7ZA-08

Shibboleth でのシングルサインオンにおける アカウント移動の一括制御

西岡　幸来†　　岡部　寿男‡
京都大学†

1. はじめに

今日，インターネットで様々なサービスを利用するにあたって個々人を識別するものとしてアカウントが広く用いられている．そして，複数のサービスに対してそれぞれアカウントを発行し，パスワードを記憶しておく煩わしさを緩和する技術の一つに，一つのアカウントにおける一回のログインで複数サービスを利用できるシングルサインオン(SSO)というものがある．SSO では複数のサービス・プロバイダ(Service Provider,SP)に対するユーザ認証を認証サーバ(Identity Provider,IdP)で一括して行っている．しかし，その IdP 自体を別の IdP に変更する場合，一回のユーザ認証で複数の SP に対して一括で引継処理を行うことができない．本研究では，主に SSO の実装に用いられているオープンソースプロジェクトである Shibboleth[1]を使用して，上記の問題に対し属性プロバイダ[2](Attribute Provider,AP)を利用した一括引継システムを提案する．

2. 問題提起

Shibboleth の SSO において SP から見たユーザは匿名となっており，そのユーザの属性(学生・教授など)は知ることができてもそのユーザが誰であるかは特定不可能であるべきである．IdP から認証情報を受け取る際にも同じユーザに対して SP ごとに異なる仮名 ID が渡される．そのため，SP 間で同じユーザを識別することはできず，SP 同士で引継情報を共有して作業の一括化を図ることは困難である．従って本研究では引継情報を仲介する機関として AP を採用している．また，ここでいう作業の一括化とは一回分の引継作業で複数の SP に対するアカウントの引継を完了する，ということを指す．

Collective Control of Accounts Translation at Single-Sign-On, Shibboleth
† SATSUKI NISHIOKA, Kyoto University
‡ YASUO OKABE, Kyoto University

3. 単一 SP での引継の流れ

本研究の目的である，複数の SP に対する引継作業を考える前に，ユーザが利用している SP(または引き継ぎたい SP)が 1 つである場合の流れについて考える．IdP1 から IdP2 への引継を行うことを考えるとき，以下のような手順での引継が可能である．

(1)ユーザは SP に対してアカウント引継を申請
(2)IdP1 でログイン
(3)IdP1 によるユーザ認証を完了
(4)SP はユーザに紐付いたアカウント情報とリンクさせた引継 ID を発行，ユーザに送信
(5)ユーザは IdP2 で再び SP にログイン
(6)ユーザ認証を完了後引継 ID を SP に渡し，SP は ID にリンクしたアカウントと現在ログインしたユーザを紐づけ，引継を完了

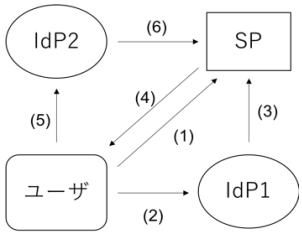

図 1：単一 SP での引継の流れ

以上のようにして 1 つの SP に対しての引継作業を行うことができると考えられる．この作業を複数の SP に対してユーザがそれぞれ実行すればすべての SP でアカウント引継ができるがユーザの負担が大きくなるため，以下 4 章からはこの作業を AP に代行してもらうということを考える．

4. 属性プロバイダを用いた一括引継の提案

3 章で述べた単一 SP での手続きの流れをもとに，SP が複数存在する場合についての引継の流れを以下のように提案する．

(1)ユーザは各 SP でアカウント作成時(または任意のタイミング)に引継 ID を発行，AP に保存
(2)AP を SP とみなして単一 SP での引継作業実行

Copyright ©2019 Information Processing Society of Japan.
All Rights Reserved.

