「マルチメディア，分散，協調とモバイル
(DICOMO2013)シンポジウム」 平成25年7月

属性提供サーバに対してサービス提供サーバを秘匿する匿名化プロキシ

岡部寿男[†1] 佐藤周行[†2] 西村健[†3] 山地一禎[†3] 中村素典[†3]

SAML に代表される ID 連携プラットフォームでは、個々のユーザの認証を行った後、そのユーザに関する属性情報に基づいて認可判断を行い、サービスを提供する形態をとる。本研究では、属性提供サーバとサービス提供サーバを相互に秘匿するために匿名化プロキシを利用することを提案し、そのプロトコルを設計・実装した。属性情報を匿名化プロキシに対して暗号化するために、DH 鍵交換を用いる方式とカスケード型に二つのプロキシを用いる方式の二方式を提案し比較を行った。

Pseudonymized Proxies Which Conceal an Attribute Provider and a Service Providers Each Other

YASUO OKABE[†1] HIROYUKI SATO[†2] TAKESHI NISHIMURA[†3]
KAZUTSUNA YAMAJI[†3] MOTONORI NAKAMURA[†3]

1. はじめに

これまで一つの大学や企業などの内部に閉じて利用されてきた認証基盤を、組織の壁を越えた社会的な ID 連携プラットフォームとして実用化する動きが始まっている。このような認証基盤の社会基盤への展開の試みは、近年学術分野において活発に始められており、国を単位として欧米を中心に 30 を超える国々で構築が進められている。このような ID 連携基盤は認証フェデレーションあるいは単にフェデレーションと呼ばれる。日本においては、国立情報学研究所が中心となって、SAML (Security Assertion Markup Language) [1]に基づく国際的な学術系フェデレーションとして学術認証フェデレーション「学認」を構築し運用している[2]。

ID 連携プラットフォームを社会において本格的に実用化するためには､匿名性を悪用した不正利用を排除しつつ、ID 連携プラットフォーム上でやりとりされる個人情報の流通を必要最小限に制御しプライバシーを保護することが求められる。SAML に代表される ID 連携プラットフォームでは、個々のユーザの認証を行った後、そのユーザに関する属性情報に基づいて認可判断を行い、サービスを提供する形態をとる。誰がどのようなサービスにアクセスしたかという情報はプライバシー情報として保護されるべきものであり、また属性情報には個人情報が含まれる。これらは本来必要としない者に対して提供されるべきではないし、関係者に対しても必要以上に開示されるべきではない。しかしながら、現状は、まずは認証連携を機能させ相互接続性を確保することが優先され、個人が特定可能な属性情報の安直な利用を前提とした仕組みが前提となっていることが多い。個人情報の保護に関する対応は、個人情報の送信の際のユーザ同意をどのように取得するか、というような個人情報の開示に関する法制度等要求に適合させることのみを考慮したものにとどまっており、プライバシー保護に関する考慮が十分ではない。

本研究では、これらの属性情報を適切に暗号化、仮名化（スードニマイズ, pseudonymize）した上で流通させることにより、ユーザがどのサービスにアクセスしたかが ID 管理側からは分からないようにすると同時に、どのユーザがアクセスしてきたかがサービス提供側には分からないようにするプライバシー保護機能を、認証の信頼性を落とすことなく実現することで、属性提供サーバにおけるプライバシー保護の課題を解決することを目的とする。ここで考える属性提供サーバ（Attribute Provider, AP）は、一般には ID プロバイダ（Identity Provider, IdP,）、サービス提供サーバ（Service Provider, SP）のいずれからも独立して、個々のユーザの属性を格納し、IdP におけるユーザ認証を経て当該ユーザの属性を SP へ提供するものであるが、本論文では、単純化のため、Shibboleth のアーキテクチャで典型的な､ID プロバイダが属性提供サーバを兼ねる場合について扱う。ユーザが認証を経て属性を提供して SP にアクセスする際に、ID プロバイダ（＝属性提供サーバ)、サービス提供サーバのいずれにおいても、ユーザが誰であるかと、

†1 京都大学
Kyoto University
†2 東京大学
The University of Tokyo
†3 国立情報学研究所
National Institute of Informatics

