Regular Paper

Centralized Control of Account Migration at Single Sign-On in Shibboleth

Satsuki Nishioka[1,†1,a)] Yasuo Okabe[2,b)]

Received: February 24, 2021, Accepted: September 9, 2021

Abstract: Single Sign-On (SSO) is adopted to use multiple services with a single log-in on the Internet. However, when a user tries to change the identity provider (IdP) which is responsible for authenticating the user, he needs to release the binding between the log-in account on the migration-source IdP and his service account on each service provider (SP) and needs to set a new binding between the account on the migration-destination IdP and the service account on the SP. There is no common migration system to support migration using the SSO function. In this research, we focus especially on Shibboleth's function as an SSO service. We propose a protocol to migrate accounts of a user on multiple SPs at once using an attribute provider (AP) in an SSO environment. We have implemented the mechanism as an open-source software using SimpleSAMLphp.

Keywords: authentication, identity management, Single Sign-On, Shibboleth, SimpleSAMLphp

1. Introduction

When using various services on the Internet, accounts are commonly used by Service Providers (SPs) to identify each user. An account enables a user to share information between multiple devices or to check his service usage history only by registering and memorizing a set of ID/password and entering it when using the service. However, the more services a user uses, the more accounts he needs. Then he needs to remember a huge set of ID/passwords. This leads not only to a hassle to remember passwords but also to loss of services due to passwords. In fact, there is an undesirable reality that many users are reusing the same passwords for different services.

Single Sign-On, SSO is considered as one of the technologies attempting to solve this problem. SSO allows a user to use multiple services through a single log-in authentication with a single account. In SSO, you can use your existing account of an Identity Provider (IdP) to log in to services offered by SPs. There is no need to set a new ID/password on each of the SPs. You only have to have one account of an IdP for using multiple SPs, without registering an account and remembering a password of each SP [1]. However, in current authentication federation [2] systems that link multiple IdPs and SPs, there is no way provided to migrate all the accounts of a user on the SPs he uses at once when the user changes his IdP that is used for log-in [3]. Despite the centralized account management at the IdP, the migration must be handled by the users at each SP. This places an extreme burden on users with problems such as missing the migration procedure on some SPs. In this study, we aim to realize an authentication federation system that can migrate the accounts of a user on multiple SPs at once so as to reduce the burden on the user.

For example, assume that a user has used single sign-on services with various SPs through the IdP of a university he belongs to, and he now has graduated and moves to another university. Due to the change in affiliation, the single sign-on services using the account issued by the previous university cannot be used anymore, and it is necessary for him to rebind his account on the IdP of the university he now has entered to the account of each SP he has been used via single sign-in. We focus on such a situation and give a solution.

Shibboleth [4], an open-source project based on SAML (Security Assertion Markup Language) [5], is an example of the implementation of the authentication federation among academic institutions. There is an excellent system in Shibboleth for preserving user privacy. In the authentication federation, IdPs and SPs can identify users with pseudonyms so as to achieve SSO preserving anonymity [6]. However, this anonymity that is the hallmark of Shibboleth makes it difficult to migrate multiple accounts at once. Thus we set our goal to preserve pseudonymity during account migration in Shibboleth-based authentication federation.

As an approach to overcome the challenge, we propose the use of an Attribute Provider (AP). An ordinary AP maintains and manages user attribute information on behalf of IdPs. It provides the information of a user upon the request of SPs [7]. Making use of this feature, we consider having an AP maintains and manages migration ID issued by each SP through the migration. With the AP, a user can migrate accounts on multiple SPs logged in with SSO with a single migration procedure. This drastically re-

[1] Graduate School of Informatics, Kyoto University, Kyoto 606–8051, Japan
[2] Academic Center for Computing and Media Studies, Kyoto University, Kyoto 606–8051, Japan
[†1] Presently with KDDI Corporation
[a)] nishioka@net.ist.i.kyoto-u.ac.jp
[b)] okabe@media.kyoto-u.ac.jp

A preliminary version of this paper was presented at 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC).

duces the burden on the user. On the other hand, this approach highly depends on the trustability of the AP. This is not to say that there is no security concern that a malicious AP could link another user's account or could initiate migration procedure even if the user does not want this. Therefore, we add some constraints depending on the trustability of the AP and provide grades according to acceptable risk. We propose a configuration enabling users to select one among three protocols according to risk tolerance and AP trustability.

In Section 2, we define the problem addressed in this study, before getting down to the main subject. Then we describe the procedure for account migration using an Attribute Provider in Section 3. The implementation of this protocol is described in Section 4. In Section 5, we discuss issues on the system proposed in Section 3 and countermeasures against them. Finally, Section 6 summarizes this study.

2. The Account Migration Problem in Single Sign-On

In this section, we describe the problem caused by Shibboleth's SSO features that occurs in achieving our goal. And we will define the requirements for this study, considering the problems. Moreover, we will mention related studies.

2.1 Difficulties in Continuity of Service Use through Privacy-Aware Account Migration

In Shibboleth-based SSO, an IdP distributes different pseudonym IDs for each SP when passing credentials, even for the same user. It makes users anonymous from the SP's perspective [8]. Let us assume that there are two SPs, "SP X" and "SP Y", federated to an IdP. For example, consider the case that there is a user, "user1", who is logged in to "SP X" and "SP Y" using the IdP. In this case, the IdP distributes pseudonym IDs such as "ID X" for "SP X", and "ID Y" for "SP Y". These pseudonym IDs allow SPs to identify the user authenticated by the IdP by matching the SP's internal ID [9]. At this time, no one excepting the IdP can identify the "user1" as the same user because the pseudonym IDs sent to "SP X" and "SP Y" are different. As above, it is impossible to identify the same user among SPs.

In this study, migrating account information collectively means completing the account migration for multiple SPs in a single procedure. To achieve this, one possible approach is that when an SP changes the binding between the IdP's pseudonym ID and the SP's internal ID, the SP shares the change information with other SPs, and the other SPs change the bindings themselves. However, this approach is difficult to implement due to the pseudonymity of the users, as is mentioned in Section 1 of Ref. [10].

Alternatively, if we try to migrate the account by sharing the information between IdPs, the migration-source IdP needs to know the migration-destination IdP. But a user may not want the source IdP to know the destination IdP. It is difficult for him to migrate his accounts collectively without sharing any extra information among SPs and between source and destination IdPs.

