Vol.2012-DPS-150 No.42
Vol.2012-CSEC-56 No.42
2012/3/1

クラウドコンピューティングにおける認証連携と属性利用技術に関する考察

下道高志† 佐々木良一†

さまざまな分野でクラウドコンピューティングの利用が進むのに伴い，複数のネットワークドメイン上で複数のサイトを連携するサービスが今後増加すると予想される．そこで必要とされる技術は，単なるサイト間の認証連携だけでなく，分散された属性情報を利用するサービスのための技術であると考えられ，扱われる属性情報は，静的属性情報に加え動的属性情報も今後増加すると考えられる．本稿では，アイデンティティ管理／サービス技術である SAML / ID-WSF に注目し，クラウド上への適用の実際と問題点を考察し，ID-WSF を拡張した高速性/安全性を備えた属性利用技術を提案する．提案する技術の有効性を確認するために，クラウド上の SAML SSO とそこで利用される SOAP 通信の latency を実測し，その結果も合わせて考察を行う．

Consideration for Technology on Federated Authentication and Usage of Attributes in Cloud Computing

Takashi Shitamichi† Ryoichi Sasaki†

Federated services are expected to be widely deployed among multiple domain networks along with getting to use cloud computing in a lot of industry areas. Technology not only for federated authentication but also services using distributed attributes, which are not only static but also dynamic, are required. This paper focuses on SAML and ID-WSF, which are technology for identity management and services, discusses deployments and problem in the real world, then proposes fast and safe technology which extends ID-WSF. In order to verify the effectiveness of the proposed technology, latency of SAML SSO and exchanging SOAP messages are measured and considered in cloud computing environment.

1. はじめに

クラウドコンピューティングの利用がさまざまな分野で進むのに伴い，同一ネットワークドメイン上のサイトで完結するサービスだけでなく，ドメインをまたいで複数のサイトで連携するサービスが増えている． 連携されたサービスをユーザが利用する場合、複数サイト間での認証連携が必要となり，SAML (Security Assertion Markup Language) ， OpenID ，OAuth(Open Authorization)といった仕様が規定されている[1][2][3]．連携するサービスを提供する各サイトが保有する個人属性情報の集合をアイデンティティと呼び，認証連携を行うことによって連携される個人属性情報の集合を，連携アイデンティティ (Federated Identity)と呼ぶ．

連携アイデンティティにおける個人の属性情報利用のためには，個人情報保護やセキュリティを考慮したプロトコルが必要であり，SAML や SAML を拡張した Shibboleth を利用する属性転送方式や，SAML のアサーションを利用しつつ属性利用の本人確認の仕様を備える Liberty ID-WSF (Identity Web Service Framework)などが規定されている[4][5].

クラウドコンピューティング上でサービスを展開する企業には，ユーザ企業のシステムと認証連携を行うためのインターフェースを提供するところもある．企業用途のクラウドサービスとして，例えば Google の Google Apps では”SAML Single Sign-On (SSO) Service for Google Apps”として SAML API の提供を行っている[6]．また Salesforce.com では” Single Sign-On with SAML on Force.com”として SAML に対応している[7]. 企業用途のクラウドコンピューティングでは SAML を利用した認証連携が地位を確立している．

SAML 等の認証連携のための技術が利用される一方，利用者の属性情報を利用するための技術は，速度と安全性の面で十分に検討する必要がある．属性情報には，氏名，住所，生年月日，性別といった静的な属性情報だけでなく，GPS で観測される位置情報等の動的な属性がある[8]．静的属性を扱う認証連携の技術に関して，ユーザデバイス機能のローミングの研究が行われ，認証時間の短縮を行う技術は考案されている[9]. しかし，日々蓄積されていると推測される動的な属性情報を扱うために，速度と安全性を十分に配慮した上で，認証連携と併せた技術の考案は過去行われていない．属性情報を取り扱うサイトにおいては，ユーザのインタラクション（会話）時に属性転送を高速に行わなければ，ユーザエキスペリエンス（ユーザ体験）に影響を与える可能性がある．サイトがユーザと“ネットワークの遠さ”= latency (遅延時間)が小さい地点に存在する場合には，属性転送時間は問題にならないと考えられるが，サイトがクラウド上の何処，地球の反対側のような地点に存在している場合は latency が大きく

† 東京電機大学
Tokyo Denki University

Vol.2012-DPS-150 No.42
Vol.2012-CSEC-56 No.42
2012/3/1

なると予想され，その結果，サービスを利用するユーザにとっての RTT (Round Trip Time)は長くなり，ユーザエキスペリエンスに大きな影響を与えると想定される．

そこで筆者は,認証連携および属性管理利用技術である SAML / ID-WSF を利用しつつ，静的な属性情報と動的な属性情報をユーザが高速かつ安全に扱うために，SAML / ID-WSF を拡張したアーキテクチャを考案した．本稿では，アーキテクチャの有効性を確認するために，実際のクラウド上で，SAML 適用における現状の性能を実測し，技術検証した結果を踏まえた上で考察を行っている．

