認証連携における仮名性を担保したユーザ同一性の確認

佐藤周行[1] 岡部寿男[2] 中村素典[3]

概要： Shibboleth/SAML を通信プロトコルに採用し、ID 管理サーバ(IdP)とサービス提供サーバ(SP)とが認証連携プロキシを介してフェデレーションを構築している状況において、異なる認証連携プロキシを介して認証連携しているユーザの同一性を、その仮名性を担保しつつ、SP が確認できるようにするための仕組みを設計・開発した。

User Identification in Authentication Federation with Pseudonimity

HIROYUKI SATO[1] YASUO OKABE[2] MOTONORI NAKAMURA[3]

1. はじめに

これまで一つの大学や企業など組織の内部に閉じて利用されてきた認証基盤をオープン化し、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ための取り組みが進んでいる。このような認証基盤を社会基盤へと展開させる試みは、近年学術分野において積極的に行われており、国を単位として欧米を中心にすでに 50 を超える国々で構築が進められている[1]。このような組織を跨がる 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。我が国においては、国立情報学研究所が中心となり、SAML (Security Assertion Markup Language) [2]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」(GakuNin)を構築し運用している[3]。

認証フェデレーションにおいては、その上でやりとりされる個人情報の流通を制御しプライバシーを保護することが可能な技術が確立されその仕様(API)が標準化されていることは当然として、授受される個人情報の扱いについて事前に合意した上で情報を提供するオープンな仕組みとその可視化も重要になる。認証と認可、さらにサービス提供のためにサービス提供者（以下 SP）に提供される（個人情報を含む）属性情報について、プライバシー保護の観点から本人同意が前提となるのは当然として、その開示範囲が必要最小限となるように配慮すべきである。同時に、誰がどのようなサービスにアクセスしたかという情報もプライバシー情報として保護されるべきものであり、開示範囲を最小化する必要があることから、仮名化が広く用いられる。一方で、認可判断における信頼性を仮名性により低下させることなく実現するためには、仮名化された状態で、ユーザ同一性の判定を行うことが求められる。

プライバシーに配慮した必要最小限の情報開示のためのスキームとして代表的なものに準同型暗号[4]がある。その処理速度の観点からの実用性は常に問題になっており、一部肯定的な結果[5]もあるが,処理が負担になる現状は変わっていない。

本研究では、Shibboleth/SAML を通信プロトコルに採用し、ID 管理サーバ(IdP)と SP とが認証連携プロキシを介してフェデレーションを構築しているという条件のもとで、異なる認証連携プロキシを介して認証連携しているユーザの同一性を、その仮名性を担保しつつ、サービス提供サーバが確認できるようにするための仕組みを設計し、開発する。

Shibboleth による認証連携では、IdP は、ePTID (eduPerson Targeted ID)と呼ばれる SP 毎に異なる仮名 ID を提示し、SP が結託して名寄せを行う攻撃から利用者の仮名性を保護している。これは、図 1 のように、異なる認証連携プロキシを介して一つの SP と連携している場合においても同様で

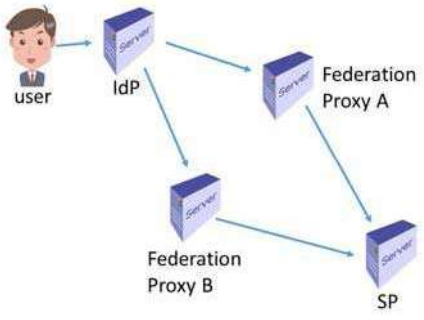


図 1 複数の認証連携プロキシを介した認証連携

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

ある。SPは、異なる認証連携プロキシを介して仮名でアクセスしてきている一人のユーザが同一であるかどうかを直接的には知ることができないし、知るべきではない。

しかしながら、このような仮名性の下では、SPが同一ユーザに対して提供するサービスの回数を限定したりすることが困難である。たとえば、1ユーザに対して1回だけ提供したいサービスでも、複数の認証連携プロキシを介することにより、SPの側ではそれらを同一のユーザであると識別できないため、連携するプロキシの数だけサービスを受けられるような抜け道ができてしまう。

そこで本研究では、サービス提供サーバに対して補助的に働く「カウンティングサーバ」(counting server)を導入してこの問題を解決する。IdP は、CID と呼ぶ補助的な仮名を、暗号化して、認証連携プロキシを介し SP に対して送出する。SP はこの CID をキーにしてカウンティングサーバへ問い合わせることで、ユーザが当該のサービスを何回受けているかを知ることができる。カウンティングサーバ自体はサービスを受けている回数だけを管理し、ユーザ名や受けているサービスの内容に関する情報は一切保持しない。これにより、ユーザの仮名性を担保しつつ、同一ユーザに対して提供するサービスの回数を限定することが可能となる。