To make matters more complicated, there may be a case that it is not until a certain period of time has passed after a user's account at the source IdP has been revoked that the user's new account at the destination IdP is issued and becomes available. Although it would be easier to implement a protocol in which a user emigrates from the source IdP and then immediately immigrates to the destination IdP, we would like to consider the above case as well. Considering the difficulties discussed above, we aim to realize the protocol satisfying the following three requirements.

(1) Being able to migrate accounts without sharing information among federated SPs
(2) Being able to migrate accounts without letting the migration-destination IdP and the migration-source IdP know which IdP each other is
(3) Allowing a period between move-out from the source IdP and move-in to destination IdP.

2.2 Related Study

As described in Section 1, there is no system to migrate accounts on multiple SPs at once when changing the IdP in Shibboleth-based authentication, as far as the authors know, but several patents have been filed for a similar purpose, as below.

Hinton [11], suggested a method to change the IdP used to log in to an SP in authentication federation by a user who has accounts in IdPs and the SP. In this patent, the method allows changing the IdP used to log-in while the account at the SP is maintained, as a part of the process of revoking the account at the source IdP. The method proposed in this patent support migration only at a single SP. Thus the purpose of this patent is essentially different from ours that is to migrate accounts on multiple SPs logged in with the same IdP at once, but it has something in common with our study in that it is a technology to change the IdP while maintaining the anonymity of the user from the SP's point of view.

Cameron et al. [12] proposed a way to integrate the processes between a user and an SP in user authentication. The processes include receiving the SP's security policy, determining and obtaining the attribute requested by the SP, registering with a provisioning service based on the attribute, and accessing SP's services based on registration to the provisioning service. These processes are generally performed at each SP, and there is a risk of leakage of confidential information. The method of this patent integrates these processes and allows the information to be transferred securely. It is tremendously important in our study to distinguish properly between information that should be hidden and information that can be transferred among organizations in the authentication federation service. However, as this patent is not focused on identity federation systems including SSO, the method cannot be applied to identity federation systems that consist of IdPs and SPs.

3. Centralized Account Migration with an Attribute Provider

3.1 Migration at a Single SP

Before considering migration of accounts on multiple SPs, which is the purpose of this study, we consider the case that a user uses only a single SP (or wants to migrate an account only

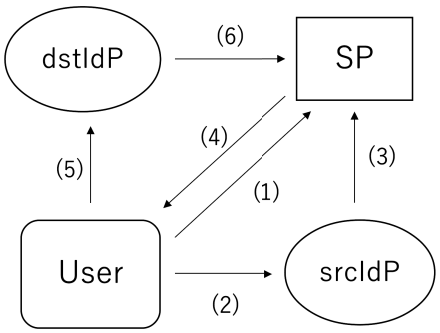

Fig. 1 Account migration at single SP.

on a single SP). In this case, a method using a migration ID can be considered. When there are two IdPs, "srcIdP" and "dstIdP", the following steps allow a user to migrate his account of the SP by changing the IdP used to log-in to the SP from srcIdP to dstIdP.

(1) The user requests an SP to migrate the user account.
(2) The user is led to log-in with srcIdP.
(3) The user is authenticated by srcIdP.
(4) The SP issues a migration ID linked to the user's account information and sends it to the user.
(5) The user logs-in with dstIdP.
(6) After authentication by dstIdP, the user sends the migration ID to the SP, and the SP completes the migration by associating the user's account linked to the migration ID with the currently logged-in user.

The account migration procedure on a single SP can be achieved through the steps above as in **Fig. 1**. The migration ID used here is linked to the user's account information in the SP and consists of a character string of sufficient length. It is issued in Step (4) and displayed on the SP's website. Although there may be services that store this information in the browser or on the IdP, we in general suppose the case where the user needs to memorize this by taking a note or by capturing a snapshot of the screen. By using this method, as mentioned in Section 2.1, the account migration without notifying the source IdP of the destination IdP can be accomplished. If the user performs this procedure to multiple SPs one by one, account migration on all federated SPs will be completed, but this imposes a considerable burden on the user. Thus, in Section 3.2, we consider having an AP perform this procedure on behalf of the user.

3.2 Account Migration on Multiple SPs

In this section, we propose a centralized account migration protocol for multiple SPs, based on the steps for a single SP in Section 3.1. We adopt an AP for intermediating the migration information. An AP is commonly used to hold the user's attribute information and provides it on demand on behalf of IdP [13]. Therefore an AP is considered as suitable for the role of managing the user's migration information and passing it to an SP upon request. The migration procedure using an AP is as follows:

(1) Each SP issues the user's migration ID and sends it to the AP when the user registers with the SP, or at any time after registration.
(2) Performs the procedure to migrate an account on a single SP regarding the AP as an SP.

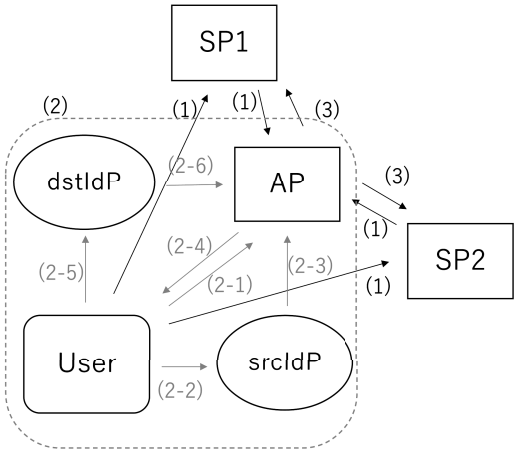

Fig. 2 Centralized account migration.

(3) The AP switches the account link for each SP based on the information in Step (2).

In **Fig. 2**, Step (2-1) to Step (2-6) is equivalent to what is Step (1) to Step (6) in Fig. 1, Section 3.1, respectively. First, the user accesses the AP by using authentication information at srcIdP. Second, the AP issues the migration ID of the user. Then, the user enters the migration ID at the AP after accessing the AP by using dstIdP. The migration ID issued by each SP in Step (1) is linked to the pseudonym ID received from the srcIdP, and is stored in the AP for each SP. Step (3) will be executed on each SP when the user logs in from the destination IdP for the first time.