2. SAML による認証連携の実際

SAML はセキュリティ情報を，ネットワークを通して交換するためのフレームワークとして，2000 年代初頭に考案された．その後，Liberty Alliance Project で作成された ID-FF (Identity Federation Framework) の仕様を取り入れて SAML2.0 へと発展し，様々なサービスで適用されてきた．クラウドコンピューティングに関しては第 1 節で述べたように企業用途のクラウドサービスの認証連携として実績がある．本節では SAML の SSO モデルの説明を行い，クラウド上での適用についての考察を行う．

2.1 SAML による連携 SSO の特徴とプロファイル

SAML による連携 SSO の特徴の一つは，トラストサークル(CoT：Circle of Trust)を形成した上で，ユーザが Web Browser 等の UA (User Agent)によって SSO を可能とすることである．CoT の技術的な実現方法として．図 1 に示すように，各サイトは個別にユーザのアカウントを保持し，個別のアカウント同士を互いのランダムな値である“見えないハンドル”（opaque handle）で紐づける．この方法により各サイトは独自にユーザ・アイデンティティを保持し続ける一方，サイト間では共通する情報が存在しないため，ユーザのプライバシーを確保することが可能となる．各サイトのうち，IdP (Identity Provider)とは，トークンの要素である名前や年齢といったクレームを作成するオーソリティであり，STS(Security Token Service)を運用する．また SP(Service Provider)はクレームを利用することによってユーザを特定し，アプリケーション・サービスを提供する．

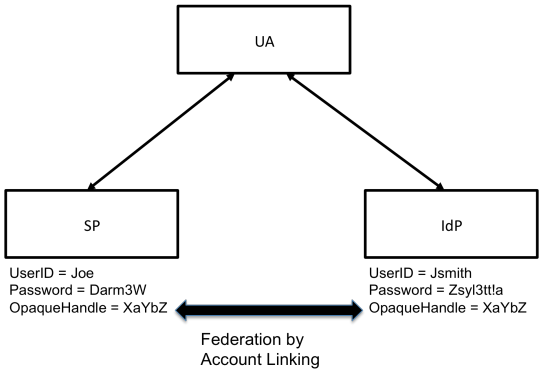


図 1 SAML の Federation

SAML 仕様書ではアサーション，プロトコル，バインディングに加え SSO 等のユースケースをプロファイルとして規定している. Web ブラウザによる SSO のプロファイルとしては次の 3 種類を定義している

(1) SP-initiated SSO: Redirect/POST Bindings
(2) SP-Initiated SSO: POST/Artifact Bindings
(3) IdP-Initiated SSO: POST Binding

この 3 種類のプロファイルの中で，Web browser に対する柔軟性と信頼されたサイト間の SOAP (Simple Object Access Protocol)通信により，セキュリティ面が考慮されている(2)の方式が注目を集めている．SAML 仕様書のシーケンスを図 2 に示す[10].

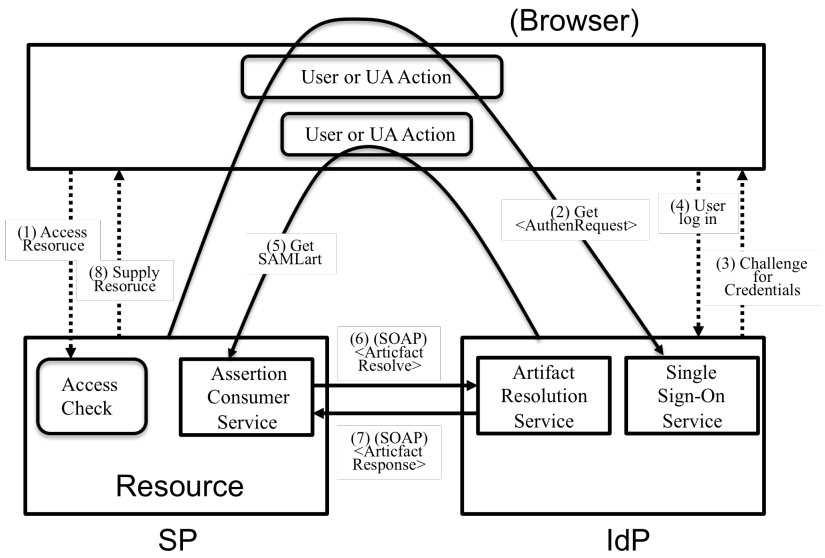


図 2 SAML SSO POST / Artifact Bindings

2.2 クラウド上での SAML SSO に関する考察

世界規模で展開されるクラウドコンピューティング上への情報システムの適用はいく

Vol.2012-DPS-150 No.42
Vol.2012-CSEC-56 No.42
2012/3/1

つかの問題を抱えている．企業が自社で管理する信用境界を越えて様々なプライバシーを含むデータが行き来する以上，個人情報保護に関する法律面での問題が存在することとなる[11]．一方，技術面では“ネットワークの遠さ”= latency (遅延時間) が問題となると考えられる. たとえば, 同じサービスを同じ仕様のサーバ上ではあるが，サーバの在処が違う 2 つのサービスがあったとする．東京在住者が東京地区にあるサーバでサービスを利用するのと，内容的に全く同じサービスをロンドンで利用するのとでは，ユーザエキスペリエンスに差が出てくる．これは距離と中継設備による latency が主な原因と考えられる. しかし, ユーザエキスペリエンスは重要である. “Web を利用するユーザは，読み込みに 3 秒以上かかると苛立ちを感じ，47%のユーザが 2 秒以内の Web ページ読み込みを期待し, Web ページの読み込み時間に 3 秒以上かかるとユーザの 40 %がそのサイトを去る”との調査結果がある[12].

