Dear Shepherd,

We have submitted the revised manuscript for your review. This revision follows the suggested “revision requirements” and our proposed revision plan. We summarize the main changes as follows, which are marked blue in this version.

**Revision requirements:**

1. **Remove or clean-up section 3.2.**

*Modifications:*

1. We removed Section 3.2 “The Identity dilemma of Privacy-Preserving SSO” in the original manuscript.
2. In this version, we highlighted the security requirements of SSO in Section 3.1 and explained the requirements of identity transformations in Section 3.2.
3. **Refine the security analysis: define proper notions to prove if you do a formal proof and provide better intuitions on the challenges in proving each notion w.r.t the chosen adversarial model. Make clear which assumptions each property relies on (e.g., pull them all into a theorem)**

*Modifications:* We heavily revised the analysis of security and privacy in Section 5.

1. The threat model in Section 4.1 considers four types of adversaries, an honest-but-curious IdP, malicious RPs, malicious users, and colluding RPs and users. Following this threat model, we added a new subsection 5.1 to define three adversarial scenarios.
   1. We considered colluding RPs and users who aim to intrude an honest user’s login flow. The attack goal is to break the security of SSO. We prove the security of UPPRESSO in this adversarial scenario. We did not prove the security of UPPERSSO in cases with only malicious users or only malicious RPs, as it is straightforward to extend the current proof to draw the conclusion.
   2. We consider the honest-but-curious IdP that aims to infer the identity of the RPs. We prove that UPPRESSO prevents IdP-based login tracing in this adversarial scenario.
   3. We consider the colluding RPs and users who aim to associate the logins to colluding RPs to link an honest user’s accounts across these RPs. UPPROSSO is proved against RP-based identity linkage in this adversarial scenario.
   4. In Table 3, we defined the notations used in the manuscript. We clarified the assumptions about the random numbers used in the UPPRESSO protocol.
2. We improved Section 5.2 to describe how to develop a Dolev-Yao style model to formally model the processing of messages and their attributes in the UPPRESSO protocol. Applying this model, we drew six conclusions to show that the identity tokens, pseudo-identities, random variables, etc. cannot be accessed or modified by unauthorized entities in the proposed protocol.
3. We revised Section 5.3 to prove the security of UPPRESSO. First, we defined four theorems to prove that UPPRESSO supports each of the four security requirements defined in Section 3.1, respectively. Then, we proved Theorem 5 to show that any SSO protocol satisfying these four requirements is considered secure.
4. We revised Section 5.4 to prove that UPPRESSO prevents two privacy threats. We proved Theorem 6 and Theorem 7 to show its protections against IdP-based login tracing and RP-based identity linkage, respectively.
5. **Clarify the protocol design in the write-up (i.e., the check in 3.3, and the double computation of PID\_RP).**

*Modifications:*

1. We clarified the statement in Step 3.3 by adding a sentence “the IdP checks if the received $PID\_{RP} is a point on E”.
2. We removed the calculation of PID\_RP in Step 2.2. However, this calculation cannot be completely avoided in the protocol, because in Step 4.2, the RP is required to “verify if PID\_RP in the received identity token matches the pseudo-identity negotiated in Step 2.2”. In this version, we moved this calculation of PID\_RP to Step 4.2: “The RP extracts PID\_RP from the token and checks if it equals [t]ID\_RP”, to make the protocol logic clearer to follow.
3. It is important for the RP to check if the received PID\_RP is associated with the current login (i.e., t) and for itself (i.e., ID\_RP). Otherwise, the attacker can manipulate PID\_RP to break the user identification or RP designation properties. We added two paragraphs in Section 4.3 to explain two possible attacks.
4. **Elaborate on the weakness of having t in the IdP context or integrating a suitable MPC scheme, as noted in the response.**