For example, if the user first accesses SP1 via SSO using dstIdP, the AP sends the migration ID received from SP1 to SP1. After that, the user accesses SP2, the AP also sends the migration ID received from SP2 to SP2. By receiving the stored migration ID from the AP, each SP can get the bindings of the previous account of the user, the migration ID, and the new pseudonym ID received from the destination IdP, and complete migration.

In this protocol, the word migration completion means the binding of the srcIdP's pseudonym ID, and dstIdP's pseudonym ID is obtained at the AP, not considering whether each SP also gets the binding. If the user does not access some SPs registered for migration service at the AP, the accounts of these SPs cannot get binding. However, the user can use previous accounts of these SPs anytime by accessing them and initiating the migration protocol. Furthermore, in this protocol, it is no problem that the user initiates the migration process while the user does not register for the migration service against some SPs (Note that the previous account of the SPs that was not registered for the migration service will not be available after changing the IdP for SSO).

With this protocol, the user only has to perform the migration procedure for a single SP so as to migrate accounts at all SPs using SSO. It greatly improves the user's work efficiency. However, since it is up to the AP to maintain the migration ID and to switch links, there is a risk that when the AP has malicious intent or is hijacked by something malicious, the AP can start migration even if the user does not want to or can switch the account link to any different account. We discuss this issue in Section 5.

4. Implementation

In this study, we implemented a system that simulates the protocol proposed in Section 3.2 [*1,*2]. As an assumed environment, we built two servers each for the IdP and the SP, and one server for the AP. And we utilized an open-source software, SimpleSAMLphp [14], to implement SSO environment [15]. In this study, we focus on Shibboleth, but by using SimpleSAMLphp, we can use this system in other SSO systems as well. Nevertheless, we implemented this system so that it works well even if the user is anonymous. Also, we looked into using GakuNin mAP (present: GakuNin Cloud Gateway Service) [16] for the implementation of an AP. However, it turned out that pseudonymity cannot be ensured because it uses the ID of the real name. Also, it is not available in the case that there is a time-lag during the revoking of the source IdP's account and the issuing of the destination IdP's account. Thus we consider it not to meet the requirements of Section 2.1.

4.1 Database

As the database to manage users on each server, we used MySQL. The contents of user data managed by IdPs, SPs, and an AP are as follow (**Tables 1**, **2**, **3**).

These data are maintained at each entity, for each user. A user ID, "uid" is generated and held as an internal ID, at an IdP, an SP, or an AP. A pseudonym ID, provided to an SP or an AP by an IdP is kept as "uid_idp", tied to their respective "uid". An SP and an AP issue the migration ID and retain it as "mig_id" or "mig_id_ap", and the migration ID of an SP is sent to an AP and managed as "mig_id_sp".

Table 1 The user data managed by an IdP.

label	description	type
uid	the user ID managed by the IdP	char
email	the email address of the user	char
password	the user's password	char
idp	something that represents who the IdP is	char

Table 2 The user data managed by an SP.

label	description	type
uid	the user ID managed by the SP	char
uid_idp	a pseudonym user ID received from an IdP	char
uid_num	the number linked to the user ID (uid) that starts from 1 and updates one by one when registering a new user	int
mig_id	the migration ID issued by the SP	char

Table 3 The user data managed by an AP.

label	description	type
uid	the user ID managed by the AP	char
mig_id_spX	the migration ID of an SP, having columns against each SP, the name of the SP applied to the name of the column (The name of SP1's column would be "mig_id_sp1")	char
mig_id_ap	the migration ID issued by the AP	char
mig_comp	something representing whether migration has completed, 1 if completed, 0 otherwise	int (0 or 1)
srcIdP	the migration-source IdP	char
dstIdP	the migration-destination IdP	char
uid_idp	a pseudonym user ID received from an IdP	char
uid_num	the number linked to the user ID (uid), starts from 1, and updates one by one when registering a new user	int

[*1] https://github.com/nishioka-s273/saml_sp1
[*2] https://github.com/nishioka-s273/saml_ap

4.2 Data Transfer

In this section, we describe the steps to send and receive the data mentioned in Section 4.1. Here, two SPs federated by an IdP are referred to as SP1 and SP2. Then, refer the migration-source IdP as srcIdP, and the migration-destination IdP as dstIdP.

The following web pages (PHP files) are managed at an SP or an AP.

SP's pages

- index.php
  Redirect the user with the button "Log-in with IdP". If the user has already authenticated, it can be possible to initiate the move-out process of the dstIdP.
- start.php
  If the user has not authenticated yet, the SP redirects the user to the discovery service (where to select the IdP to log-in). The SP confirms whether the logged-in user is an existing user or a new user. For a new user, a new SP account (proper user ID) is issued.
  With the button "Register for Migration Service", the user can register for migration service, by sending SP's migration ID to the AP.
  With the button "Change the IdP for log-in", the user can initiate the move-out process after registering for the migration service.
- complete.php
  This page receives the migration ID issued by the SP, from the AP. The SP discovers the user account related to the received migration ID. The user account switches the associated IdP from the source IdP to the new IdP (destination IdP).

AP's pages

- index.php
  SPs send their migration IDs to this page and the AP keeps the proper IDs of the AP with the user's account. The AP generates and manages the user accounts. The registration process can be done here.
- migr.php
  The migration IDs of the AP are issued and provided to users. If the user has once completed the migration at an SP, the migration (move-in) process can be also initiated here.
- movin.php
  The first migration process can be initiated here. The migration ID of the user and other information are sent to movin2.php.
- movin2.php
  The AP gets the combination of the migration ID and the pseudonym user ID provided by the source IdP and the destination IdP. Then the AP sends the migration ID issued by an SP to the SP (SP1's migration ID is sent to SP1).