ユーザがどのようなサービスを受けたかの二つを結び付けることができないようにすることが望まれる。本研究ではその要件をさらに厳格化し、IdP にとってはユーザがどの SP でサービスを受けたかを知りえず、SP にとってはサービスを提供しているユーザがどの IdP に所属するかを知ることができないようにする。具体的には、ユーザが認証を経てサービス提供サーバ(SP)にアクセスする際に、どのサービス提供サーバにアクセスしたかというプライバシー情報を ID 管理側(IdP)すなわち属性提供サーバ(AP)に秘匿し、同時に SP 側でも当該ユーザがどの IdP で認証を受けたかが分からないようにするための仕組みを実装する。従来からの対面でのサービス提供において、大学が大学連合名で発行する割引券を学生が窓口で提示して学生割引を受ける場合に、大学はどのようなサービスに対して当該ユーザが学割サービスを受けたかについて関知せず、またサービスを提供する側も当該ユーザがどの大学に所属するかは関知しない。このような仕組みをオンライン上で実現することに相当する。

以上の機能を持つ通信プロトコルの設計・実装と、認証連携機構の設計・実装を行った。IdP が、ユーザ認証を経て属性提供サーバとして発行する属性アサーションを SP に伝達する際に、具体的な SP の存在を IdP に見せないようにすると同時に、SP に対しても IdP の存在を秘匿するために、属性アサーションを中継するプロキシ（以下、匿名化認証連携プロキシ）を導入することでプライバシー保護を実現した。ここで、属性アサーションに記載された属性は匿名化認証連携プロキシに対しては秘匿されるべきことから、IdP と SP の間で、一対一の直接の信頼関係を結ぶことなしに暗号化を行う必要がある。そこで、SP として任意のものを許すわけではなく、事前に信頼関係を構築している SP 群のうちの何れかであることが前提とし、前提条件の異なる 2 通りのシナリオに対応する匿名化認証連携プロキシの新たなアーキテクチャを設計し実装した。

以下、2 章では SAML と Shibboleth について述べる。3 章では、属性提供サーバとサービス提供サーバを相互に隠蔽することの必要性とそれを実現するための匿名化認証プロキシについて説明する。4 章で匿名化認証プロキシのプロトコルの提案と動作の詳細の説明を行い、5 章では設計したプロトコルについて考察を行う。

2. SAML と Shibboleth

SAML (Security Assertion Markup Language)とは、Web サービスに関する標準化組織である OASIS [3]によって策定された、認証情報を表現するための XML 仕様である。Web サイトや Web サービスの間で、ユーザの認証や属性、認可に関する情報を、SAML で記述されたアサーション(assertion)の形で交換することで、一度の認証で複数のサービスが利用できるシングルサインオン(SSO：Single Sign-On)が実現される。認証情報の交換方法は SAML プロトコルとしてまとめられており、メッセージの送受信には HTTP もしくは SOAP が使われる。

Shibboleth [4]は、米国 Internet2 が主導する学術系の ID 連携基盤のアーキテクチャとそのオープンソースによる実装を創出するプロジェクトである。アーキテクチャおよびソフトウェア実装にも Shibboleth の名称が用いられる。Shibboleth のアーキテクチャは SAML に基づきそのサブセットとして IdP (Identity Provider)と SP (Service Provider)の間で認証と属性交換、認可が行われる。

Shibboleth ではプライバシーの保護に注意が払われており、ユーザ個人を特定する IdP 側のアカウント名や実名、電子メールアドレス等は、真に必要とされる場合以外は SP 側へ伝えないことを原則としている。ユーザの匿名性を担保しつつ複数回のログインで同一ユーザであることを紐づけ、インシデント発生時にのみ追跡を可能とするために、ePTID (eduPersonTargettedID)と呼ばれる仮名 ID が用いられる。ePTID は、各 IdP において SP ごとに異なるものが用いられ、SP 同士が結託することによるいわゆる名寄せのリスクを排除している。大神らは、仮名 ID が用いられている場合でも、SP 側が管理するログが IdP 側に逆向きに漏洩した場合にはプライバシー上の問題が生じることを指摘し、プロキシ IdP において仮名 ID を変換することによる解決法を提案している[5].