(3)AP は(2)での移行情報をもとにそれぞれの SP に対してアカウント紐付けを切り替え

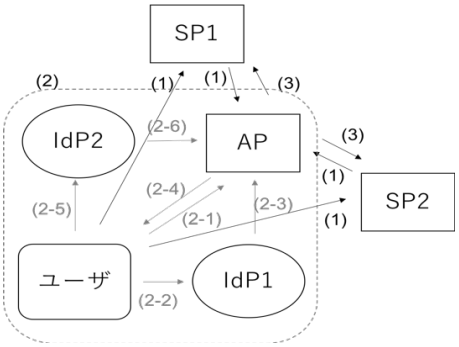

図 2：属性プロバイダを用いた一括引継の流れ

このシステムでは，ユーザは実質一回分の引継作業だけを行えば良く，作業効率が格段に上がると考えられる．しかし，引継 ID の保持や紐付けの切り替えは AP に一任されているため AP が悪意を持っていた場合，ユーザが引継を望んでいない時に勝手に引継作業を実行したり，異なるアカウントと紐付けを行ったりすることができるというリスクがある．そこで，いくつかの制約を追加したよりセキュアなシステムを以下 5 章で提案する．

5. よりセキュアなシステムの提案

4 章で提案したシステムにさらに以下の制約を加えたシステムを提案する．

(A) ユーザは各 SP に対して IdP を移行したい旨を明示的に申請しなければならない

(B)引継作業の申請時に移行元・移行先の IdP を指定しなければならない

(C)引継作業申請を受け付けてからそのユーザの引継 ID を用いた転入手続きが一度完了したらそれ以降の転入は受け付けない

以上の制約を追加したシステムの作業手順は以下のようになる．

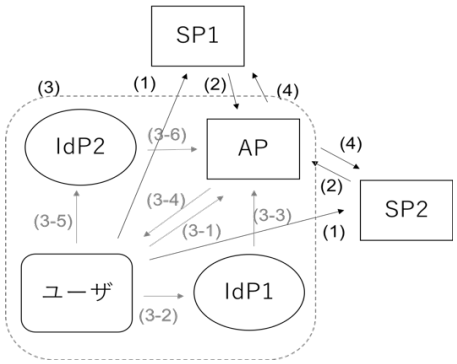

図 3：さらに制約を追加した引き継ぎの流れ

(1)ユーザは IdP1 でログインしている SP それぞれに対し明示的に引継を申請．このとき移行元・移行先の IdP を指定(※制約(A),(B))

(2)SP はそれぞれ引継 ID を発行して AP に送信，AP は ID を保存

(3)AP を SP とみなして単一 SP での引継作業実行

(4)AP は(3)での移行情報をもとにそれぞれの SP に対してアカウント紐付けを切り替え

このシステムでは 4 章で述べたものに比べてユーザの作業量は少し増えるが，それぞれの SP は制約(A)，(B)よりユーザが引継を行いたいということ・その移行元と移行先の IdP を知っているので AP にできることがある程度制限される．また，制約(C)によって，AP による異なるアカウント紐付けを発見することができるようになる．ただし，その防止はできないというリスクは背負うことになる．

そこで，さらにセキュリティを求められる場合のシステムとして，手順(1)で SP に対し引継申請を行う際に暗証番号を設定するということを考える．ただし利便性を考えて SP ごとに暗証番号を設定するかどうかは選択できるようにする．暗証番号は AP には知り得ずユーザと SP だけが知っているものとする．この暗証番号は IdP2 でログインする際に SP に渡し，SP が引継 ID と暗証番号のセットを確認することで引継を正しく完了する．

この場合，AP が異なるアカウント紐付けを行うことはほぼ不可能な状態にすることができる．しかしユーザの作業量がかなり増えてしまい研究の目的である，ユーザの負担を減らすということがあまり果たされていないといえる．

6. おわりに

以上の設計をもとに SimpleSAMLphp[3]を使用したシステムを現在開発中である．今後はシステムの実装とともに，セキュリティの質を落とさずにユーザの負担を減らすことのできるような方式をさらに検討していきたい．

参考文献

[1] Shibboleth. http://shibboleth.internet2.edu/ (accessed 2019.1)

[2] 千葉 昌幸，漆嶌 賢二，前田 陽二：属性情報プロバイダ：安全な個人属性の活用基盤の提言，情報処理学会論文誌，47(3)，676-685，2006-03-15.

[3] SimpleSAMLphp. https://simplesamlphp.org/ (accessed 2019.1)

Copyright ©2019 Information Processing Society of Japan.
All Rights Reserved.