SP1 and SP2 have PHP files to perform the same process, and we refer to each of them like "sp1/XXX.php" for SP1's pages and "sp2/XXX.php" for SP2's pages. Also, files held by AP are

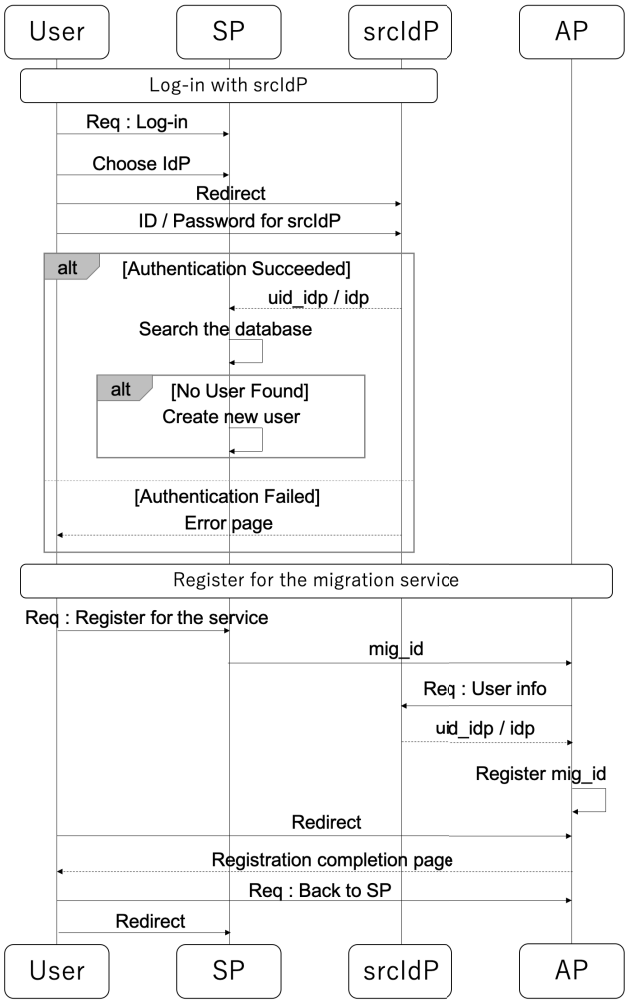

Fig. 3 Registration of migration ID (sequence diagram).

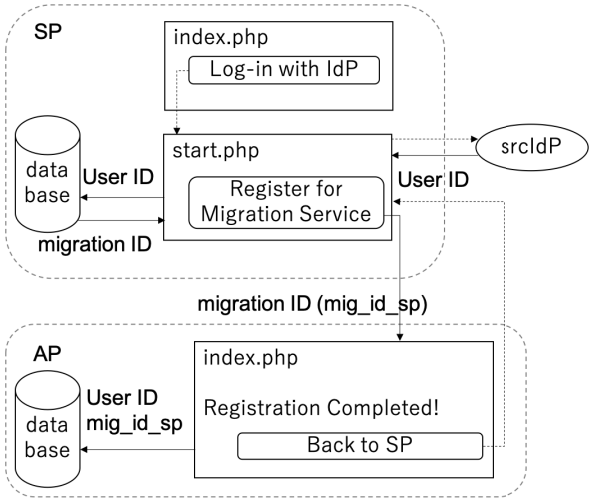

Fig. 4 Registration of migration ID.

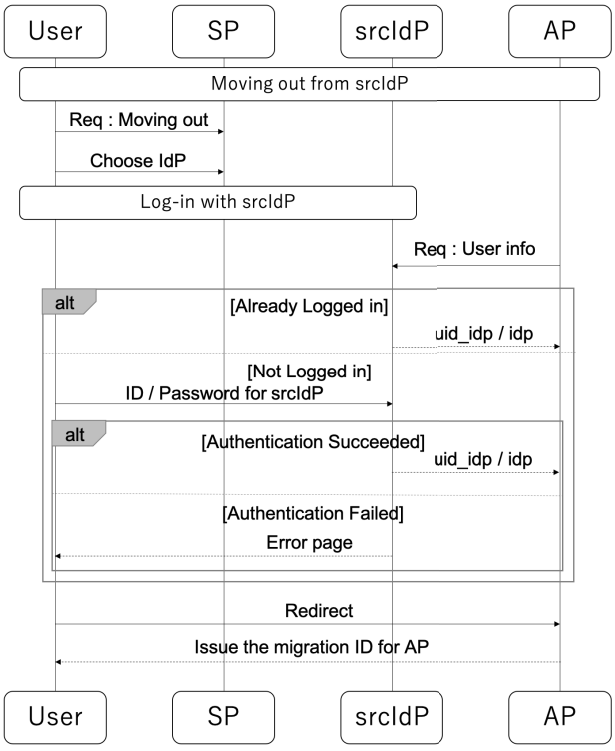

Fig. 5 Move-out from srcIdP (sequence diagram).

expressed like “ap/XXX.php”.

First, we will explain below the steps to issue the migration ID at the SP and register it to the AP.

[Registration of the migration ID]

(1) The user access to SP1’s website (sp1/index.php).

(2) Clicks the button, “Log-in with IdP”, and choose srcIdP to use to log in.

(3) Redirected to srcIdP’s log-in page, enter the srcIdP’s proper ID/password, and go back to SP1’s page. At this time, srcIdP sends uid_idp (pseudonym ID for the SP) and idp (who the IdP is) to the SP. The SP finds the user corresponding to uid_idp and if not found, register it as a new user.

(4) The user clicks the button, “Register for Migration Service” at SP1’s log-in completed screen (sp1/start.php) and is redirected to the AP’s website (ap/index.php). At the same time, the SP sends mig_id to the AP. The AP receives uid_idp (pseudonym ID for the AP) from srcIdP using SSO, links it to mig_id_sp (equivalent to mig_id managed at the SP), and saves them in its database.

(5) The user clicks the button, “Back to SP” at the AP’s registration complete page (ap/index.php) and completes the registration.

Figure 3 and **Fig. 4** illustrates this procedure.

The user also performs the above steps on SP2 to save the migration ID on the AP.

Next, we describe the procedure for migration. First, how to move out from srcIdP, which needs to be done while the srcIdP’s account is available.

The procedure is explained in **Fig. 5**, **Fig. 6**, and the steps below.

[Move-out from srcIdP]

(1) The user goes to the SP1’s website (sp1/index.php) and clicks the button, “Change the IdP for log-in”.

(2) Select srcIdP for log-in and redirected to its log-in page.

(3) The user enters his own set of ID (proper ID of the srcIdP)/password to log in.

(4) After logged in, the user will be redirected to the AP’s website (ap/migr.php). Then make a note the migration ID displayed on the page.

