Webサービスセキュリティの最新動向

― WS-SecurityとWS-SX（WS-SecureExchange）関連仕様（1）

佐藤史子 （日本 IBM　東京基礎研究所）

Web サービスは，ネットワークを介してメッセージ交換を行うため，なりすまし，改ざん，情報漏えいなどのセキュリティ上のリスクを負います．これらのリスクを回避するために，SOAP メッセージそのものを署名・暗号化することにより安全なメッセージ交換を実現するものが Web サービスセキュリティです．

本稿では，Web サービスセキュリティの基本仕様である WS-Security（OASIS Web Services Security Specification）[1] とその関連仕様（WS-SecureExchange）を解説します．これらの仕様の位置づけを**図 -1** に示します．まず最初に SSL との比較により，Web サービスセキュリティの特徴を説明します．

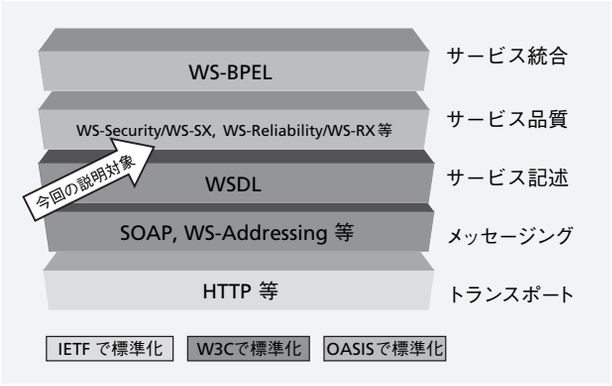

図 -1　WS-Security/WS-SX の位置付け

SSLとWebサービスセキュリティ

最初に述べた Web サービスのセキュリティ上のリスクを回避するには，認証（Authentication），データ完全性（Integrity），データ秘匿性（Confidentiality），否認不能性（Non-repudiation）を保証する仕組みが必要です．そのための既存の仕組みとしては，SSL が挙げられます．**図 -2** を見てください．Web サービスに SSL を適用すると，Point-to-Point で保護されたトランスポート上でメッセージ交換が行われるので，隣接する 2 つの Web サービス間での秘密情報（セキュリティコンテキスト）を安全に共有できます．

しかし，プロキシサービスのような中間ノードを介したメッセージ交換の場合，SSL では不都合が生じます．リクエスタからの情報は一度中間ノードで復号され，再度暗号化されて Web サービスへ送られます．すなわち，中間ノードにもセキュリティコンテキストが開示されてしまいます．

そこで，中間ノードを介したメッセージ交換においても，End-to-End でセキュリティコンテキストを共有するための仕組みとして，Web サービスセキュリティが提案されました．Web サービスセキュリティは，SOAP メッセージそのものに XML 署名・XML 暗号化を行うことで，メッセージを認証可能にし，完全性・秘匿性・否認不能性を保証します．リクエスタはメッセージの受信者である Web サービスとの間で共有している鍵を用いてメッセージを署名・暗号化すれば，署名検証・復号が行えるのは Web サービスだけになります．したがって，Web サービスセキュリティにより保護されたメッセージが中間ノードを経由しても，中間ノードは鍵を共有していないため，メッセージを読むことはできません．このように，End-to-End でセキュリティコンテキストを共有できることが，Web サービスセキュリティの特徴です．

Webサービスセキュリティ関連仕様

ここまで，Web サービスセキュリティがメッセージを直接署名・暗号化することで保護する仕組みであることを述べました．この仕組みは，複数の仕様で決められており，「Web サービスのセキュリティ：アーキテクチャとロードマップの提案」というホワイトペーパー [2] に**図 -3** のように表されています．

Web サービスセキュリティの基本となる仕様が，WS-Security（Web Services Security：SOAP Message Security）[1] で，署名・暗号化されたメッセージ構造を含む Security ヘッダの基本構造を規定しています．WS-Security は 2002 年から OASIS で標準化活動が行われ，

