Contribution Title*

First Author[1][0000−1111−2222−3333], Second Author[2,3][1111−2222−3333−4444], and Third Author[3][2222−−3333−4444−5555]

[1] Princeton University, Princeton NJ 08544, USA
[2] Springer Heidelberg, Tiergartenstr. 17, 69121 Heidelberg, Germany
lncs@springer.com
http://www.springer.com/gp/computer-science/lncs
[3] ABC Institute, Rupert-Karls-University Heidelberg, Heidelberg, Germany
{abc,lncs}@uni-heidelberg.de

Abstract. The abstract should briefly summarize the contents of the paper in 150–250 words.

Keywords: First keyword · Second keyword · Another keyword.

1 Introduction

2 Introduction

As a widely deployed identity management and authentication mechanism in the current Internet, single sign-on (SSO) systems such as OpenID Connect [**?**], OAuth [**?**] and SAML [**?**] allow a user to log in to a website, called the *relying party* (RP), using the account registered at another website, called the *identity provider* (IdP). The RPs delegate user authentication to a trusted IdP, who generates *identity proofs* for her visits to these RPs. Thus, the user only needs to remember one credential for the IdP, instead of maintaining different credentials for different RPs. SSO has been widely integrated with many application services. For example, we find that 80% of the Alexa Top-100 websites support SSO [**?**], and the analysis on the Alexa Top-1M websites identifies 6.30% with the SSO support [**?**]. Meanwhile, many email and social network providers (such as Google, Facebook, Twitter, etc.) are serving the IdP roles in the Internet.

However, SSO systems have been continuously found vulnerable and insecure [**?**,**?**,**?**,**?**,**?**,**?**,**?**,**?**,**?**,**?**]. Moreover, the adoption of SSO raises a public concern about user privacy [**?**,**?**,**?**,**?**], that is whether an adversary is able to track to which RP(s) the user has logged in. Unfortunately, almost all the existing SSO protocols leak user privacy in different ways. Take a widely used SSO protocol OpenID Connect (OIDC) as an example. As shown in Fig. **??**, the login process starts when a user sends a login request to the RP, who then constructs a request for identity proof with its identity and redirects the request to the IdP. After authenticating the user, the IdP generates an identify proof with the

* Supported by organization x.

user's and RP's identities, which is returned to the user and forwarded to the RP. Finally, the RP verifies the identity proof to decide if the user is allowed to log in. In such login instances, by design, an IdP can always see when and where its users log in, in order to generate the identity proof. As a result, a curious IdP can always discover the RPs that a target user has visited over time. This data can be further analyzed to profile users' online activities. Thus, we call this privacy attack ***IdP-based login tracing***, which has also been reported by previous research [**?**,**?**]. Similarly, by design, the RPs can learn users' identities from the identify proofs. If the IdP binds an unique or relevant user identifier(s) to identity proofs generated for the same user but different RPs [**?**,**?**], these RPs can collude to correlate the identifier(s) with the user's identity. We denote this privacy risk as ***RP-based identity linkage***, which allows the adversaries to not only track the user's online activities but also associate her attributes across multiple RPs by linking her login requests [**?**].

As SSO becomes a popular safeguard for various privacy-sensitive web services, the privacy concern is considered more prominent and severe than it was in the past. On one hand, privacy-savvy users may provide no or few personal information to web applications to avoid user tracking or profiling. On the other hand, the use of popular SSO services such as Google Account opens a door for IdPs and application providers to recover users' online traces and profiles, which makes users' privacy protection effort in vain. Several large IdPs, especially the social IdPs, are known to be interested in collecting users' online behavioral data for various purposes (e.g., Screenwise Meter [**?**] and Onavo [**?**]). Serving the IdP role makes it possible for them to collect such information. Meanwhile, service providers hosting multiple web applications take an advantaged position to correlate users' multiple logins at different RPs through internal information integration. Finally, privacy-preserving record linkage [**?**] and private set intersection [**?**] technologies allow multiple RPs to share data without violating their clients' privacy, which pave the path for cross-organizational RP-based identity linkage.