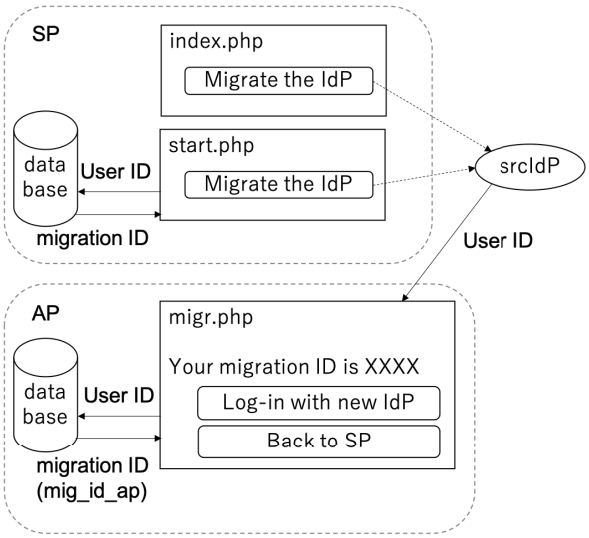

Fig. 6 Move-out from srcIdP.

Through the above procedure, the user can get the migration ID linked to the AP's account from srcIdP.

Finally, we will describe the steps to move into dstIdP. There are two possible scenarios. One is the case that the dstIdP's account is already available immediately after the user gets the migration ID. In this case, the move-in procedure can be accomplished by following steps.

[Move-in with dstIdP (Case 1)]

(1) The user clicks the button, "Log-in with New IdP", and select dstIdP at AP's page (ap/migr.php).
(2) Enter the ID (proper ID of the dstIdP)/password at the dstIdP's log-in page to log in.
(3) Redirected to the AP's page (ap/movin.php) and enter AP's migration ID wrote down earlier.
(4) The AP searches the account corresponding to the migration ID and if exists, completes migration by regarding the user currently logged in as the user linked to the migration ID. At this time, the AP changes the value of mig_comp from 0 to 1, which means the migration at the AP has been completed.
(5) Then, the confirmation screen of migration completion is displayed (ap/movin2.php). And the user selects the button, "Return to SP and Complete the Migration". At the same time, the AP sends this user's migration ID received from SP1 (mig_id_sp1).
(6) The user returns to SP1's page (sp1/complete.php) and completes the migration. SP1 links the uid_idp from dstIdP obtained by SSO and the account linked to migration ID then completes the migration.
(7) Next, the user accesses SP2's page (sp2/index.php) and logs in with dstIdP.
(8) Like SP1, select the button, "Change the IdP for log-in" (sp2/start.php).
(9) The user will be redirected to the AP (ap/migr.php) and will see that the migration has been completed. The AP asks if it is okay to complete the SP2 migration as well, then the user answers "Yes" to complete the migration.
(10) The user goes back to SP2 (sp2/complete.php) and completes the migration. At this time, the AP sends the migration ID received from SP2 (mig_id_sp2), and SP2 completes the migration based on the migration ID.

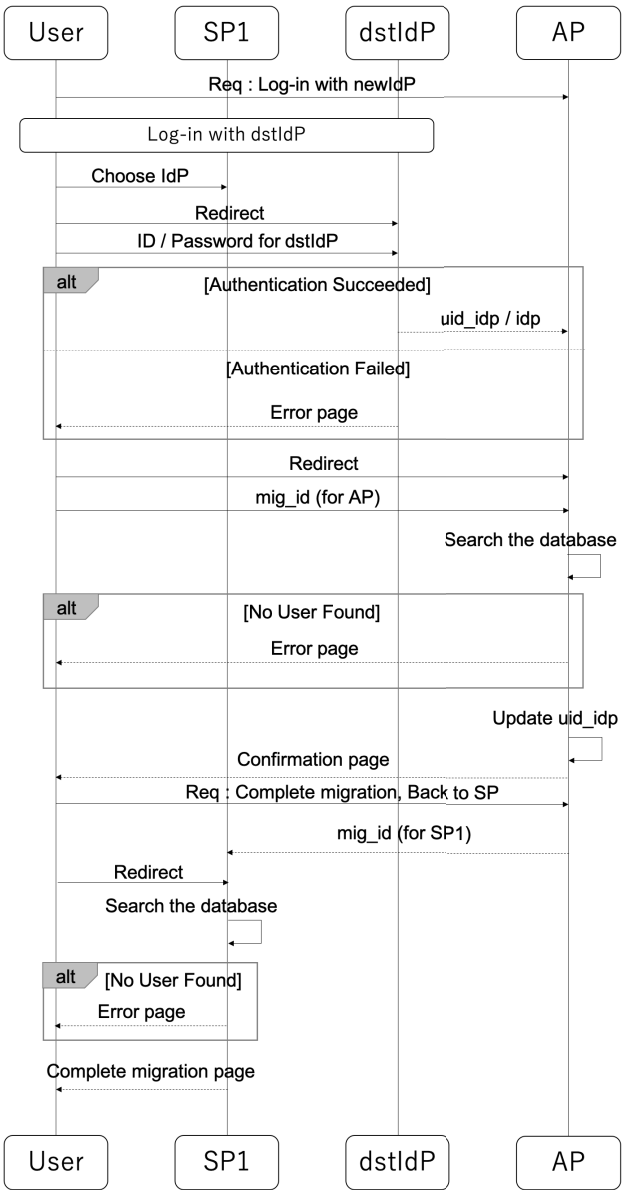

Fig. 7 Move-in with dstIdP (Case 1) for SP1 (sequence diagram).

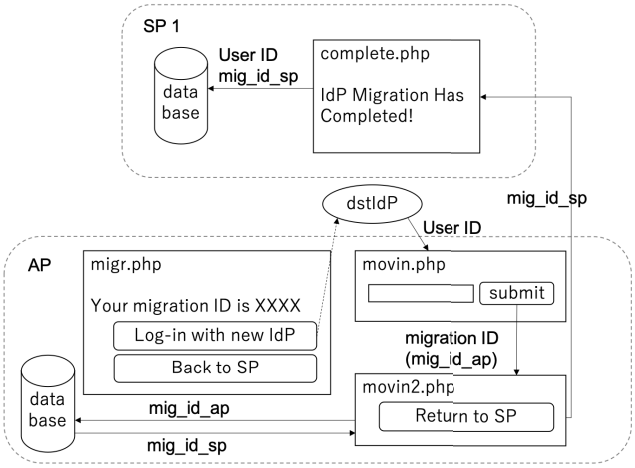

Fig. 8 Move-in with dstIdP (Case 1) for SP1.