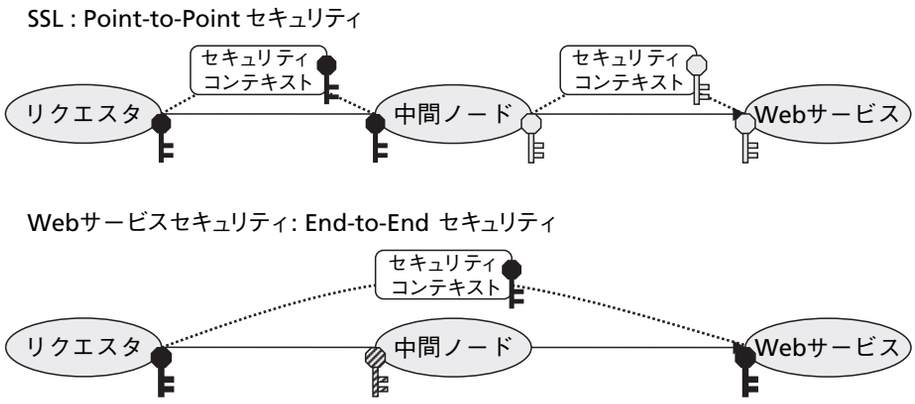

図 -2
SSL と Web サービスセキュリティ

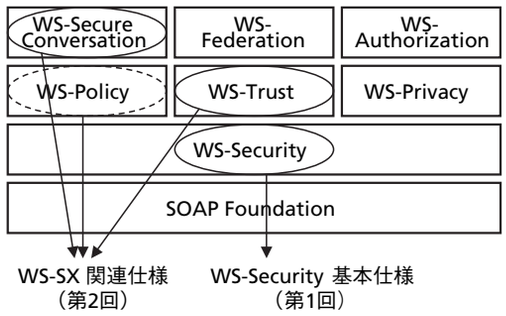

図 -3　WS-Security roadmap

現在は Version1.1 が公開されています.

WS-Security が Security ヘッダの構造のみを規定しているものであるのに対し，WS-Security で保護されたメッセージをどのような順序で交換するかということや，署名・暗号化のための鍵をどのように表すかなどを規定したものが，図 -3 で WS-Security の上に積まれている応用仕様です．これらのうち，OASIS の WS-SX TC（Web Services Secure Exchange Technical Committee）[3] で標準化活動が行われている仕様が WS-Trust（Web Services Trust）[4]，WS-SecureConverstaion（Web Services Secure Conversation）[5]，WS-SecurityPolicy（Web Services Security Policy）[6] です．現在 WS-Trust 1.3 と WS-SecureConvresation 1.3，WS-SecurityPolicy 1.2 の Public Review が行われています．本稿ではこれらの 3 つの応用仕様を WS-SX 関連仕様と呼ぶことにします.

図 -3 の WS-Policy には，現在 W3C で標準化が進められている WS-PolicyFramework と，この仕様に基づき具体的なポリシーの記述方法を規定した仕様が含まれます．このうち，セキュリティポリシーの記述方法を規定した仕様が OASIS で議論されている WS-SecurityPolicy です．図 -3 で WS-Policy が点線で囲まれているのは，WS-SX 関連仕様に含まれるのは WS-SecurityPolicy のみであることを表しています.

他の応用仕様のうち，WS-Federation は OASIS での標準化活動が始まろうとしています．WS-Authorization と WS-Privacy は，現状では標準化活動のための議論には至っていません.

本稿の構成

この連載では，図 -3 に示したように 2 回に分けて Web サービスセキュリティ関連仕様の解説をしていきたいと思います.

第 1 回目の今回は，基本仕様である WS-Security について説明します．まず，WS-Security を適用すると 1 つのメッセージがどのように保護されるのかを解説します．次に，WS-Security を使った安全なメッセージ交換の仕組みについて見ていきます.