*Modifications:*

1. We added a paragraph in Section 4.4 to explain that t is critical for the RP and the user (using the IdP script) to calculate the PID\_RP independently. Based on our threat model, the IdP is honest-but-curious, so the IdP script downloaded from the IdP is also considered honest. We explained that the IdP script should be trusted because it knows the RP’s domain which reveals the RP’s identity. Therefore, MPC-based solutions are not necessary.
2. An alternative solution is to use a trusted browser extension instead of the IdP script. We do not recommend this design because it requires users’ agreement and involvement to install the extension for each IdP in every browser they use.
3. **Illustrate the differences compared to PrivacyPass [27] and TrustTokens [26] and highlight where the protocol actually shows significant novelty.**

*Modifications:* (edit needed)

1. We revised Sections 2.2 to provide a conceptual categorization of the existing solutions related to privacy-preserving identity management systems and classified the existing work into privacy-preserving SSO, privacy-preserving identity federation, and anonymous identity federation.
   1. Privacy-preserving SSO solutions are compatible with representative SSO protocols, while preventing one or multiple privacy threats. Our UPPRESSO protocol belongs to this category.
   2. PrivacyPass/TrustToken adapted the oblivious pseudorandom function (OPRF) protocol to generate anonymous tokens. It was not designed for SSO services and therefore cannot be used directly to support SSO (so, we did not include it in Table 1). With some modification, it could support an SSO-like service, in which the users only need to authenticate once at the token server and then use the anonymous tokens to access the server. However, this application still cannot support the full SSO service since it does not support user identification at RPs. The same user with different anonymous tokens would be considered as different users by the same RP.
2. We also compared the cryptographic technique used in PrivacyPass/TrustToken and in UPPRESSO. We added this discussion in Section 2.3 “Anonymous Tokens and OPRF Applications”.
   1. Both UPPRESSO and PrivacyPass/TrustToken adapted a cryptographic technique that was first proposed for oblivious pseudo-random functions (OPRFs). We added three new OPRF references [38, 39, 51] in the revised manuscript. PrivacyPass/TrustToken utilizes OPRFs to generate anonymous tokens and the obliviousness property to ensure the token signing and redemption are unlinkable. Similarly, UPPRESSO utilizes the oblivious pseudo-random properties to generate the pseudo identities of RPs and users, and the obliviousness property to ensure the RPs’ pseudo-identities are indistinguishable from each other.

[38] S. Jarecki, A. Kiayias, and H. Krawczyk. Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In 20th International Conference on the Theory and Application of Cryptology and Information Security (AsiaCrypt), pages 233–253, 2014.

[39] S. Jarecki, A. Kiayias, H. Krawczyk, and J. Xu. Highly efficient and composable password-protected secret sharing (or: How to protect your Bitcoin wallet online). In 1st IEEE European Symposium on Security and Privacy (EuroS&P), pages 276–291, 2016.

[51] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. Journal of the ACM, 51(2):231–262, 2004.