This procedure would be like **Figs. 7**, **8**, **9** and **10**.

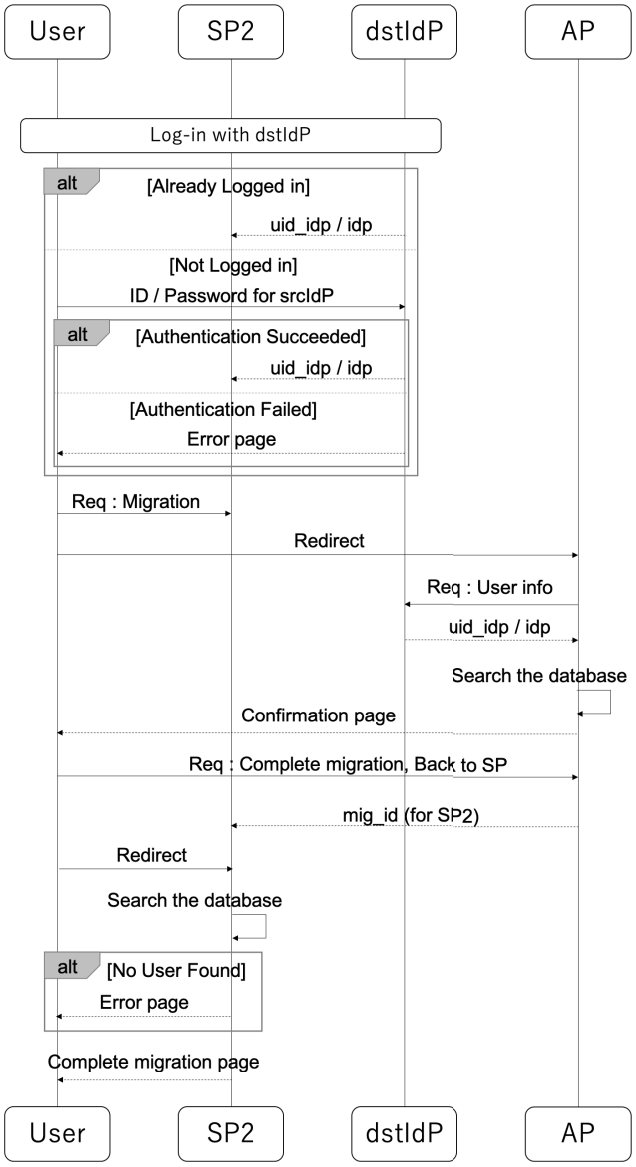

Fig. 9 Move-in with dstIdP (Case 1) for SP2 (sequence diagram).

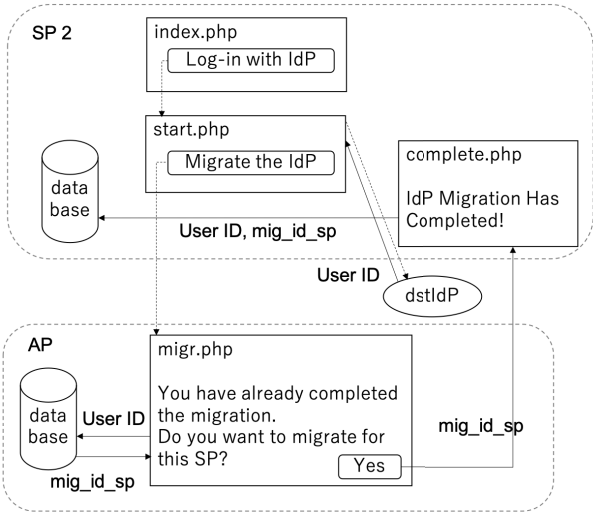

Fig. 10 Move-in with dstIdP (Case 1) for SP2.

The steps above are one scenario, and another scenario is as follows. The second case is that there is a period after the move-out procedure is completed before the dstIdP account becomes

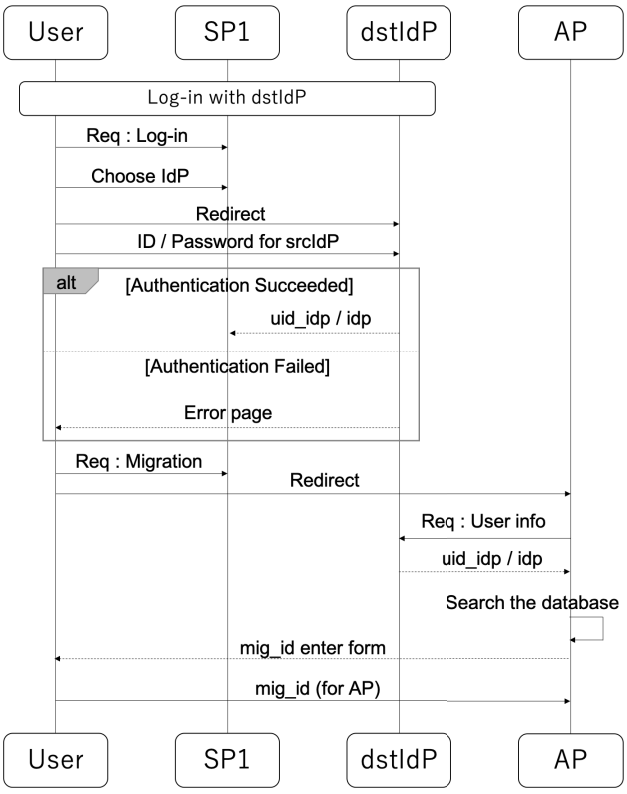

Fig. 11 Move-in with dstIdP (Case 2) (sequence diagram).