2.1 節で説明したように，クラウド上での SAML SSO サービスは，3 つのプロファイルの中で Artifact Bindings の適用が，セキュリティの観点から望ましい．ユーザが利用する Web Browser とサービスを提供するサイトとの latency に起因する問題は多かれ少なかれ必ず発生するが， Artifact Bindings はさらに， IdP と SP 間の SOAP 通信による latency が加わり，ユーザエキスペリエンスに影響を与えると予想される． IdP / SP 間通信に限らないが，サービスを提供するサイト間通信の latency を小さくすることはユーザエキスペリエンス向上に必要と考える．

3. 属性利用技術

今日，属性を利用するさまざまなコンシューマーサービスが公開されている．Facebook や Twitter 等のように，属性を積極的に利用するための仕組みや API を独自に提供しているところもある．コンシューマーサービスにおいて属性を利用する技術は，認可の技術である OAuth を利用する場合が多い．最近では OAuth2.0 をベースとした OpenID Connect が提案され標準化の動きもあるが，セキュリティに関する考察や実証はこれからの課題となっている．コンシューマーサービスでは厳格なセキュリティに関する検証等を行うよりも，可能な限り早くサービスを立ち上げることが優先されることが多く，REST (Representational State Transfer) アーキテクチャスタイルが多用される[13][14].

一方，SOAP をベースとした属性利用技術としては，セキュリティコンテキストを表現する SAML アサーションを利用した ID-WSF が，Liberty Alliance Project によって規定されている．

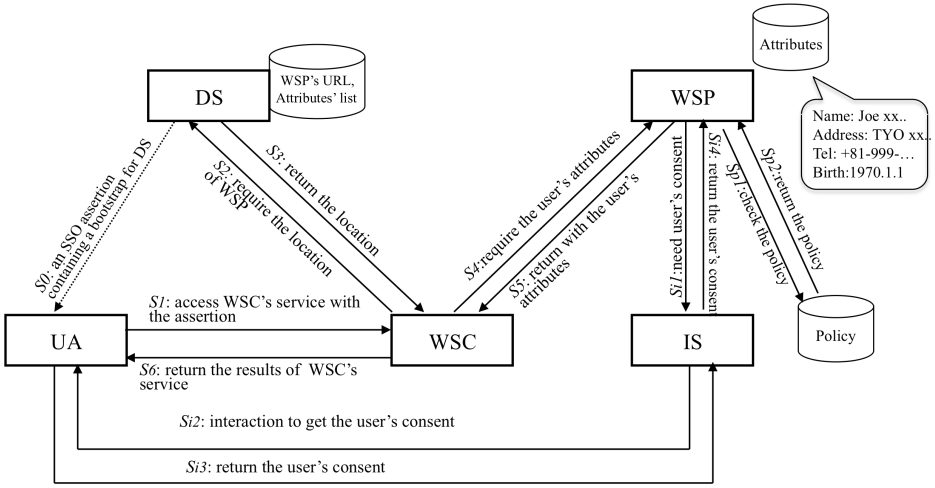


図 3 ID-WSF のシーケンス

3.1 ID-WSF

ID-WSF は異なるサイト間で，ユーザの意思に基づき属性情報を安全に流通させるための仕様である．安全性を保障するための仕組みとして通信プロトコルに SOAP を利用した上で，セキュリティメカニズム仕様を規定している．通信の秘匿性とメッセージの完全性を組み合わせて定義し，セキュリティの種別を“セキュリティメカニズム ID”と呼ばれる識別子で規定している．組み合わせには多くの方法があるが，たとえばメッセージの完全性を確保するために SAML トークンを使うことにより，以下を実現できる[15].

(1) アサーションによりユーザの特定
(2) 情報要求元である送信者の認証が XML 署名により検証
(3) 第三者機関(IDP)のアサーションに含まれる送信者の公開鍵により送信者の承認

ID-WSF ではユーザが事前に特定の属性プロバイダである WSP(Web Service Provider)に属性を登録し，WSP の URL を検索サイトである DS (Discovery Service) に登録する．ユーザがサービス提供サイトである WSC（Web Service Consumer）を利用する時に，DS のポンタが WSC に通知されることによって，WSC は WSP の URL に登録されている属性情報の項目リストを DS から入手する．WSC は項目リストにより

Vol.2012-DPS-150 No.42
Vol.2012-CSEC-56 No.42
2012/3/1

属性を WSP に要求し，ユーザの属性情報を入手する. WSP は WSC から属性要求があった場合，次の 2 通りの動作を行う．

(1) 事前に定めたポリシーに従い属性を返す
(2) 属性の持ち主に可否を逐次問い合わせる(IS : Interaction Service)

図 3 に ID-WSF のシーケンスを示す．属性の管理／利用技術として，ID-WSF はさまざまな分野で適用されている．コンテンツ視聴のための複数デバイス間の情報連携や, 機微情報を扱う医療分野等, 国内での多くの研究と実績がある[16][17][18][19][20].