以上の機能を持つ通信プロトコルと認証連携機構を設計し、Shibboleth ならびに SimpleSAML.php を用いて実装した。

以下、2 章では SAML と Shibboleth について述べる。3 章では、一人のユーザが複数の認証連携事業者を介してサービス提供者にアクセスする場合のユーザ同一性確認の問題について述べる。4 章で、仮名性を担保しつつユーザ同一性を確認する機構の提案とプロトコル設計、実装の説明を行い、5 章ではそれによって実現できるプライバシー保護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 [6]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 [7]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)の間で認証と属性情報交換、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など関係者の合意が得られる場合に限定して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。これは、Windows CardSpace や OpenID における PPID (Private Personal Identifiers)[8] [9]と同じ考え方に基づくものである。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[10]。佐藤らは、認証連携プロキシに匿名化の役割を担わせることで、認証連携において IdP と SP とを相互に秘匿することで高度な仮名性を実現しつつ、プロキシに対しては中継される属性情報を暗号化して秘匿する方式を提案している[11]。中村らは、汎用かつ共通的な ID を用いずに SAML における attribute aggregation を実現する方法を提案している[12]。

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状況を同時に実現できる。

実際には、認証基盤間の一般的な ID 連携を、このような IdP と認証連携プロキシとの関係とみなすことができる。たとえば以下のような認証連携を考える。大学が IdP を運用し学生に ID を配布しているとする。一方、学生は、認証連携を行っている民間の事業者において独自にアカウントを取得しているとする。大学と認証連携事業者との認証連携により学生の大学での ID と事業者でのアカウントを紐付けることで、仮名性を担保したまま大学は当該ユーザが学生であることの身分保証を行う。これにより、学生が民間の事業者のアカウントで学外の SP が提供するサービスを利用する際に、仮名性を保ったまま、学生の身分を属性として認証連携事業者に提供してもらうことが可能となる。

しかしながら、このような仮名性の下では、SP が同一ユーザに対して提供するサービスの回数を限定したりすることが困難である。たとえば、学割サービスを提供する際に 1 人のユーザに対する割引の適用機会にある上限を設けておきたいようなケースを考える。当該の大学とそのような認証連携を行っている認証連携事業者が一つに限られるのであれば、単に仮名 ID ごとに提供したサービスの回数をカウントしておけばよい。しかし、図 2 のように、複数の認証連携事業者を介して大学から当該 SP のサービスが利用できるようになっていたり、あるいは大学から認証連携事業者を介さずに直接 SP と認証連携したりしている場合には、異なる経路でアクセスしてきた学生を同一のユーザであると識別する手段がない。

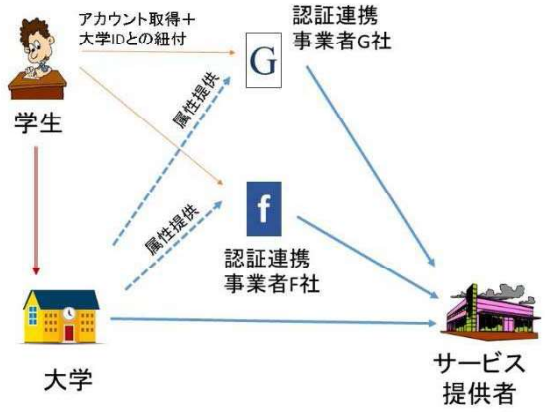
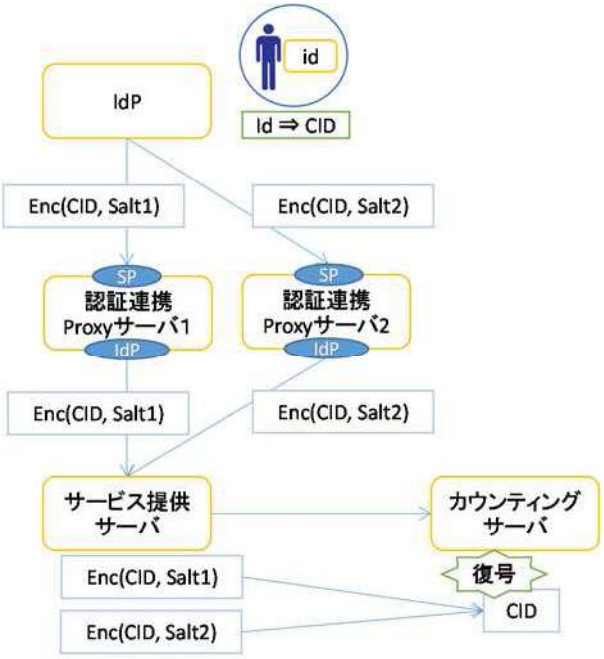