表 1 DH 鍵交換型プロキシのプロトコルにおける記号

p	SP の生成する乱数。SP のみが知り得る
q	IdP の生成する乱数。IdP のみが知り得る
A	開示される属性情報
g	十分大きい素数を法とする有限体の原始元。フェデレーション内で共有
$K=g^{pg}$	Diffie-Hellman 鍵交換で交換される共有鍵
$e_k(A)$	A を k で暗号化したもの
$e_s(p)$	SP の秘密 p を、SP だけが解ける暗号 e_s で暗号化したもの

Shibboleth に基づく認証連携で IdP が持つユーザの属性が SP に開示される際、個人情報に該当するものについては原則として本人同意が必要となる。Orawiwattanakul らは、IdP 側で開示する属性をユーザに提示して個別に同意を取る仕組みを提案しており[6]、uApprove.jp として実装されて学認でも利用されている。

3. 問題の定義

Shibboleth による認証連携では、2 章に述べたように、ePTID のような仮名を用いることで、IdP 側ではユーザが SP でどのようなサービスを受けているかを知りえず、SP 側ではサービスを提供しているユーザが誰であるのかを知りえないことを同時に実現できる仕組みを提供している。これはたとえば、図書館の電子ジャーナルサービスにおいてある研究者がどのような文献を閲覧したかという、プライバシー上あるいは知財上保護されるべき情報が秘匿できるという点で重要である。

しかし、IdP の側ではユーザがどの SP でサービスを受けたかまではわかるし、SP の側ではユーザがどの IdP に所属しているかまではわかってしまう。IdP に登録されているユーザの範囲あるいは SP が提供するサービスの性質によっては、それすら秘匿したいことも珍しくない。たとえば転職の仲介サービスを行う SP を考えると、そのような SP にアクセスしていることを IdP の管理者である当該ユーザが所属する企業の人事担当者が知りえることは好ましくない。

この問題は、IdP と SP の間に、認証連携を中継する匿名化認証連携プロキシを挟むことで緩和できる。これにより、IdP はユーザがどの SP でサービスを受けていることを知りえず、SP はユーザがどの IdP に所属するかを知りえないことを同時に実現できる。匿名化認証連携プロキシではどの IdP のユーザがどの SP でサービスを受けているかまではわかるが、ユーザが誰であるかや SP でどのようなサービスを受けているかまではわからない。

しかしながら、IdP から SP に、ユーザに関する属性情報が渡される場合には、新たな課題が生じる。属性情報は、匿名化認証連携プロキシに対しては秘匿されるべきであり、IdP で暗号化して送出され SP で復号することが求められる。しかるに、IdP と SP は互いに相手のことを知りえないことが前提であり、IdP と SP とで事前に共通鍵を共有したり、公開鍵暗号のように相手の秘密鍵を用いて暗号化するようなことはできない。そのような状況でどのように暗号化に必要な鍵を交換するかが課題となる。

4. 属性提供サーバとサービス提供サーバを相互に隠蔽する機構

3章で述べた課題に対する解として、SAMLを通信プロトコルに採用し、Shibbolethの IdPおよびSPでフェデレーションが構築されているという条件のもとで、IdPからSPへ開示される属性情報を中継するプロキシシステムであって、IdPとSPとが互いに秘匿されるとともに、プロキシに対しては属性情報の値が秘匿されているようなプロトコルを設計した。

設計したプロトコルは、IdPとSPとの間で、IdPが送出した属性情報がどのSPに開示されるかがIdPに対して秘匿される（サービスの匿名性）とともに、属性情報を受け取ったSPに対しても、その情報がどのIdPから送出されたものであるかが秘匿される（ユーザの匿名性）ようなプロキシであって、属性情報を中継するプロキシにおいては中継される属性情報が暗号化により秘匿されているものである。

これを実現するプロキシとして、以下の二方式を提案し、そのプロトコルを設計・開発、ソフトウェアとして実装した。

- DH鍵交換型プロキシ
- カスケード型プロキシ

ここでDH鍵交換型プロキシとは、IdPとSPがDiffie-Hellman鍵交換プロトコルにより動的に秘密鍵を共有し、それを用いてプロキシに秘匿した形で属性情報を送る方式である。これに対しカスケード型プロキシはIdP側プロキシとSP側プロキシの二つの独立に運用されるプロキシからなり、IdP側プロキシとSP、SP側プロキシとIdPがそれぞれ公開鍵暗号を用いて動的に秘密鍵を共有し、それによりIdP側プロキシ、SP側プロキシのいずれに対しても秘匿された形で属性情報が送られる。

4.1 DH 鍵交換型プロキシ

DH鍵交換型プロキシのプロトコルにおいては、IdPとSPの間に属性情報を中継するプロキシサーバ（以下Proxyと表記する）を置き、SPとIdP間では Diffie-Hellman 鍵交換アルゴリズムを用いることで、情報の秘匿を行う。