3.2 クラウドコンピューティング上での ID-WSF の問題点

ID-WSF は機能を提供するサイトやレポジトリの数が多く，シーケンスも複雑なものとなっている．その一方で， 2003 年の ID-WSF1.0 発表後，People Service 等を加えた現在の ID-WSF2.0 まで，プライバシーとセキュリティを配慮した機能とアーキテクチャは，Web サービスの他の仕様では存在しない．しかしながら ID-WSF が設計された時代と，クラウドやさまざまなコンシューマーサービスを高速に利用できる現在の環境とを比較すると，いくつかの点で ID-WSF の仕様として不足していると考える．具体的には次の点である．

(1) ユーザの属性情報は，静的な属性として想定されており，恒常的に増加するユーザの動的属性情報を用いるようなアーキテクチャや技術となっていない
(2) 地球上の“どこか“に存在する各サイト間，特に WSC と WSP がネットワーク的に遠いところに存在する場合，SOAP 通信の latency の問題が生じる

4. クラウドのための属性利用技術の提案

ネット上の多くのビジネスでは，サービス事業者はあらかじめ利用者から個人情報の提供を受けた上でサービスを実施している[21]．しかし近年では，スマートフォン等の携帯型デバイスが急増し, GPS 等も含んだセンサーデータが大量に発生している．また, 人の行動をデータ化したライフログも, 広い意味での動的な属性情報といえる．第 1 節で述べたように，動的な属性情報が“ネット上のどこか”に保存されている．しかし今までの Web サービスの技術では，ネットを流れる情報は，主に静的な属性情報を対象としてきた．Facebook 等による動的な属性情報を API 経由でアクセスできようにする取り組みもあるが，連携認証の下での技術については，ID マッピング情報の登録方式やクラウド向け認証基盤等の研究があるが，動的な属性情報を，安全かつ高速に扱える包括的な属性利用技術はほとんど考案されていない[22][23][24]．そこで筆者は ID-WSF を拡張した新たな属性利用技術を提案する．

4.1 動的属性提供サービスの分離

ID-WSF では PIP に記述したクレームを，Data Service Template にもとづいたフォーマットに従って，WSP に保存してある属性を SOAP 通信で WSC に送る．動的属性も PIP のクレーム記述を追加するだけで，基本的には取り扱うことが可能である．しかし，動的属性は短い時間間隔で追加され，かつ，データ量が多い． 3.2 節の(1)で問題点としたように，頻繁には追加されない従来同様の属性情報として扱うことには無理がある．そこで筆者は，動的属性を別のリポジトリとサービスに分離した．静的属性を取り扱う WSP を WSPs，動的属性を取り扱う WSP を WSPd とした．そして WSPs は静的属性情報のレポジトリを，WSPd は動的属性情報のレポジトリを持つこととした．技術的には次の 3 点の拡張である．

(1) PIP の拡張を行い，クレームとして動的属性サービスへのポインタを埋め込む
(2) WSC が動的属性サービスへとポイントする機能を追加
(3) 動的属性サービス用の Data Service Template を追加

図 4 に全体のアーキテクチャを示す.