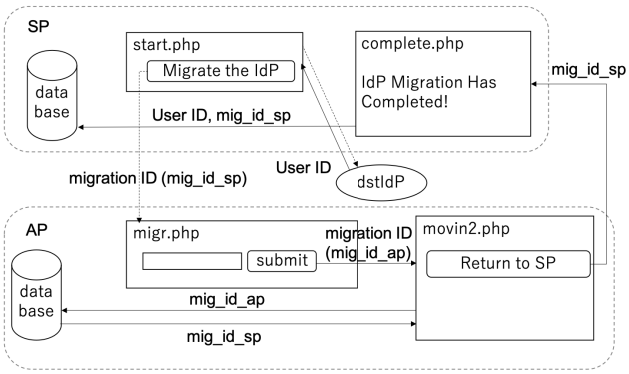

Fig. 12 Move-in with dstIdP (Case 2).

available. In this case, the move-in procedure would be realized as follows.

[Move-in with dstIdP (Case 2)]

(1) The user accesses SP1's page (sp1/start.php) and selects the button, "Change the IdP for log-in".

(2) When the user first logs in from a new IdP, the form to enter the migration ID is displayed (ap/migr.php). Then the user enters the migration ID.

Figures 11 and **12** illustrate this procedure. The steps after this are the same as Step (4) of Move-in with dstIdP (Case 1). As mentioned above, the user can migrate accounts of multiple SPs at once.

5. Discussion

5.1 Security Enhancement Based on the AP Trustability

The protocol proposed in Section 3.2 relies heavily on the AP, and some problems below will occur when the AP has malicious intent or is attacked by something malicious.

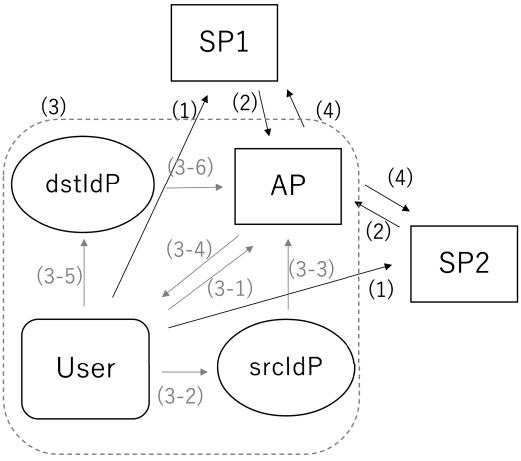

Fig. 13 The protocol that adds constraints.

- The AP can initiate the migration of accounts for users who do not want to migrate.
- The AP can link the user account to another existing user who is not linked to the migration ID.

For these issues, we propose some protocols with additional constraints. The constraints are as follows.

(A) The user must request to migrate the account explicitly for each SP

(B) When requesting the account migration, the user must specify the migration-source IdP and the migration-destination IdP

(C) The move-in procedure will not be accepted once the procedure using the user's migration ID has been completed

The protocol that adds these constraints is processed as follows.

(1) The user requests the migration explicitly for each SP logged in with srcIdP. At this time, the source IdP and the destination IdP are specified. (Constraints (A), (B))

(2) Each SP issues the migration ID and sends it to the AP. The AP saves these migration IDs.

(3) The user performs the migration procedure on a single SP with the AP regarding it as an SP.

(4) The AP switches the account links for each SP based on the migration information obtained in Step (3).

The overview of this protocol is expressed in **Fig. 13**. In Figure 13, the Step (3-1) to Step (3-6) is equivalent to what is the Step (1) to Step (6) in Fig. 1, Section 3.1.

In this protocol, although the user's workload increases compared to what is described in Section 3.2, each SP knows that the user wants to migrate his account (by the constraints (A)), and the source and destination IdP (by the constraints (B)). Thus, the AP can only switch the user to another user in the specified IdP. Also, by the constraint (C), the move-in procedure is not available after the illegal switch by the AP. Then the user cannot move in with the destination IdP, which reveals that the AP illegally switched the account. However, this illegal switching cannot be prevented.

Therefore, in case higher security is required, we consider setting a code number when requesting the migration in Step (1). For convenience, the user can decide whether or not to set the code number, for each SP. This code number cannot be recognized by the AP. Only the user and the SP know it. This code number is sent to the SP when logging in with dstIdP, and the migration is completed correctly by confirming the set of the migration ID and the code number by the SP.

In this case, The AP cannot switch the user's account to a different account because it cannot answer the correct code number to the SP. On the other hand, users need to remember the code number for each SP, which is a heavy burden for them. The goal of reducing the burden on users is not been fully achieved.

5.2 Tradeoff between Security and Usability

The proposed protocols described in Sections 3.2 and 5.1, are divided into three stages based on security level and user burden. Each has pros and cons, as follows:

Protocol 1 A protocol that almost trusts the AP and delegates the work (in Section 3.2)

Pros Users only need to perform a single migration procedure, which realizes high usability.

Cons The account migration may not be processed correctly because of unintentional initiation of the account migration or switching the account to a different user, by malicious AP.

Protocol 2 A protocol that combats malicious AP attacks by adding constraints (in Section 5.1)

Pros The AP cannot initiate the migration without the user's request, and we can notice unauthorized switchovers of accounts.

Cons Unable to prevent unauthorized switching of accounts, and also users have more to do than protocol 5.2.

Protocol 3 A protocol with high security with an additional constraint (in Section 5.1)

Pros In addition to protocol 2, unauthorized switching of accounts can be prevented.

Cons Users have to do much more than protocols 1 and protocol 2.

As mentioned above, the protocols proposed so far have both pros and cons. There is a trade-off between security and usability. Both can be optimized for the situation by allowing the user to choose one among the protocols based on their requirements.

5.3 Issues with the Proposed Protocol

Protocol 3 with the highest security described in Section 5.2 is where the user sets a code number to prevent unauthorized switching between users by the AP. However, the code number like 4-digit number can be broken by a brute force attack unless we place a limit to the number of attempts of incorrect input. Nevertheless, it is very difficult to notify the user of the input count exceeded, and reset the code number if there is an incorrect input count limit. In an SSO environment, the user's personal information generally belongs to the IdP and is not known to SPs. Therefore, SPs cannot retrieve the user's contact information such as email address [17].

Moreover, even if an SP could obtain the user's contact information, the email address may belong to the IdP's account that has already been revoked at the time of migration. As a countermeasure, there may be a way to get users to register the contact

information available after the account of the source IdP is revoked, when setting the code number. But this has problems that it contradicts our goal of reducing the user's burden, and the SP can identify the individual user contrary to Shibboleth's hallmark of user anonymity.