DH鍵交換型プロキシのプロトコルを図1に示す。図1の記号の意味は表1のとおりである。

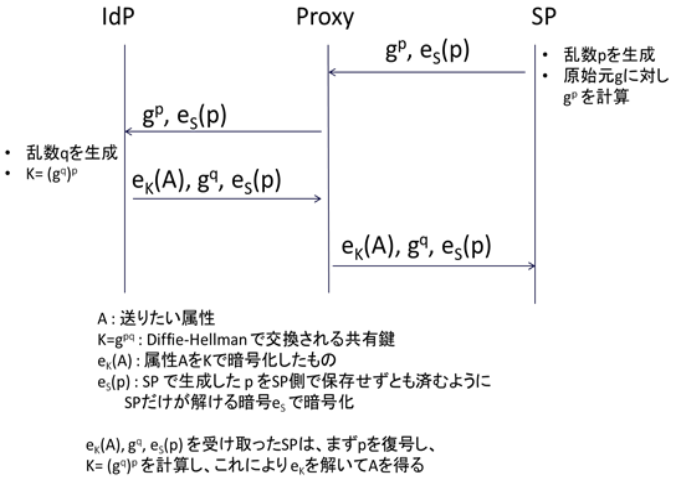

図 1. DH 鍵交換型プロキシのプロトコル

表 2 カスケード型プロキシのプロトコルにおける記号

K_1	SP, PxI において有効な共有鍵
K_2	PxS, IdP において有効な共有鍵
$e_{SP}(A)$	A を SP のみが復号できる鍵で暗号化したもの。e_{IdP}, e_{PxI}, e_{PxS} も同様
$K_1 \oplus K_2$	K_1と K_2のビットワイズ排他論理和。 $K_1 \oplus K_2 = K_2 \oplus K_1$と $A \oplus K \oplus K = A$ が成り立つ

Proxyは、SPに対してはIdPとして動作し、IdPに対してはSPとして動作することで、IdPに対して属性情報の要求元を秘匿すると同時に、SPに対して認証情報の提供元を秘匿する。ProxyはSAMLプロトコルで利用されるHTTP Redirectによって、SPからIdPへの認証要求を送信し、本来のSAMLプロトコルにより規定されたHTTP Redirect、POST、Artifact等によってIdPからSPへの認証結果と属性情報を返送する。

認証要求と認証結果の往復の際、SPは同時に、2つの情報を送る。

g^p ... Diffie-Hellman 鍵交換における共有鍵の片側（SP 側）

$e_s(p)$ ... Diffie-Hellman 鍵交換における SP 側の秘密を SP のみが復号できる鍵で暗号化したもの

これらを受け取った IdP は、その場で共有鍵のもう片側である q を生成し、共有鍵である $K=g^{pq}$ を生成する。この鍵を用いて、IdPは新たに以下の情報を生成し、SPへ返送する。

g^q ... Diffie-Hellman 鍵交換における共有鍵の片側（IdP 側）

$e_k(A)$ ... 属性情報 A を共有鍵 $K=g^{pq}$ によって暗号化したもの

$e_s(p)$ ... Diffie-Hellman 鍵交換における SP 側の秘密。SP から送信されたものをそのまま返信する

SPは自分が送信し、そのままの状態で返ってきた $e_s(p)$ からpを復号し、g^qを復元します。この値と新たにIdPから送られたg^pから共有鍵 $K=g^{pq}$ を取得、それによって暗号化された$e_k(A)$を復号し、最終的にAを得る。

なお、この過程においては、SPは自分自身の中にこの認証要求に関する追加の状態を保存しないことを仮定し、SP側で後に必要になる情報をSP自身が持つ対象鍵で暗号化して要求と応答に埋め込む形としている。SPがこの認証プロトコルの過程で自身の秘密であるpを覚えておける場合には$e_s(p)$は省略でき、プロトコルも簡略化される。具体的には、Key-Value Store型のデータベースがSP側で提供されている場合、SAMLプロトコルのIDを元にしたkeyとpをvalueにしたデータをデータベースへ保存することで、属性情報が到達した際にpを取得することができるため、$e_s(p)$をProxyおよびIdPへ送信してそれが認証後に返すよう要求する必要はない。なお、同様の省略は後述するカスケード型プロトコルにおいても可能である。