Several solutions have been proposed to protect user privacy in SSO login [**?**,**?**,**?**,**?**]. However, to the best of our knowledge, none of them provides a comprehensive protection to defend against IdP-based login tracing and RP-based identity linkage *at the same time*. For example, as recommended by NIST [**?**] and specified in several SSO protocols [**?**,**?**], pairwise pseudonymous identifier (PPID) is generated by the IdP to identify a user to an RP, which cannot be correlated with the user's PPID at another RP. Thus, collusive RPs cannot link a user's logins from her PPIDs. However, PPID-based approaches cannot prevent IdP-based login tracing, since the IdP needs to know which RP the user visits in order to generate the correct identify proof. On the contrary, BrowserID [**?**] and SPRESSO [**?**] were proposed to defend against IdP-based login tracing. However, both solutions are vulnerable to RP-based identity linkage. In BrowserID (and its prototypes known as Mozilla Persona [**?**] and Firefox Accounts [**?**]), the IdP does not know the identity of the requesting RP. Instead, it generates a special "identity proof" to bind the user's unique identifier (e.g., email address) to a

public key, so that the user can sign another subsidiary identity proof to bind her identity with the RP's identity and send both identity proofs to the RP. Obviously, when a user logs in to different RPs, the RPs can extract a same user identifier from different identity proofs and correlate these logins. In SPRESSO, the RP creates a one-time pseudo-identifier in each login. Then, the IdP generates an identity proof binding this pseudo-identifier and the user's identity (i.e., email address). Similarly, the RPs can correlate a user's logins using her unique identifier in the identity proofs.

Recently more and more IdPs have been considering the user's privacy serious. For example, one of the most worldwide popular instant messaging applications, WeChat, also working as the IdP service provider, enables a user to create different plain accounts for login on multiple RPs, shown in Figure 1. Moreover, Active Directory Federation Services and Oracle Access Management support the use of PPID [**?**,**?**]; identity service providers such as NORDIC APIS and CURITY suggest adopting PPID in SSO to protect user privacy [**?**,**?**].

Unfortunately, the techniques proposed by previous research cannot be directly integrated to address the two major types of privacy risks in SSO at the same time. In fact, it requires a non-trivial redesign of the SSO system to defend against IdP-based login tracing and RP-based identity linkage while providing a secure and compatible SSO service. In this paper, we first conceptualize the privacy problem in SSO as *an identifier transformation problem* and explain the reasons that limit existing solutions from fully protecting user privacy against curious IdPs and collusive RPs. Based on our analysis, we propose an Unlinkable Privacy-PREserving Single Sign-On (UPPRESSO) system to provide a comprehensive protection against both types of privacy attacks.

UPPRESSO designs three one-way identifier-transformation functions based on elliptic curve cryptography. Using the one-way trapdoor function $\mathcal{F}_{ID_{RP}\mapsto PID_{RP}}(ID_{RP}, T)$, the RP converts its identity ID_{RP} into a privacy-preserving pseudo-identifier PID_{RP} based on a randomly selected trapdoor T. Similarly, the IdP uses the one-way function $\mathcal{F}_{ID_U\mapsto PID_U}(ID_U, PID_{RP})$ to generate a privacy-preserving pseudo-identifier PID_U for the user based on her identity ID_U and PID_{RP}. Finally, using a special identifier-transformation function $\mathcal{F}_{PID_U\mapsto Account}(PID_U, PID_{RP}, T)$, the RP is able to map all the different privacy-preserving pseudo-identifiers of a user, which are created in her different login sessions to that RP, to a same *Account* that identifies the user to the RP. The three identifier-transformation functions work cooperatively to ensure: (a) when a user logs in to an RP multiple times, the RP can always map PID_Us to a unique *Account* without knowing the user's identity ID_U; moreover, when a user logs in to multiple RPs, (b) a curious IdP learns nothing about the identities of these RPs from PID_{RP}s, and (c) collusive RPs cannot link PID_Us to a particular user (d) nor correlate *Account*s of a same user at different RPs. We summarize our contributions as follows.

– We are among the first to conceptualize the privacy problem in SSO as an identifier-transformation problem and analyze the strengths and limitations of existing SSO privacy protection solutions.

- We propose a comprehensive solution to hide the users' login traces from curious IdPs and collusive RPs. To the best of our knowledge, UPPRESSO is the first SSO system that secures SSO services against IdP-based login tracing and RP-based identity linkage.
- We provide the reduction from UPPRESSO scheme to DDH Assumption proving that it is protected from IdP-based login tracing and RP-based identity linkage, and analyze the security of UPPRESSO based on a formal model of the web infrastructure and formally prove that it provides satisfying security properties.
- We implement a prototype of UPPRESSO based on an open-source implementation of OIDC, which requires only small modifications to support three identifier-transformation functions for privacy protections. Thus, UPPRESSO is compatible with existing SSO systems. Moreover, our prototype leverages HTML 5 features in the implementation so that it can be used across platforms (e.g., PCs, smart phones and other devices).
- We compare the performance of the UPPRESSO prototype with the state-of-the-art SSO systems (i.e., OIDC [**?**] and SPRESSO [**?**]) and demonstrate its efficiency.