6. Conclusion

In this study, we have proposed an AP-based approach to the problem of difficulty in switching IdPs in privacy-aware authentication federation. The proposed protocol realizes migration of accounts for all federated SPs in a single migration procedure, assuming some trustability of the AP. Considering the user's privacy and convenience, we have added some constraints and have designed additional protocols according to the situations.

In this study we have considered an approach to utilize an AP, but some other solutions to the problem are possible. To reduce the hassle of remembering the migration ID for each SP, it is possible to implement an application that stores this information on one's device collectively [18]. Moreover, by implementing an authentication proxy on the SP front end to handle the complex exchanges in the authentication federation, it can collectively handle the changing of the IdP used for log-in, too [19]. Also, an approach using the threshold-based secret sharing method is considered as a solution to the dependency on a single IdP in the authentication cooperation [20].

As stated above, various approaches can be considered for the problems that have been handled in this study. In the future, we need to consider how to deal with the issues in the proposed protocols in this study which realize both security and usability.

Acknowledgments The authors are grateful to Professor Motonori Nakamura, Dr. Shuichi Miyazaki, and Dr. Daisuke Kotani of Kyoto University for their suggestions on this research.

References

[1] Sugiyama, K. and Akiyama, T.: Design and Implementation of Mutual Authentication based on PAKE in the Web SSO, The Institute of Electronics, Information and Communication Engineers (IEICE) Technical Report, Vol.112, No.489, pp.31–36 (2013).
[2] Sato, H. and Nishimura, T.: Federated Authentication in a Hierarchy of IdPs by Using Shibboleth, *2011 IEEE/IPSJ International Symposium on Applications and the Internet*, pp.327–332 (2011).
[3] Kakizaki, Y., Maeda, K. and Iwamura, K.: Identity Continuance in Single Sign-On with Authentication Server Failure, *2011 5th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pp.597–602 (2011).
[4] Shibboleth, available from ⟨https://www.shibboleth.net/⟩ (accessed 2020-04).
[5] OASIS: SAML Technical Overview (2008), available from ⟨http://www.oasis-open.org/committees/download.php/20645/sstc-saml-tech-overview-2%200-draft-10.pdf⟩.
[6] Ying, W.: Research on Multi-level Security of Shibboleth Authentication Mechanism, *2010 3rd International Symposium on Information Processing*, pp.450–453 (2010).
[7] Roh, J.-H. and Jin, S.-H.: Personal Information Management Using Attribute Provider, *2009 11th International Conference on Advanced Communication Technology*, Vol.2, pp.1293–1295 (2009).
[8] Oogami, W., Komura, T. and Okabe, Y.: Secure ID Transformation for Robust Pseudonymity against Backflow of Personal Information in SAML Federation, *2012 IEEE 36th Annual Computer Software and Applications Conference Workshops*, pp.64–69 (2012).
[9] Sato, H., Okabe, Y. and Nakamura, M.: User Identification of Pseudonyms without Identity Information Exposure - A Scenario in Access Federations, *Journal of Information Processing*, Vol.25, pp.788–795 (2017).
[10] Nakamura, M., Nishimura, T., Yamaji, K., Sato, H. and Okabe, Y.: Privacy Preserved Attribute Aggregation to Avoid Correlation ofUser Activities across Shibboleth SPs, *2013 IEEE 37th Annual Computer Software and Applications Conference Workshops*, pp.367–372 (2013).
[11] Hinton, H.M.: Method and System for Identity Provider Migration Using Federated Single-Sign-On Operation, US PATENT US7657639B2 (2006).
[12] Cameron, K., Nanda, A.K., Kwan, S.L.S. and Shwchuk, J.P.: Method and System for Integrating Multiple Identities, Identity Mechanisms and Identity Providers in a Single User Paradigm, US PATENT US7788729B2 (2005).
[13] Berbecaru, D. and Lioy, A.: On the Design, Implementation and Integration of an Attribute Provider in the Pan-European eID Infrastructure, *2016 IEEE Symposium on Computers and Communication (ISCC)*, pp.1263–1269 (2016).
[14] SimpleSAMLphp, available from ⟨https://simplesamlphp.org/⟩ (accessed 2020-04).
[15] Iso, Y., Konno, M., Take, Y. and Saito, T.: A Proposal and Implementation of SSO with SAML That Provides User Attribute Selectively, *Information Processing Society of Japan (IPSJ) 75th National Convention*, Vol.2013, No.1, pp.573–574 (2013).
[16] Nishimura, T., Nakamura, M., Otani, M., Yamaji, K. and Sonohara, N.: Goup Management System for Federated Identities with Flow Control of Membership Information by Subjects, *2012 IEEE 36th Annual Computer Software and Applications Conference Workshops*, pp.94–99 (2012).
[17] Watanebe, R. and Miyake, Y.: A User Account Management Method with Enhanced User Anonymity, *DICOMO 2011 Multimedia, Distributed, Cooperative, and Mobile Symposium*, Vol.2011, pp.351–357 (2011).
[18] Tasaki, Y., Yasui, H. and Yokoyama, T.: Realization of SSO method by reverse proxy systems without holding passwords, *Information Processing Society of Japan (IPSJ) 79th National Convention*, Vol.2017, No.1, pp.153–154 (2017).
[19] Flanagan, H.: Federation 2.0 - Arrival of the IdP-SP Proxy (2019), available from ⟨https://refeds.org/a/2128⟩.
[20] Ito, T., Kotani, D. and Okabe, Y.: A Threshold-Based Authentication System Which Provides Attriubutes Using Secret Sharing, *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, Vol.2, pp.739–737 (2017).

Satsuki Nishioka was born in 1996. She received her B.E. and M.Informatics degree from Kyoto University in 2019 and 2021, respectively, and is presently with KDDI Corp. Her research interests are Single Sign-On, identity management, identity federation.

Yasuo Okabe received M.E. from Department of Information Science, Kyoto University in 1988. From 1988 he was an Instructor of Faculty of Engineering, from 1994 he was an Associate Professor of Data Processing Center, and from 1998 he was an Associate Professor of Graduate School of Informatics, Kyoto University. He is now a Professor of Academic Center for Computing and Media Studies, Kyoto University. Ph.D. in Engineering. His current research interest includes Internet architecture, network security and distributed algorithms. He is a fellow of IPSJ and IEICE, and a member of ISCIE, JSSST, IEEE, and ACM.