DH鍵交換型プロキシにおいては、以下の性質が保証される。

- SAMLにおける基本的な構成と同様に、SPとIdPはフェデレーション内で属性情報を受け渡すことができる
- IdPは認証と属性情報を要求したSPを知ることはできない
- SPは属性情報を返答したIdPを知ることができない
- Proxyは属性情報の中身を知ることができない(Diffie-Hellman鍵交換の性質による)

4.2 カスケード型プロキシ

カスケード型プロキシのプロトコルは、プロキシを 2 つ (PxS, PxI) 直列に配置して共有される秘密を分割し、IdP と PxS、SP と PxI とでそれぞれ暗号鍵を共有して二重に暗号化することで、プライバシー保護の強化を図ったものである。

カスケード型プロキシのプロトコルを図 2 に示す。図 2 の記号の意味は表 2 のとおりである。

本プロトコルにおける2つのProxy PxSおよびPxIの基本動作はDH鍵交換方式と同様であり、SP側からはIdPとして動作し、IdP側からはSPとして動作する。

SPが認証と属性情報の要求を送信する際、SPは乱数列K_1を生成し、それをSP自身とPxIのみが読めるように暗号化する ($e_{PxI}(K_1)$, $e_{PxS}(K_1)$が該当)。さらにSP自身の署名を付加してPxSに送出する。

PxSは乱数列K_2を生成し、それをPxS自身とIdPのみが読めるように暗号化しPxS自身の署名を付加した上で、SPから送られてきた暗号化された情報に加えてPxIへ送信する。

往路においては、PxIはSPから送られてきた情報の署名を検証した上で取り除き、PxSから送られてきた追加の情報と共に、認証要求をIdPへそのまま送る。

認証後、IdPは認証要求に付されたPxSの署名を確認の上、PxSから与えられるK_2とビット単位の排他的論理和演算を行うことにより属性値Aを暗号化し ($A \oplus K_2$)、PxIへ送る。PxIはK_1を復号することでPxIから送られてきた($A \oplus K_2$)を更に暗号化する ($((A \oplus K_2) \oplus K_1) = A \oplus K_2 \oplus K_1$)。

PxSにおいて、PxSはK_2を知ることができるため $A \oplus K_2 \oplus K_1$に対してK_2を適用し、ビット単位の排他的論理和の性質から$((A \oplus K_2 \oplus K_1) \oplus K_2) = A \oplus K_1$を得る。

SPにおいてはK_1を得られることから、$A \oplus K_1 \oplus K_1 = A$が得られる。

DH鍵交換型プロキシのプロトコル同様、SP、 PxS、 PxIにおいて当該セッション中に状態を保存できる場合、上記のフローで自分自身の秘密鍵向けに向けに暗号化された各種情報は省略できる。

カスケード型プロキシにおいては、以下の性質が保証される。

- SAML同様、SPとIdPはフェデレーション内で属性情報を受け渡すことができる
- IdPは認証と属性情報を要求したSPを知ることはできない
- SPは属性情報を返答したIdPを知ることができない
- PxS、PxIは属性情報の中身を知ることができない
- PxS、PxIのどちらか片方において中間者攻撃を受けた際に、IdPとSP、および残った側のProxyはその事実を知ることができる

4.3 実装

以上の設計仕様に基づくプロキシを、既存のSAML実装と整合する形で実装した。実装を行う上で利用したソフトウェアは下記の通りである。

- Ubuntu 12.04.1 LTS (Linux ディストリビューション)
- Apache 2.2.22-1ubuntu1 (Webサーバ)
- Tomcat 6.0.35-1ubuntu3.1 (Javaサーブレットコンテナ。Shibboleth IdPの動作に必要)
- Shibboleth IdP 2.3.8 (Tomcatと協調動作するSAML IdPのJava実装)
- Shibboleth SP 2.4.3 (Apache と協調動作するSAML SPの C++ 実装)
- OpenSSL 1.0.1-4ubuntu5.5 (セキュリティ機能を提供するライブラリ。秘密鍵を利用する際に用いる)
- SimpleSAMLphp 1.10.0 (Proxy 機能を提供するSAML認証の php 実装)
- Firefox 19.0+build1-0ubuntu0.12.04.1 (ブラウザ)