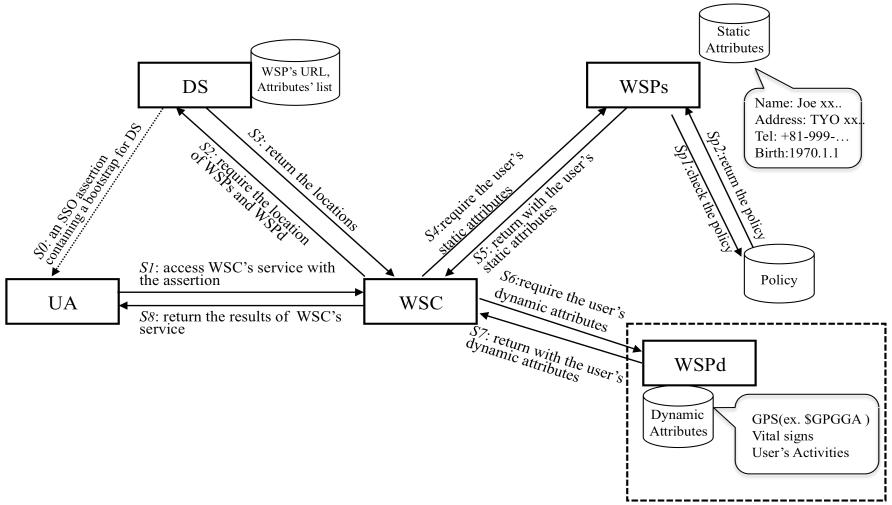


図 4 静的属性と動的属性を分離したアーキテクチャ

Vol.2012-DPS-150 No.42
Vol.2012-CSEC-56 No.42
2012/3/1

4.2 動的属性提供機能のローミング

前 4.1 節で動的属性を静的属性と分離したが，WSC は WSPd に保存される非常に多量になると想定される動的属性データに，ネット越しにアクセスしなければならない．この方式において，WSC と WSPd との間での通信が高速に行われることが必要である．特に WSC によってユーザに提供されるサービスが，ユーザとのインタラクション（会話）によって WSC と WSPd で都度属性情報がやり取りされる場合，WSC と WSPd との間の latency が，ユーザエキスペリエンスに影響を与える可能性が高い． WSC と WSPd がネットワーク的に近いところ，つまり latency が低いところであれば，ネットワークを起因とするユーザエキスペリエンスの問題は起らないと思われる．この問題を解決すべく筆者は，WSC と近い拠点に WSPd と同様の処理を行うサイトである WSPr を設置し，WSPd の動的属性リポジトリをローミングさせることにより WSPd の機能を WSPr に代替させる方式を考案した．図 5 がローミング前と後の機能イメージを示している．

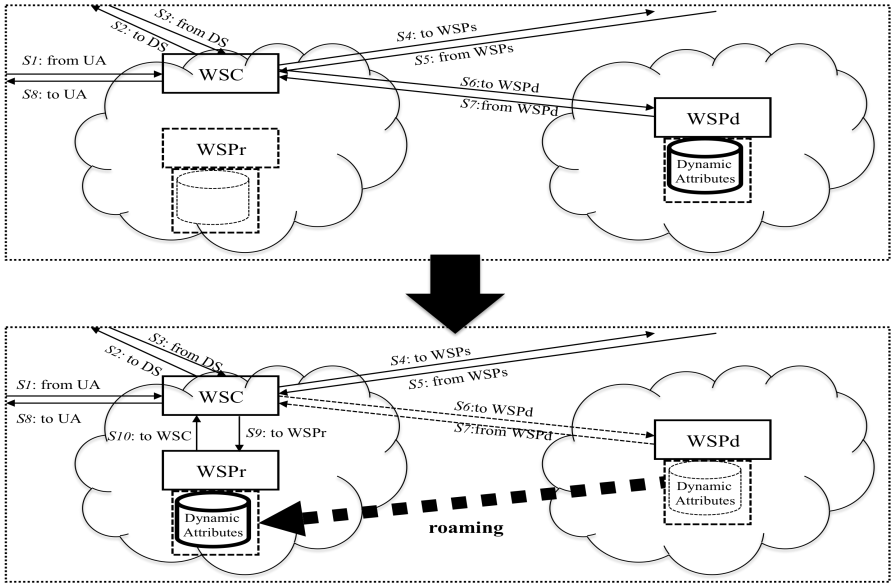


図 5 動的属性情報サービスをローミングしたアーキテクチャ

表 1 使用した Amazon EC2 インスタンスの仕様

Region	us-east-1(Virginia), us-west-1(Oregon), us-west-2 (California), eu-west-1(Ireland), sa-east-1(Sao Paulo), ap-northeast-1(Japan(Tokyo)), ap-southeast-1(Singapore)
OS / SW	Fedora 15 community Edition 32bit / JDK1.6, Tomcat6, OpenAM9.5.3
InstanceType	Small (1.7 GB of memory, 1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit), 160 GB of local instance storage)
MISC	Elastic IP for all (14 instances) instances

表 2 クライアント PC 環境の仕様

PC	Apple MacBook Air 11inch (Core2 DUO 1.6GHz, 4GB memory, 128GB SSD)
OS	Mac OS X 10.6.8
Web Browser	Firefox 9.0.1
LAN	Wired, up to 100Mbps
Internet	NTT FLET'S Hikari NEXT (100Mbps)

5. クラウド環境における latency の実際

前節で ID-WSF を拡張し，静的属性，動的属性を分離した上で，動的属性をローミングするアーキテクチャを考案した．2.2 節および 3.2 節において問題点として指摘したように，SOAP 通信の latency の問題があると仮定した．最適実装を行うための基礎データとして，SAML SSO POST / Artifact Bindings を実装したサイトをクラウド上に導入し，サイト間およびシーケンス毎の経過時間を実測し考察を行った．

5.1 クラウド上での SAML SSO の実測

全世界に広がるクラウドコンピューティングのインフラを利用した連携認証が，十分な速度を確保できているかを測定した． AWS (Amazon Web Service)が世界で 7 つのリージョンで提供する IaaS サービスである Amazon EC2 (Elastic Computing Cloud) でそれぞれ IdP と SP を立ち上げ，計 14 個の仮想ノードを使って SAML SSO サイトを構築し測定した。使用した AWS EC2 の仕様を表 1 に示す．測定に用いたクライアント PC 環境の仕様を表 2 に示す．

5.2 Web Browser, IdP, SP の RTT

クライアント PC と 7 リージョン，および，7 リージョン同士の計 56 通りの RTT の計測を行った結果を表 3 に示す．実測により以下の結果が導かれた．

Vol.2012-DPS-150 No.42
Vol.2012-CSEC-56 No.42
2012/3/1

表 3 クライアント，AWS EC2 リージョン間の RTT(msec，( )内は hop 数)

