Comment 1:

Dear Shepherd,

We followed the “revision requirements” and took the first round of revision. The first revision draft was uploaded to the system for your review.

Meanwhile, we recognized that it may be difficult to track the modifications and map them to the revision requirements. So, we added the below plan to explain how we understand each requirement and how we plan to address it. We hope to get your agreement and approval on the plan before we make further changes. Any comments or suggestions from you are highly appreciated.

Comment 2:

Dear Shepherd,

Based on the revision requirements, we came up with a revision plan as follows.

\textbf{1. Remove or clean-up Section 3.2}

\textbf{Revision 1:} We will remove Section 3.2 as suggested. We agree with the reviewers that the identity dilemma discussed in this subsection was specific to our proposed solution only. So, we plan to remove all the texts related to the identity dilemma in the manuscript, and instead, directly explain the proposed identity-transformation concept for securing SSO in this section. Accordingly, we will integrate Section 3.3 of the previous manuscript into Section 3.2 of this revision.

\textbf{2. 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).

\textbf{Revision 2.1:}

We defined the general threat model in Section 4.1. We plan to elaborate the adversarial model and the assumptions about the adversaries, which include (1) an honest-and-curious IdP, (2) collusive RPs, and (3) an malicious RP colluding with a malicious user. This adversarial model will be a new subsection in Section 5 “The Analysis of Security and Privacy”, before we prove the security/privacy notions.

\textbf{Revision 2.2:} We plan to formally define the notions to prove, which include 4 SSO security requirements, namely RP designation, user identification, integrity, and confidentiality, and 2 privacy requirements, namely privacy against IdP and privacy against colluding RPs. In the definitions of notions, we will emphasize if an argument is known (or unknown) to each entity, which is consistent with the adversarial model.

\textbf{Revision 2.3:} Then, we will provide a formal proof to prove each of the six security/privacy requirements under the adversarial model. We will also extend Table 3 to emphasize if an argument is known (or unknown) to each entity.

\textbf{3. Clarify the protocol design in the write-up} (i.e., the check in 3.3, and the double computation of PID\_RP).

\textbf{Revision 3.1:} Sorry for the confusion. In Step 3.3, “the IdP checks whether the received $PID\_{RP} is valid” is to ensure that the received $PID\_{RP} is a point on the EC curve. We will rephrase this statement to avoid confusion.

\textbf{Revision 3.2:} We acknowledged the double computation issue and carefully reviewed the protocol to see if we could improve it. We found that it seems both the user and the RP need to compute PID\_RP independently to ensure the security of the protocol. We will explain the reason below. Please see if you agree with this argument.

$PID\_{RP}$ is a secret value negotiated between the user and the RP following the equation $PID\_{RP} = [t]ID\_{RP}$, where $t$ is a random number. $t$ can be selected by either party and securely shared with the other party. Without loss of generality, we let the user select $t$ and share it with the RP. $t$ cannot be disclosed to or derived by any other entities in the system. Since our threat model considers malicious users and collusive RPs, an honest user or an honest RP needs to negotiate $PID\_{RP}$ in the process and compute the number by itself. They cannot trust the $PID\_{RP}$ received from another party.

We describe two possible attack scenarios as follows.

1. Let’s assume the user generates $t$, calculates $PID\_{RP} = [t]ID\_{RP}$, and sends both to the RP, which will extract $PID\_{RP}$ from a token. If the RP does not verify the received $PID\_{RP} by computing $PID\_{RP} = [t]ID\_{RP}$ with the received $t$, a malicious user can choose a random $t'$ and compute $PID\_{RP}=[t'u'^{-1}]Acct$, where $Acct$ is the account of a victim user to impersonate.

Here, if the RP verifies the $PID\_{RP}$, it can find $PID\_{RP} \neq [t’]ID\_{RP}$ and reject the request. But if it does not check the received $PID\_{RP}$ and uses it directly, it will compute $[t'^{-1}]PID\_{U'} = Acct$, where $PID\_{U'} = [u'][t'u'^{-1}]Acct = [t']Acct$.

