**\textbf{1. IdP-RP collusion} [Review-B/C]**

**As discussed in Section 7, if a malicious IdP colludes with malicious RPs, a user has to finish a login \emph{entirely} with colluding entities. Then, a user needs to hold a long-term secret to mask the relationship of the (pseudo-)identities in tokens and accounts at the RPs; otherwise, privacy is broken.**

**However, a long-term use secret requires browser extensions or plug-ins, violating our design goals.**

**\textbf{2. Objectives, analysis and theorems} [Review-A/C]**

**Our security objective is described in Section 5.2: \emph{An identity token $TK$ requested by a user to visit an honest RP, enables only this user to log into only this RP as her account at this RP.} It is achieved under the assumption of an honest IdP, malicious RPs and malicious users, given in Section 5.1.**

**It is decomposed into two properties:**

**\emph{RP Designation} (Theorem 1): $TK$ designates the target RP, and based on $TK$ only this designated RP derives a meaningful account which responds to some user.**

**\emph{User Identification} (Theorem 2): At the designated RP, $TK$ identifies only the user requesting $TK$ from the IdP.**

**Two privacy objectives are given in Sections 5.3 and 5.4, namely \emph{IdP untraceability} against IdP-based login tracing and \emph{RP unlinkability} against RP-based identity linkage, respectively. These objectives are analyzed and proved as below.**

**IdP untraceability (Theorem 3): All information about the target RP obtained by the honest-but-curious IdP, i.e., $PID\_{RP}$, is indistinguishable from random variables.**

**RP unlinkability (Theorem 4): Malicious RPs and colluding users cannot link any login visiting some malicious RP by an honest user to any subset of logins visiting any other malicious RPs by honest users. That is, the logins are indistinguishable to colluding RPs.**

**We will re-organize the designs and the protocol, and present stricter proofs.**

**\textbf{3. Intuition of the identity-transformations approach} [Review-A]**

**In regular SSO systems, an IdP traces a user's logins because the unique identity of the visited RP needs to be known by the IdP to sign an identity token, enclosing the RP identity. An ephemeral RP identity (i.e., $PID\_{RP}$) is used, to prevent IdP-based login tracing.**

**In regular SSO systems, colluding RPs link a user because the unique user identity is enclosed in tokens to visit these RPs. An ephemeral user identity (i.e., $PID\_U$) is used to prevent RP-based identity linkage.**

**Here comes the main challenge: A token encloses both $PID\_{RP}$ and $PID\_U$, but it needs to correctly designate the visited RP and correctly identify the user requesting this token. A \emph{permanent} account needs to be derived, and such a token cannot be exploited to (1) visit other RPs, i.e., RP designation in Theorem 1, or (2) impersonate other users, i.e., user identification in Theorem 2.**

**\textbf{4. Performance and Comparison} [Review-B]**

**The overheads mainly result from script downloading and browser window opening. While our main contribution is the identity-transformation approach to protect user privacy in SSO, UPPRESSO performs better SPRESSO preventing only IdP-based login tracing.**

**We compare UPPRESSO with MITREid Connect (OIDC w/ PPID) and SPRESSO. According to the designs, BrowserID and POIDC shall perform almost the same as MITREid Connect, especially in WAN.**

**\textbf{5. Account rotation and forward privacy} [Review-B]**

**Current designs do not consider forward privacy, because we aim to support non-anonymous SSO where an RP provides services customized for a certain user.**

**In UPPRESSO a user’s identity is generated and always restored by hashing her username concatenated with the IdP’s private key. To rotate a user’s accounts and support forward privacy is rather straightforward: a user’s identity is generated and restored by hashing her username concatenated with the IdP’s private key and a validity period (e.g., year = 2025).**

**\textbf{6. Privacy threat from distinctive attributes and cross-site tracking} [Review-C]**

**Such privacy threats can be prevented by other mechanisms. However, even when such mechanisms are successfully deployed, two threats (i.e., IdP-based login tracing and RP-based identity linkage) due to the login flow of SSO protocols still exist.**

**UPPRESSO prevents these two privacy threats which are introduced by SSO protocols, and works complementarily with others.**

**\textbf{7. User convenience} [Review-C]**

**UPPRESSO needs multiple tabs (browser windows), but it brings little inconvenience because this UX has been adopted in Google Identity for several years and receives not critical complains.**