From \ To	Ireland	Sao Paulo	Virginia	Tokyo	Oregon	California	Singapore
Client PC	262.257 (25)	315.358 (22)	176.226 (19)	11.294 (11)	124.768 (21)	116.191 (19)	80.260 (14)
Ireland	0.595 (6)	209.370 (17)	94.720 (15)	284.699 (17)	174.144 (24)	157.955 (22)	458.114 (19)
Sao Paulo	208.338 (17)	0.582 (10)	140.327 (18)	314.552 (21)	219.416 (21)	183.778 (20)	351.245 (21)
Virginia	95.516 (18)	150.102 (19)	0.963 (8)	189.889 (18)	98.864 (17)	83.758 (15)	260.546 (17)
Tokyo	267.631 (21)	294.169 (18)	194.126 (19)	0.542 (6)	137.046 (20)	126.557 (18)	82.490 (14)
Oregon	170.743 (22)	224.769 (19)	98.878 (19)	123.098 (21)	0.718 (10)	20.546 (14)	317.168 (23)
California	160.125 (19)	204.288 (17)	83.377 (14)	1174.463 (18)	20.568 (13)	0.494 (6)	288.349 (18)
Singapore	442.614 (20)	376.727 (20)	248.398 (18)	86.477 (16)	305.802 (19)	300.779 (16)	0.619 (6)

- クライアント PC から最も大きい RTT は Sao Paulo であり，Tokyo の 30 倍近い値を示している．
- リージョン間で最も大きい RTT は Ireland/Singapore であり，リージョン間で最も小さい RTT を示している Oregon/California の 20 倍以上の値を示している．
- 同一リージョン内における RTT は 1msec 未満である

5.3 AWS EC2 上での SAML SSO における経過時間計測

SAML SP-Initiated SSO: POST/Artifact Bindings に基づき，AWS EC2 7 リージョンに各々IdP, SP を立ち上げた．シーケンスを図 6 に示す．計測点は *t1* ~ *t24* の 24 カ所である．ここで，

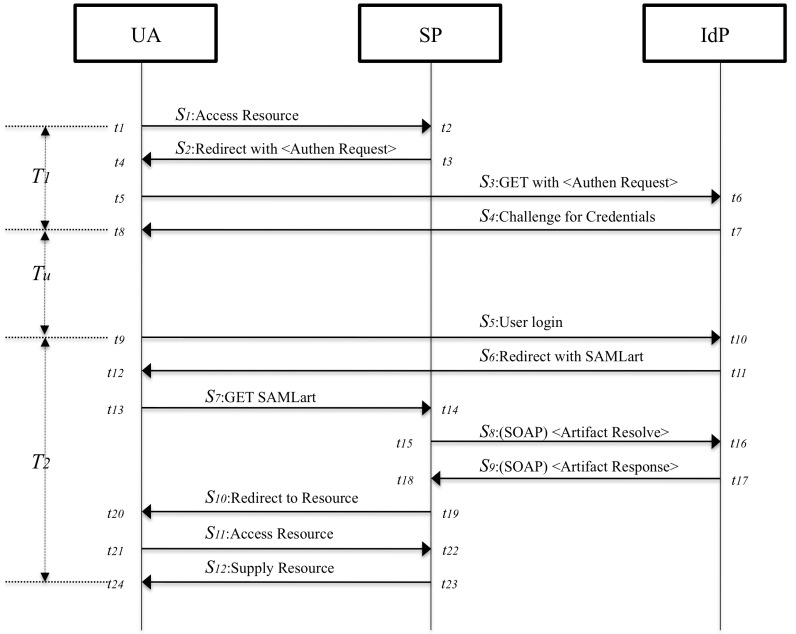
- T_1 = SP のサービスリソースにアクセスし IdP からのログイン画面が出力されるまでの時間
- T_2 = ログイン後 SP からサービスリソースが表示されるまでの時間
- T_u = ユーザがログイン入力する時間

とすると，最初に SP のサービスリソースにアクセスしてからリソース画面の表示までの時間 T は

- $T = T_1 + T_u + T_2$

である．T_u は属人的な値であるため，当該シーケンスにおける処理時間 T_s は，$T_s = T_1 + T_2$ であり，T_s が実際のサービスのレスポンス時間となる．故に T_s の大きさがユーザエキスペリエンスに影響を与えると考えられる．
IdP / SP のセットの実測は表 4 に示す通り，以下の結果となった．

- T_1 が最大となるのは，Ireland / Sao Paulo の 989msec である
- T_2 が最大となるのは，Singapore / Ireland の 1,929msec である．
- T_s が最大となるのは，Ireland / Sao Paulo の 2,540msec である．
- T_1，T_2，T_s とも Tokyo / Tokyo が最小である．



T_1:サービスにアクセスし，ログイン画面が表示されるまでの時間
T_u:ユーザがログイン入力を終了するまでの時間
T_2:ユーザがログイン終了後，サービスの画面をブラウザに提供するまで時間

図 6 SAML SSO POST/Artifact Bindings のシーケンス

T_s の最大値である Ireland / Sao Paulo は Tokyo / Tokyo の 10 倍近い値を示している．T_1 は最初にユーザがアクションを起こし，次の入力，すなわちログインまで「待たさ