図 2. 認証連携事業者を介したサービス提供

この問題に対する単純な解は、IdP が SP に対して、当該サービスに対してユーザを識別する独自の仮名を属性として提供することである。この仮名は、認証連携プロキシとして働く認証連携事業者に対しては暗号化して秘匿することができる[8]。しかしながら、IdP が、ユーザがどの SP でどのサービスの提供を受けるかを知ってしまう点、また SP 側が、同一ユーザが異なる認証連携事業者からのアクセスしてきた場合に紐づけができてしまうという点で、Shibboleth におけるプライバシー保護の考え方からすると後退している。



4. 仮名性を担保したユーザ同一性の確認機構

4.1 ユーザ同一性の確認機構の設計

3 章で述べた課題に対する解として、SP に対して補助的に働く「カウンティングサーバ」(counting server)を導入してこの問題を解決する（図 3)。IdP は、対象となるサービスごとに、CID と呼ぶ補助的な仮名を、暗号化して、認証連携プロキシを介し SP に対して送出する。SP はこの CID をキーにしてカウンティングサーバへ問い合わせることで、ユーザが当該のサービスを何回受けているかを知ることができる。カウンティングサーバ自体はサービスを受けている回数だけを管理し、ユーザ名や受けているサービスの内容に関する情報は一切保持しない。これにより、ユーザの仮名性を担保しつつ、同一ユーザに対して提供するサービスの回数を限定することが可能となる。

以上のアイデアについて、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て、IdP から SP への認証を認証連携プロキシが中継するという条件のもとで、異なる認証連携プロキシを介して認証連携しているユーザの同一性に関し、その仮名性を担保しつつ、SP がサービスごとに当該ユーザのこれまでの全ての認証連携プロキシを介したサービスの総利用回数を確認できるようにする仕組みを実装した。

図 3 において、認証連携プロキシサーバは、IdP から SP を隠蔽する。SP に対しては ID 管理サーバとして認証要求を受け取り、IdP に対してはサービス提供サーバとして、同じ内容の認証要求を ID 管理サーバに送り直し、受け取

表 1. 各サーバで使用している主なソフトウェア

サーバ	説明
ID 管理サーバ(IdP)	shibboleth-identityprovider-2.3.8, tomcat-6.0.24, apache-2.2.15 暗号化 ID の生成のため、shibboleth-identityprovider-2.3.8 用のモジュール(DataConnector)を追加
認証連携プロキシサーバ	simplesamlphp-1.10.0, apahce-2.2.15 simplesamlphp-1.10.0 用のモジュールを追加
サービス提供サーバ(SP)	shibboleth-sp-2.5.1, apache-2.2.15, simplesamlphp-1.10.0 暗号化 ID を受け取る認証関係については、shibboleth-sp-2.5.1 を使用。カウンティングサービス回りについては、simplesamlphp-1.10.0 用のモジュールを追加して使用
カウンティングサーバ	simplesamlphp-1.10.0, apache-2.2.15, MySQL-5.1.66 カウンティングサービスのサーバとして、simplesamlphp-1.10.0 用のモジュールを追加して使用。カウンタのデータを保存するために、MySQL-5.1.66 を使用

った結果を元の SP に返す。このとき、元のサービス提供サーバに関する情報は ID 管理サーバには渡さないようにする。

IdP においては、認証連携プロキシサーバの要求により、ユーザの ID から暗号化した ID（以下、暗号化 ID）を生成する。暗号化 ID は、ユーザの ID から生成する匿名化された CID と、認証連携プロキシごとに異なるソルト文字列を、対象となるカウンティングサーバの公開鍵により暗号化したものを想定している。

カウンティングサーバは、IdP により生成された暗号化 ID とサービス名およびカウンタ要求から構成されるカウンティングサービス要求を受け取り、暗号化 ID より元の CID を取り出し、取り出した CID とサービス名をキーとするカウンタを、カウンタ要求に応じてサービスする。カウンタは不揮発性の記憶領域に保存される。カウンタ要求として、現在値の照会、インクリメント、デクリメント、リセット、並びに未使用のカウンタの取得に対応する。