開発したソフトウェアは、商用のクラウド環境にインストールしてデモ環境を構築し、シナリオ通りに動作することを検証した。

5. 考察

DH 鍵交換型プロキシ、カスケード型プロキシとも、通信路における盗聴や中間者攻撃は、IdP と Proxy、Proxy と SP の間の通信を SSL により相互認証を行った上で暗号化することで防止できる。ただし、DH 鍵交換型プロキシでは、Proxy が悪意を持った第三者によって操作された際などにおいて、Diffie-Hellman 鍵交換に対する中間者攻撃により、属性情報の盗聴や改竄が生じる可能性は排除できない。SP と IdP の双方で、共有鍵である $K=g^{pg}$ のハッシュ値をログとして記録しておくことにより、万一中間者攻撃が行われた場合でも疑義が生じた時点でそのことを検出できる点で、一定の抑止力にはなるが、中間者攻撃のリスクを完全に排除するためには、カスケード型プロトコルを用いるのが適当であると考えられる。

カスケード型プロキシのプロトコルでは、プロキシを 2 つ直列に配置して共有される秘密を分割し、二重に暗号化されているので、DH 鍵交換型プロキシにおいては回避できない 1 箇所への中間者攻撃の検知が実現されている。

カスケード型プロキシを運用する上での課題として、信頼性と独立性が担保される二か所に分散してプロキシを配置しなければならないことが挙げられる。たとえば大学と商用サービス提供者との認証連携において、大学連合的な機関が PxI を、商用サービス事業者の組合のような機関が PxS を運用するようなことで、実際のフェデレーションに親和する形での運用が可能ではないかと考えている。

提案方式を運用するにあたって、IdP と SP におけるユーザの紐づけに ePTID のような仮名 ID を用いる場合には、IdP 側で SP が IdP を特定しにくい仮名 ID を付ける配慮が要る。またその場合でも、文献[5]で指摘されている個人情報の逆流出の問題は生じうる。本提案方式に文献[5]で提案されている仮名 ID の変換を併用することは容易であり、これによりさらなるプライバシー保護の強化が実現できる。

匿名化認証連携プロキシを運用する上では、IdP 側での属性開示に伴うユーザの本人同意を、開示先の SP を秘匿した状態で行わなければならない点には注意が必要である。このような状況では、開示先が全く不明な DH 鍵交換型プロキシに比べ、開示先をある程度限定できるカスケード型プロキシの方が運用しやすい。いずれにせよ開示を承認することの意味をユーザにわかりやすい形で提示することが求められることから、それらを 2 章で述べた uApprove.jp 上に実現するなどを検討していきたい。

6. おわりに

本研究では、目的とした属性提供サーバに対しサービス提供サーバを隠蔽する機構について、プロキシサーバにより実現することを考案してプロトコルを設計し、オープンソースとして公開可能なソフトウェアとして実装した。これにより、日本の個人情報保護制度を踏まえ、プライバシー保護意識の比較的高い応用分野にも適用可能なプライバシーの保護・利用シナリオの実現ができるようになった。

今後は、開発成果のソフトウェアのパッケージングその他、オープンソースとして公開するための作業や、設計したプロトコルの標準化提案等を精力的に行い、成果を社会に還元する作業を行っていきたい。

謝辞　匿名化プロキシの要件について議論いただいた崎村夏彦氏をはじめ野村総研（株）の諸氏に深謝する。ソフトウェアの実装に協力いただいた（株）mohka の各位に感謝する。なお研究は、総務省「戦略的国際連携型研究開発推進事業」（平成 24 年度，情報セキュリティに関する研究開発課題の委託）による支援を受けて「情報流通連携のためのオープンな ID 連携プラットフォームにおけるプライバシー保護機能の高度化の研究開発」として実施したものである．

参考文献

1)　S. Cantor, J. Kemp, R. Philpott, and E. Maler ed., "Security Assertion Markup Language (SAML) V2.0," http://saml.xml.org/ saml-specifications, March 2005.
2)　西村健，中村素典，山地一禎，大谷誠，岡部寿男，曽根原登：日本における学術認証フェデレーションとその役割および効果，信学技法, Vol. 111 No. 375, IA2011-55 pp.5-8, 2012.
3)　OASIS: Advancing open standards for the information security, https://www.oasis-open.org/, last visited May 15, 2013.
4)　Shibboleth Consortium, http://shibboleth.net/, last visited Apr. 1, 2013.
5)　Wataru Oogami, Takaaki Komura, Yasuo Okabe, Secure ID Transformation for Robust Pseudonymity against Backflow of Personal Information in SAML Federation, Proc. 2012 IEEE 36th International Conference on Computer Software and Applications Workshops (6th IEEE International Workshop on Middleware Architecture in the Internet (MidArch2012)), pp.64-69, July 2012.
6)　Orawiwattanakul, T., Yamaji, K., Nakamura, M., Kataoka, T., Sonehara, N., User Consent Acquisition System for Japanese Shibboleth-based Academic Federation (GakuNin), International Journal of Grid and Utility Computing (IJGUC)　2(4) 284-294, Nov.2011.