Vol.2012-DPS-150 No.42
Vol.2012-CSEC-56 No.42
2012/3/1

表 1　IdP / SP 間の経過時間

IdP \ SP		Ireland	Sao Paulo	Virginia	Tokyo	Oregon	California	Singapore
Ireland	T_1	874	989	838	674	715	426	448
	T_2	955	1,551	1,039	1,042	965	933	1,426
	T_3	1,829	2,540	1,877	1,716	1,680	1,359	1,874
Sao Paulo	T_1	950	702	645	403	902	492	459
	T_2	1,459	1,114	1,267	1,190	1,096	1,121	1,353
	T_3	2,409	1,816	1,912	1,593	1,998	1,613	1,812
Virginia	T_1	655	569	407	229	447	548	539
	T_2	1,003	1,272	640	879	795	702	1,052
	T_3	1,658	1,841	1,047	1,108	1,242	1,250	1,591
Tokyo	T_1	448	395	385	63	173	182	121
	T_2	1,466	1,498	869	199	1,763	683	412
	T_3	1,914	1,893	1,254	262	1,936	865	533
Oregon	T_1	471	509	496	325	468	377	365
	T_2	1,229	1,291	826	560	600	534	1,072
	T_3	1,700	1,800	1,322	885	1,068	911	1,437
California	T_1	523	517	503	190	346	486	237
	T_2	1,066	1,325	799	440	674	660	995
	T_3	1,589	1,842	1,302	630	1,020	1,146	1,232
Singapore	T_1	579	522	318	206	234	267	218
	T_2	1,929	1,595	1,057	442	1,009	1,035	405
	T_3	2,508	2,117	1,375	648	1,243	1,302	623

れる」時間である．しかしながら最大値でも 1 秒程度である．T_2 はログイン後，サービス画面をブラウザに提供するまでの時間，つまり画面表示が始まる時間であり最大値の場合，ユーザは約 2 秒「待たされる」ことになる．そこで T_2 の最大値に注目して，

表 5　T_1 経過時間

		Singapore / Ireland		Tokyo / Tokyo	
		elapse	Δ	elapse	Δ
T_1	t_1	0	0	0	0
	t_2	150	150	7	7
	t_3	152	2	13	6
	t_4	301	149	20	7
	t_5	388	87	30	10
	t_6	430	42	38	8
	t_7	537	107	56	18
	t_8	579	42	63	7

表 6　T_2 経過時間

		Singapore / Ireland		Tokyo / Tokyo	
		elapse	Δ	elapse	Δ
T_2	t_9	0	0	0	0
	t_{10}	41	41	8	8
	t_{11}	58	17	23	15
	t_{12}	131	73	30	7
	t_{13}	136	5	34	4
	t_{14}	286	150	42	8
	t_{15}	754	468	89	47
	t_{16}	1,048	294	89	0
	t_{17}	1,051	3	91	2
	t_{18}	1,230	179	92	1
	t_{19}	1,467	237	121	29
	t_{20}	1,615	148	127	6
	t_{21}	1,629	14	140	13
	t_{22}	1,779	150	188	48
	t_{23}	1,780	1	191	3
	t_{24}	1,929	149	199	8

Ireland / Singapore のセットについて Tokyo / Tokyo と比較し詳細に経過時間を示したのが，表 5，表 6 である．

ここで，図 6 で示される S_8 および S_9，すなわち IdP / SP 間の SOAP 通信に注目する．表 6 では，$t_{14} \sim t_{19}$ が該当する．さらにΔに注目すると，Δt_{15} = 468 msec，Δt_{16} = 294 msec，Δt17 = 3 msec　となっている．TCP/IP 通信の詳細を調べると，Δt_{15} は [SYN] ⇒ [SYN, ACK] ⇒ [SYN] であり，TCP/IP 接続確立の時間である．また Δt_{16} は POST のあと[ACK]が返ってくる時間であり，両方とも大きな値を示している．それに対し，IdP が S8 のメッセージを受信し，Artifact Response を組み立て，S8 メッセージを送信するまでの Idp 滞留時間である Δt_{17} は 3 msec と非常に小さい．処理が重く時間がかかると言われている SOAP メッセージの取扱いも，latency の問題に比べれば問題が小さいということが本節の実証で明らかになったといえる．

5.4 クラウド上での属性サービスに影響を与える latency に関する考察

インターネット上のサービスは，Web Browser と単一サービスプロバイダによる一対一の関係で成り立つものが多かった．Web2.0 と言われた REST 技術による複数サービスのマッシュアップにおいては，API の提供によりサイト間でサービスを連携する

ことを可能とした．また，SSO 技術の発展と適用の広がりにより，複数のサイト間でユーザの属性情報を安全に共有する技術的ベースができた．

一方，属性サービスは本稿で取り上げたように，静的属性だけでなく動的属性を扱うサービスが今後増えると予想される．今まではサービス提供サイトが独自に属性情報を保持し，サービスの提供を行っていたが，最近では Facebook や Twitter のように API によって属性を提供するサイトも出現してきている．今後サービスの高度化に伴い，属性の扱いはますます増えると予想され，独自に動的属性を提供するプロバイダが出現してくる可能性は高いと思われる．そしてサービス提供サイトは，動的属性提供プロバイダから大量の情報を読み込んで処理を行うことになる．