SP においては、ユーザごとのカウントが必要なサービスを行う場合、カウンティングサービス機能を使ってサービスの回数をカウントし、カウンタの値に基づいてサービス提供の可否を判断する。

4.2 ユーザ同一性確認機構の実装

4.1 節で設計したユーザ同一性確認機構の動作は次のようになる。まずユーザが SP にアクセスすると、SP は認証を要求する。ユーザは DS (Discovery Service)を経て IdP を選択し、その認証を受ける。この際、ユーザは認証連携プロキシサーバ、IdP の二度、DS による選択を経由する必要がある。ユーザが IdP による認証に成功すると、IdP はユーザの ID からハッシュにより匿名化された CID を生成した後、認証連携プロキシサーバごとに異なるソルト文字列と CID とを、カウンティングサーバの公開鍵で暗号化して暗号化 ID を生成する。暗号化 ID は、IdP が発行するユーザの属性情報とともに、認証連携プロキシサーバを介して、SP に送られる。この結果、SP は、カウンティングサービスを受けるために必要なユーザごとの暗号化 ID を取得することになる。

SP は、取得した暗号化 ID とサービス名、カウンタ要求をカウンティングサーバに送り、カウンティングサーバは要求を処理して結果を返す。要求は SAML の AttributeQuery の形式で、カウンティングサーバが用意するカウンティングサービス用の URL に対して SOAP を用いて送られ、結果は AttributeQuery に対する AttributeStatement として返される。

また、実際のサービスで必要となる、「ID 管理サーバにおいてユーザ ID を暗号化 ID に変換するサービス」も実装している。これはたとえば、あるユーザが何らかの原因で受けるはずだったサービスを受けられなかったため、再度サービスを提供するためにカウンタの値をリセットしたいといった場合に必要なものである。本システムでは、ユーザが認証に成功した結果得られる属性情報として、ユーザ ID に対応する暗号化 ID を得ることができるが、カウンタの値をリセットするだけのためにサービスの管理者に特定ユーザの認証情報を渡すことはできないし、ユーザにカウンタをリセットする権限を与えることもできない。ユーザ ID と CID の対応をつけられるのは IdP のみであるため、サービスの管理者が「ユーザ ID を暗号化 ID に変換するサービス」を用いて暗号化 ID を取得し、カウンティングサービスを呼び出すことができるようにした。このとき、サービスの管理者は認証連携プロキシサーバを経由して ID 管理サーバのサービスを呼び出す。このサービスは、カウンティングサービスと同様、AttributeQuery とその結果の AttributeStatement により行われる。

表 2. カウンティングサービス要求一覧

cmd の値	説明
query	カウンタの現在値を取得する。
increment	カウンタの値を argValue で指定した値だけ増やす。増やした結果が cnsMaxValue を超える場合は、カウンタの値を変更せず、status として-2 を返す。
decrement	カウンタの値を argValue で指定した値だけ減らす。減らした結果、0 より小さくなる場合は、カウンタの値を変更せず、status として-2 を返す。
reset	カウンタの値を 0 に設定する。
new	未使用のカウンタ名を取得する。

以上を、CentOS 6.3 上で、Web サーバとして apache-2.2.15（mod_ssl 等を含む）、shibboleth-identityprovider が動作する java アプリケーションコンテナとして tomcat-6.0.24 を用いて実装した。表 1 に、各構成要素で使用されている主なソフトウェアを、表 2 に、カウンティングサービスの要求の一覧を、図 4 に、カウンティングサービス要求(increment)に対する応答の例を示す。

5. 考察

4 章の機構において、ユーザに対して SP で提供されるサービスは、3 章で述べたプライバシーに関する要件により、IdP ならびにカウンティングサーバに対して秘匿されるべきである。そのため図 4 に示すようにカウンタ名は対象 SP やサービスの内容に関する情報を含まない仮名的なものとしているが、それ以外にやりとりされる情報についても、対象 SP やサービス名が IdP やカウンティングサーバに対して隠蔽されるよう配慮する必要がある。

提案方式を用いていても、SP が、異なる認証連携プロキシを介してのサービス要求が同一ユーザからのものであることを知りうるケースは存在する。たとえば、サービス利用者がゼロの状態で、二つの異なる認証連携プロキシから続けてサービス要求が来た際に、カウンタ値が 2 になれば、二つの要求は同一ユーザからのものであることは明らかである。この例を含め、提案方式は、絶対的なプライバシー保護を目指しているわけではなく、3 章で考えたようなサービスモデルにおいて、個人を特定する機会はできるだけ減らそうという緩いポリシーに基づいて設計されている。