2. If the RP chooses a random $t$ and computes $PID\_{RP}$, the user also needs to check if $PID\_{RP} = [t]ID\_{RP}$ by himself. Otherwise, the attacker can log in on behalf of any victim user. For example, if a malicious user $U'$ initiates a login request to an honest RP and receives $PID\_{RP}$. It can collude with a malicious RP (denoted as $RP'$), which sends $PID\_{RP}$ to an honest user $U$. The honest user should check if $PID\_{RP} = [t']ID\_{RP'}$ holds using the received $t’$ and $ID\_{RP’}$ from $RP'$ and reject it. Otherwise, if the user blindly trusts the received $PID\_{RP}$, it will present a token binding $PID\_U$ and $PID\_{RP}$ to $RP'$. This token will enable the malicious $U'$ to login on behalf of the honest user at the honest RP.

In conclusion, we believe that both the user and the RP need to compute the $PID\_{RP}$ independently; or, more exactly, one party computes $PID\_{RP} = [t]ID\_{RP}$, and the other party checks if $PID\_{RP}$ is equal to $[t]ID\_{RP}$ or not.

\textbf{4. Elaborate on the weakness of having t in the IdP context or integrating a suitable MPC scheme}, as noted in the response.

\textbf{Revision 4.1:} We plan to add a discussion in Section 4.4 to clearly state that $t$ is critical to calculating $PID\_{RP}$ and it should not be disclosed to any unintended party in any means. In practice, the user will download scripts from IdP to deal with all the computation at the browser side. Since we assume the IdP is honest-but-curious, we extend this trust to the script downloaded from the IdP. We will note the leakage risk of malicious IdP scripts in Section 4.4.

\textbf{Revision 4.2:} We re-evaluated the suggestion of adding MPC into the protocol scope. Under the current threat model, we do not need MPC to protect $t$ from the IdP script. If we modify the threat model and assume a malicious IdP script, we found that it could extract $ID\_{RP}$ from $Cert\_{RP}$ and leak it directly to the IdP. In this case, MPC does not help much in preventing the leakage.

Based on this consideration, we propose to keep the current threat model and explore other prevention in the future work. For example, we will add a discussion about the use of trusted browser extensions to prevent potential leakage of the RP’s identity (i.e., $ID\_{RP}$). Please instruct if this plan is appropriate.

\textbf{5. Illustrate the differences compared to PrivacyPass [27] and Trust-Tokens [26], and highlight where the protocol actually shows significant novelty.}

We plan to improve the discussion of related work in Section 2.3 and add comparison with recent related work, including PrivacyPass [27] and Trust-Tokens [26].

\textbf{Revision 5.1:} In particular, we consider PrivacyPass and Trust-Tokens as “anonymous tokens”. PrivacyPass [27] and Trust-Tokens [26], similar as UPRESSO, adopt cryptographic techniques proposed in [1] for OPRFs. We will add discussion about this comparison in terms of cryptographic constructs and techniques in Section 2.3.

\textbf{Revision 5.2:} Then, we plan to elaborate the novelty of UPRESSO, compared with PrivacyPass and Trust-Tokens, from three aspects: (1) UPPRESSO identifies each user at an RP, while PrivacyPass and TrustToken anonymous tokens support anonymous SSO service, which rely on one consistent private key to serve all users. (2) UPPRESSO utilizes the cryptographic technique differently from PrivacyPass and Trust-Tokens, by using them to transform identities in SSO. (3) UPPRESSO supports more privacy requirements, i.e., the unlinkability across RPs, than anonymous SSO enabled using PrivacyPass and Trust-Tokens. This property of the cryptographic skills is not considered in either OPRFs or anonymous tokens.

We will add this new reference to the revised manuscript.

[1] 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 (EuroSP), 2016, pp. 276–291.

\textbf{Revision 6:} We will rigorously improve the writing of the paper and address typos/errors in the presentation.