The rest of the paper is organized as follows. We first introduce the background and preliminaries in Section **??**. Then, we describe the identifier-transformation-based approach and the threat model in Sections **??** and **??**. Section **??** presents the details of our UPPRESSO design, followed by a formal analysis of its privacy and security in Section **??** and **??**. We explain the implementation specifics and experiment evaluation in Section **??**, discuss the extensions and related works in Section **??** and **??**, and conclude our work in Section **??**.

3 First Section

3.1 A Subsection Sample

Please note that the first paragraph of a section or subsection is not indented. The first paragraph that follows a table, figure, equation etc. does not need an indent, either.

Subsequent paragraphs, however, are indented.

Sample Heading (Third Level) Only two levels of headings should be numbered. Lower level headings remain unnumbered; they are formatted as run-in headings.

Sample Heading (Fourth Level) The contribution should contain no more than four levels of headings. Table 1 gives a summary of all heading levels.
Displayed equations are centered and set on a separate line.

$$x + y = z \tag{1}$$

Please try to avoid rasterized images for line-art diagrams and schemas. Whenever possible, use vector graphics instead (see Fig. 2).

Table 1. Table captions should be placed above the tables.

Heading level	Example	Font size and style
Title (centered)	**Lecture Notes**	14 point, bold
1st-level heading	**1 Introduction**	12 point, bold
2nd-level heading	**2.1 Printing Area**	10 point, bold
3rd-level heading	**Run-in Heading in Bold.** Text follows	10 point, bold
4th-level heading	*Lowest Level Heading.* Text follows	10 point, italic

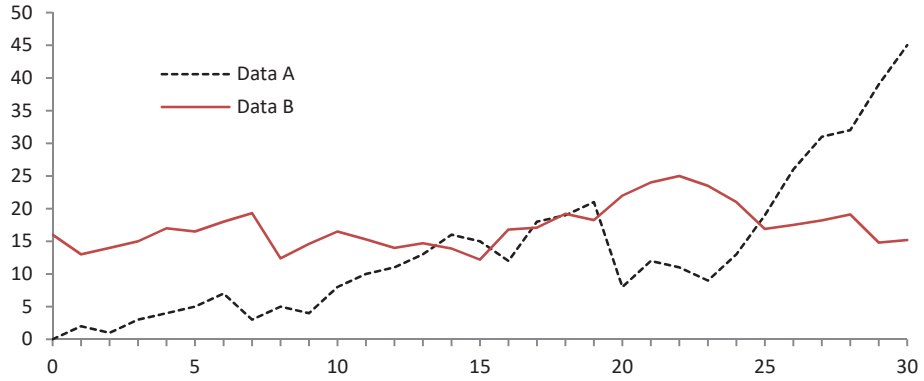

Fig. 1. A figure caption is always placed below the illustration. Please note that short captions are centered, while long ones are justified by the macro package automatically.

Theorem 1. *This is a sample theorem. The run-in heading is set in bold, while the following text appears in italics. Definitions, lemmas, propositions, and corollaries are styled the same way.*

Proof. Proofs, examples, and remarks have the initial word in italics, while the following text appears in normal font.

For citations of references, we prefer the use of square brackets and consecutive numbers. Citations using labels or the author/year convention are also acceptable. The following bibliography provides a sample reference list with entries for journal articles [1], an LNCS chapter [2], a book [3], proceedings without editors [4], and a homepage [5]. Multiple citations are grouped [1–3], [1, 3–5].

References

1. Author, F.: Article title. Journal **2**(5), 99–110 (2016)
2. Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016). https://doi.org/10.10007/1234567890
3. Author, F., Author, S., Author, T.: Book title. 2nd edn. Publisher, Location (1999)
4. Author, A.-B.: Contribution title. In: 9th International Proceedings on Proceedings, pp. 1–2. Publisher, Location (2010)

5. LNCS Homepage, http://www.springer.com/lncs. Last accessed 4 Oct 2017