosures have responded. We thankfully received $750 bug bounties.

Enhancing the Specifications. We are actively working on extending the OAuth SBCPs [39] by adding a section with our findings. In an effort to improve future protocols, we are working towards a contribution to the OAuth 2.1 draft [29]. Once we submit the proposed changes, we will also contact the IETF OAuth Working Group to discuss them. Furthermore, we plan to submit a presentation proposal for the OAuth Security Workshop [46]. This workshop is explicitly targeted at SSO implementors, and we hope to reach a broad audience of brokers.

# Vulnerable		Vulnerabilities		Impact	
Brokers	# SPs	OR	XSS	Leak	AT
CivicPlus	538	●	–	■	✓
TP	282	●	–	▣	✓
uLogin	235	–	○	▣	✓
Vox Media	221	●	–	⊟	✗
Gigya	153	–	○	▣	✓
Janrain	149	●	–	▣	✓
uID.me	108	●	●	▣	✓
Super Socializer	87	●	○	⊟	✗
Oxi	68	●	●	■	✓
LoginRadius	64	–	○	▣	✓
HIKO	60	●	●	▣	✓
Zephr	58	–	○	▣	✓
BLOX Digital	54	●	–	⊟	✗
Okas	52	–	○	▣	✓
Login4Play	26	●	○	▣	✓
CONNECT	22	●	–	⊟	✗
miniOrange	17	●	●	👤	✗
Zendesk	16	●	–	⊟	✗
CAS	16	●	–	▣	✓
YITH	15	●	–	⊟	✗
Schoolwires	14	●	–	■	✓
Rdeskbw	13	●	–	▣	✓
Sandhills Global	11	●	○	▣	✓
NexusMedia EasyAuth	11	○	●	⊟	✗
Magic	11	●	–	▣	✓
Prisa	10	●	–	▣	✓
Amadeus	9	●	–	▣	✓
Ymcart	9	●	–	▣	✓
Mediacorp	7	●	●	▣	✓
Scribbr	7	●	–	▣	✓
Heateor Social Login	7	●	○	⊟	✗
Froonze	6	●	–	▣	✓
OmniAuth	6	–	○	■	✓
Vaave	6	●	–	■	✓
Cyberbiz	5	●	–	▣	✓
Izea	5	●	○	▣	✓
Shoplazza	5	●	–	👤	✗
Wordpress Connect	4	●	–	⊟	✗
Webmercs	4	●	○	▣	✗
LiveRe	4	●	–	▣	✓
miniOrange Social Login	4	●	●	▣	✓
Alumnforce	3	●	–	■	✓
Social Login	3	●	–	▣	✓
Loghy	3	●	–	▣	✓
FNP	3	●	–	⊟	✗
StoreHippo	3	●	–	▣	✓
WirKaufenDeinAuto	3	●	–	▣	✗
Prepr	2	●	–	⊟	✗
E-Prosveta	2	●	–	▣	✓
49	2,421	42	7	6x ■,30x ▣ 2x 👤, 11x ⊟ 34x ✓, 15x ✗	

● Vulnerable, ○ Secure, – Not Applicable
Leak: ■ IdP Token, ▣ Custom Token, 👤 User Data, ⊟ No Data
AT: ✓ Confirmed Account Takeover, ✗ No Account Takeover
Vuln.: Open Redirect (OR), Cross-Site Scripting (XSS)

Table 6: Insufficient Validation of Redirection Chains in brokered SSO. We identified 49 brokers used by 2,421 SPs that fail to validate the redirection chain, leading to XSS, user data leakage, and account takeovers.

Appendix C.
Meta-Review

The following meta-review was prepared by the program committee for the 2025 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

C.1. Summary

This paper focuses on the security of brokered SSO. The authors implement an automated tool called IDB-Detector to detect the brokers. The paper proposes classification criteria to systemize all the brokers on the web and analyze the broker SSO ecosystem. Based on the characteristics of broker SSO, the paper identifies three categories of threats associated with it: (1) insufficient validation of redirect chains enabling injection attacks, (2) unauthorized data access enabling account takeovers, and (3) violations of security best current practices.

C.2. Scientific Contributions

- Creates a New Tool to Enable Future Science
- Identifies an Impactful Vulnerability
- Provides a Valuable Step Forward in an Established Field
- Establishes a New Research Direction

C.3. Reasons for Acceptance

(1) Paper identifies a new broker ecosystem in SSO that deserves more investigation and attention from the security community to identify flaws and fix them.

(2) Paper identifies vulnerabilities in this space and authors are in the process of notifying and alerting affected parties. This has led/is leading to real impact with existing brokers in the SSO ecosystem.