6. おわりに

本研究では、異なる認証連携プロキシを介して認証連携しているユーザの同一性を、その仮名性を担保しつつ、SP が確認できるようにするための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、大学のようなプライバシー保護に対して厳格である必要のある機関が、民間の事業者と認証連携を行う上でのハードルを下げることができたと考える。

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行うとともに、実利用の検討を進めていきたい。

謝辞

ソフトウェアの実装に協力いただいた（株）オクトパスの各位に感謝する。なお本研究は、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] REFEDS: Federations, https://refeds.org/federations, last visited May 15, 2015

[2] 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.

[3] 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.

[4] Gently, C.: Fully homomorphic encryption using ideal lattices, Proc. 41st ACM Symp. Theory of Computing, pp. 169--178, 2009.

[5] Naehig, M., Lauter, K., Vaikuntanathan, V.:Can homomorphic encryption be practical? Proc. 3rd ACM workshyop on Cloud Computing Security, pp. 113--124, 2011.

[6] OASIS: Advancing open standards for the information security, https://www.oasis-open.org/, last visited May 15, 2015.

[7] Bertocci, V., Serack, G., Baker, C.: Understanding Windows CardSpace: An In-troduction to the Concepts and Challenges of Digital Identities. Addison-Wesley, Reading, Massachusetts (2008)

[8] ICAM, OpenID 2.0 Profile, 2009.

[9] Shibboleth Consortium: http://shibboleth.net/, last visited Apr. 1, 2013.

[10] 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.

[11] Hiroyuki Sato, Yasuo Okabe, Takeshi Nishimura, Kazutsuna Yamaji, Motonori Nakamura: Privacy Enhancing Proxies in a Federation:

Agent Approach and Cascade Approach, Journal of Information Processing, Vol.22, No. 3, pp.464-471, July 2014.
[12] Motonori Nakamura, Takeshi Nishimura, Kazutsuna Yamaji, Hiroyuki Sato, Yasuo Okabe: Privacy Preserved Attribute Aggregation to Avoid Correlation of User Activities Across Shibboleth SPs, The 7th International Workshop on Middleware Architecture in the Internet (MirArch2013) (Proc. IEEE COMPSAC2013 Workshops), July 2013.

```
<samlp:Response
    xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    ID="_854c6140f6ee41bc804b2da90ca20d06675d2c0834"
    Version="2.0"
    IssueInstant="2013-02-15T07:33:23Z"
    Destination="https://kusp1.octopath.co.jp/shibboleth-sp"
    InResponseTo="_a8dbd0d807dcf097314fedb1d91d79344a5a7f8125">
  <saml:Issuer>https://kuap1.octopath.co.jp/simplesaml/saml2/idp/metadata.php</saml:Issuer>

      :

    <saml:Subject>
      <saml:NameID
Format="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent">cnt1</saml:NameID>
      <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
        <saml:SubjectConfirmationData                    NotOnOrAfter="2013-02-15T07:38:23Z"
Recipient="https://kusp1.octopath.co.jp/shibboleth-sp"
InResponseTo="_a8dbd0d807dcf097314fedb1d91d79344a5a7f8125"/>
      </saml:SubjectConfirmation>
    </saml:Subject>
    <saml:Conditions NotBefore="2013-02-15T07:33:23Z" NotOnOrAfter="2013-02-15T07:38:23Z">
      <saml:AudienceRestriction>
        <saml:Audience>https://kusp1.octopath.co.jp/shibboleth-sp</saml:Audience>
      </saml:AudienceRestriction>
    </saml:Conditions>
    <saml:AttributeStatement>
      <saml:Attribute Name="counterName">
        <saml:AttributeValue xsi:type="xs:string">cnt1</saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="cmd">
        <saml:AttributeValue xsi:type="xs:string">increment</saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="argval">
        <saml:AttributeValue xsi:type="xs:integer">1</saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="cnsMaxValue">
        <saml:AttributeValue xsi:type="xs:integer">10</saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="stValue">
        <saml:AttributeValue xsi:type="xs:string">6</saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="status">
        <saml:AttributeValue xsi:type="xs:integer">0</saml:AttributeValue>
      </saml:Attribute>
    </saml:AttributeStatement>
  </saml:Assertion>
</samlp:Response>
```

図 4. カウンティングサービス要求に対する SAML 応答の例