そのような処理スキームでは，サービス提供サイトと動的属性提供サイトの間の RTT が非常に重要となる．一方でクラウドを使ったサービスが浸透することにより，サービス提供サイトや動的属性提供サイトが，ネットワーク的に遠いところに存在する可能性もある．そのような中でも 2.2 節で説明したように，レスポンスは 2 秒以内を実現することがユーザエキスペリエンス上で望ましい．

本稿で紹介したローミングによる動的属性サービスは，RTT を短くすることが可能であり，ユーザエキスペリエンスの向上に有効であると考えられる．

6. おわりに

本稿では，クラウドコンピューティングの環境を前提に，静的属性と動的属性を分離し，さらに SAML / ID-WSF を拡張し，サービス提供サイトとの間の latency の低いサイトに動的属性情報をローミングし，RTT を短くすることによって，ユーザエキスペリエンスを向上させる方式を提案した．また，ネットワーク的に遠いサイト間での通信を，実際のクラウド環境のサイト上で実験することにより，ローミング方式がユーザエキスペリエンス向上に対し有効となる可能性が高いことを確認できた．これらの実験の結果から得た定量的データのさらなる分析を進め，ローミング方式を改良しながら実装に反映していく必要がある．

参考文献

1) OASIS Security Services (SAML) TC,
http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security
2) OpenID Authentication 2.0 – Final, http://openid.net/specs/openid-authentication-2_0.html
3) The OAuth 1.0 Protocol, http://tools.ietf.org/html/rfc5849
4) http://shibboleth.internet2.edu/
5) Liberty Alliance ID-WSF 2.0 Specifications including Errata v1.0 Updates,
http://projectliberty.org/resource_center/specifications/liberty_alliance_id_wsf_2_0_specifications_including_errata_v1_0_updates/?f=resource_center/specifications/liberty_alliance_id_wsf_2_0_specifications_including_errata_v1_0_updates
6) SAML Single Sign-On (SSO) Service for Google Apps
http://code.google.com/intl/en/googleapps/domain/sso/saml_reference_implementation.html
7) Single Sign-On with SAML on Force.com
http://wiki.developerforce.com/page/Single_Sign-On_with_SAML_on_Force.com
8) 下江達二：アイデンティティ管理技術の進展と変遷(<特集>Web アイデンティティと AI)，人工知能学会誌 24(4)，社団法人人工知能学会
9) Y.Takeda, S.Kondo, Y.Kitayama, M.Torato, T.Motegi : Avoidance of Performance Bottlenecks Caused by HTTP Redirect in Identity Management Protocols, DIM '06 Proceedings of the second ACM workshop on Digital identity management, ACM
10) Security Assertion Markup Language (SAML) V2.0 Technical Overview, 25 March 2008, OASIS
11) 下道高志：クラウドコンピューティングの現状と欧米におけるプライバシーへの取組み（《特集　ネット検索サービス事業の諸問題》），法とコンピュータ学会誌 No.28 July 2010，法とコンピュータ学会
12) Forrester Research : eCommerce Web Site Performance Today - An Updated Look At Consumer Reaction To A Poor Online Shopping Experience - , August 17, 2009
13) 井上，他：REST アーキテクチャスタイルにおけるセッションに関する一検討，通信講演論文集 2，2009 年電子情報通信学会総合大会
14) 江波戸，他：OAuth へのコンシューマ認可の組み込みに関する研究，情報処理学会創立 50 周年記念（第 72 回）全国大会
15) 菅野：ID-WSF2.0 を利用したセキュアな情報流通，カンターラ・イニシアチブ・セミナー2011，平成 23 年 06 月 03 日
16) 藤井，石川，他：複数デバイス間での認証情報連携によるシームレスなコンテンツ視聴サービス，社団法人映像情報メディア学会技術報告，2008 年 9 月 25 日
17) 爰川，他：医療・健康情報の流通・活用に向けた情報連携基盤の提案，情報処理学会研究報告，Vol.2009-DPS-141 No.14
18) 堀川：コンシューマ向け ID 連携サービスの構築・運用の実際とその戦略性，信学技法，IN2009-117(2010-1)，電子情報通信学会
19) 山村，他：放送を起点とした個人向け通信サービス利用におけるユーザー機器認証フレームワーク，FIT2010（第 9 回情報科学技術フォーラム），L-035
20) M.Hatakeyama, S.Shima : Privilege Federation between Different User Profiles for Service Federation, DIM '08 Proceedings of the 4th ACM workshop on Digital identity management, ACM
21) 千葉，他：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，Vol 47 No.3，Mar. 2006
22) 畠山：異なる連携プロトコルを仲介するプロキシ型属性情報管理システム，情報処理学会創立 50 周年記念（第 72 回）全国大会，5F-1
23) 鷲尾，他：クラウド向け認証基盤プラットフォームの実装と検証，情報処理学会第 73 回全国大会，4E-1
24) 牧，他：ID マッピング情報の登録方式に関する一考察，情報処理学会第 73 回全国大会，4E-2