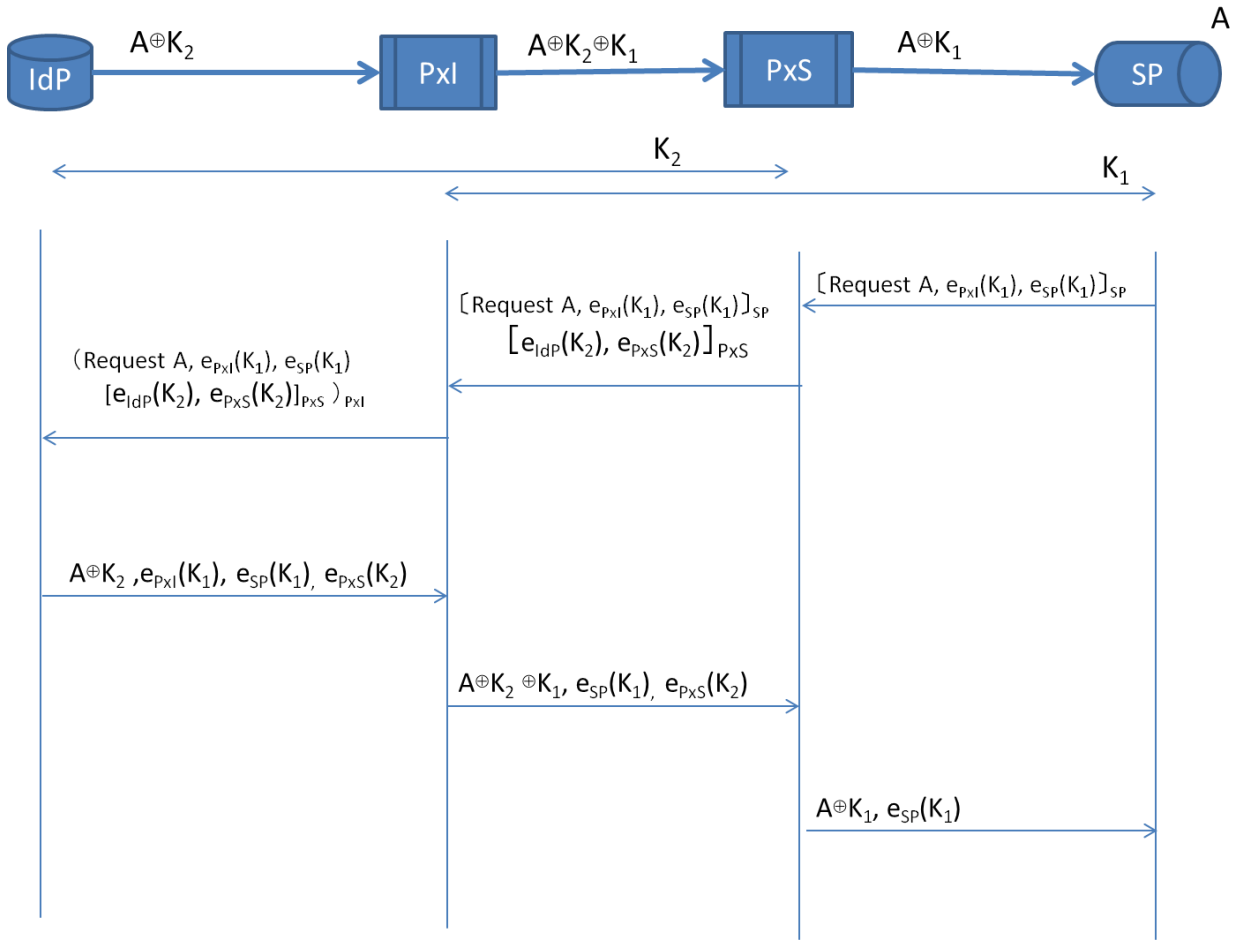

図 2. カスケード型プロキシのプロトコル