* 1. PrivacyPass/TrustToken directly adopted the OPRF construct to generate anonymous tokens by utilizing a random number e to blind each token. Like the OPRF protocol, it is a two-party protocol involving the token server and the users. It unlinks the token generation and token redemption processes to protect user privacy when using the token. However, SSO systems naturally involve three parties, which pose several challenges to OPRF-based designs. PrivacyPass/TrustToken cannot address these challenges without non-trivial modifications. For example, when attempting to log into an RP, a user needs to embed the login context into a token before the server signs it. This login context inevitable reveals the RP’s identity. Meanwhile, the anonymous tokens are designed to be indistinguishable from each other, but the SSO services requires the users to be uniquely identifiable at the RPs. In this process, UPPRESSO applies pseudo-random functions but uses the secret key of pseudo-random functions as a user identity, while it is used as secret keys in other OPRF-based applications. This OPRF application requires an in-depth understanding of the variables in OPRFs and the (pseudo-)identities in SSO services.
  2. We leverage the properties of OPRFs to provide an additional privacy property for SSO, called RP unlinkability, which is not considered or supported in PrivacyPass/TrustToken. It ensures that colluding RPs cannot link any two logins across RPs, even if they share the knowledge about the pseudo-identities and permanent accounts of the users who initiate these logins. This privacy protection is proved in Theorem 7.
  3. UPPRESSO explores even more properties of the pseudo-random functions than OPRF protocols. It uses the obliviousness property to ensure IdP untraceability, the deterministicness property of pseudo-random functions to enable the RP to derive the permanent account for any t, and the randomness property of pseudo-random functions to provide RP unlinkability. However, UPPRESSO requires an additional property in SSO services, called RP designation, and it is ensured only if no collision exists in RPs’ pseudo-identities (see Theorem 1 and Lemma 1), which are actually the blinded inputs of the evaluated pseudo-random function. This property is not required in other OPRF-based approaches and therefore may not be supported by all OPRF protocols. Thus, an OPRF protocol cannot always be applied to implement identity transformations in UPPRESSO, unless no collision exists in the blinded inputs of the pseudo-random function.

1. **In this revision, we also incorporated suggestions in each individual review. Here, we briefly summarize the changes beside the above 5 revision requirements.**
2. Review A: (a) the prototype of UPPRESSO is open-sourced at https://github.com/uppresso. We added the URL to the footnote on Page 13. (b) we improved the writing of the Abstract. (c) We added the validity field to the identity token in Figure 1. (d) we added a sentence about r in Table 3 to describe that r is known only to the IdP but not the RP.
3. Review B: (a) we added a detailed comparison with PrivacyPass and TrustedTokens in Section 2.3. (b) FedCM focuses on disabling iframe and third-party cookies that may be exploited to track users. So, it is orthogonal to UPPRESSO but could be used as an additional protection measure. We added a brief discussion at the end of Section 4.2. In Section 2.4, we also cited PESTO that builds a proactively-secure distributed SSO system by combining distributed partially-oblivious PRFs and distribution signatures.
4. Review C: (a) we removed the identity dilemma discussion in original Section 3.2. (b) Identity transformation algorithms were defined in Section 3.3. (c) adversarial scenarios and formal proofs were added for the security and privacy analysis. (d) We clarified in the assumptions in Section 4.2 that the communications between the user and the IdP are over HTTPS and thus considered secure. Meanwhile, tracking by network traffic analysis or crafted web pages/iframes are out of the scope of this work. (e) We improved the writing of the Abstract, clarified the notations in Table 1, fixed the typo in Figure 4, and polished the writing of the paper.
5. Review D: (a) UPPRESSO is designed to support existing SSO protocols. We discussed its compatibility with the widely used OIDC protocol in Section 4.6 and its integration with commercial-off-the-shelf browsers in Section 4.4. (b) We added formal proof (Theorem 7) to show that UPPRESSO can prevent colluding RPs and colluding users from linking logins of honest users. (c) The SSO protocols commonly consider IdP as honest because it maintains the long-term credentials of the users. A malicious IdP could perform arbitrary actions on the user’s behalf. We then extend this assumption to consider an honest-but-curious IdP since many Internet service providers tend to track users’ online activities based on the information they observe. (d) Since RSA-2048 was used in MITREidConnect and SPRESSO, we also used it in the prototype for fair comparisons.
6. Review E: (a) We will discuss potential post-quantum crypto algorithms for UPPRESSO. Did we? [根据现有综述，OPRF只有2个方案是PQC的；是不是可以在2.3的最后，我们讲一句“我们会在现有的PQC OPRF方案中，看看是不是有blinded input不碰撞特性，进而考虑可行性；是我们的future work”] (b) We corrected the editorial issues in the revised manuscript.
7. **We carefully polished the writing of the manuscript.**

*Modifications:*

We made small edits throughout the paper to improve its presentation. To avoid distractions, we did not mark these edits in color blue.