第 2 回目は，第 1 回目の最後に説明する安全なメッセージ交換の方法を規定している 3 つの WS-SX 関連仕様（WS-Trust, WS-SecureConversation, WS-SecurityPolicy）について解説します.

WS-Securityによるメッセージ交換

まず最初に，WS-Security を使ったメッセージ交換の流れを見てみましょう．**図 -4** を見てください．これは RSA 暗号方式による WS-Security の適用例です．リクエスタは，自分の秘密鍵で SOAP メッセージを署名し，一時的に生成したセッション鍵を用いて暗号化します．セッション鍵は Web サービスの公開鍵で暗号化され，SOAP メッセージに追加されます.

リクエスタ側で WS-Security が適用されると，SOAP

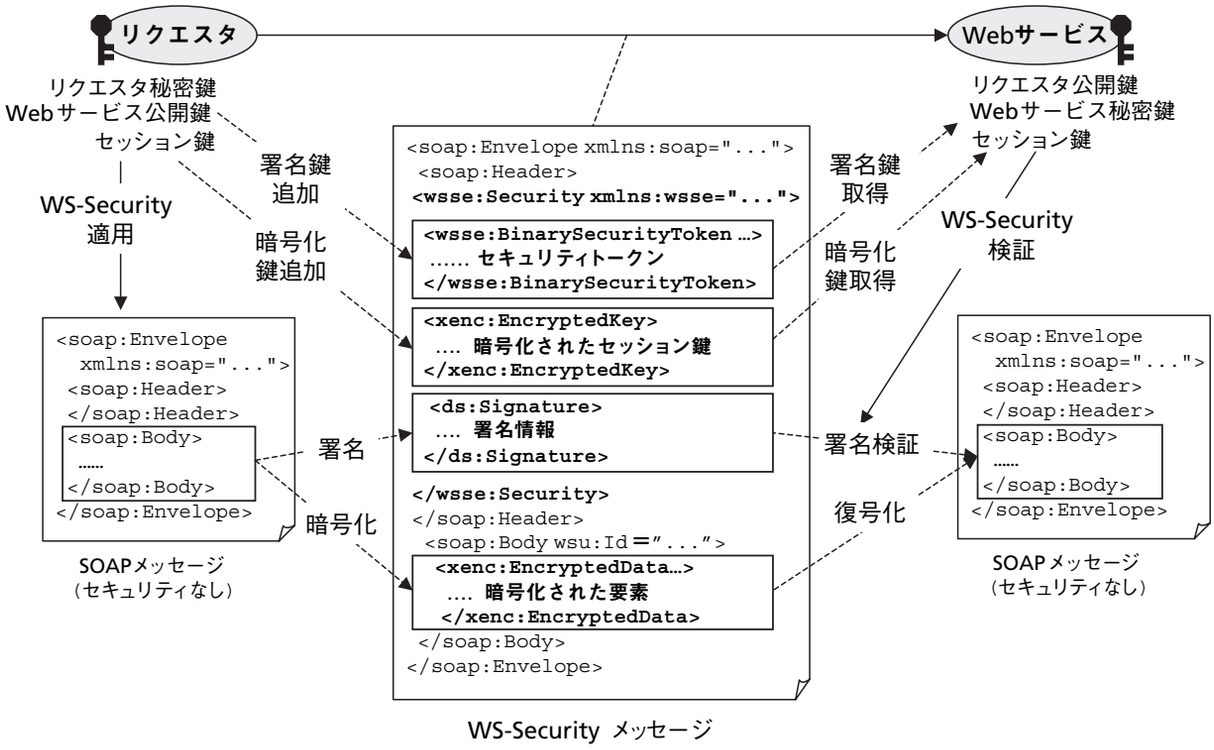

図 -4　WS-Security メッセージの処理

ヘッダの下に Security ヘッダが挿入されます．ここに，Body 要素への署名（Signature 要素），暗号化に使われた鍵を表す情報（EncryptedKey 要素），署名に使われた鍵を表す情報（BinarySecurityToken 要素）が挿入されます．また，暗号化された Body 要素の子要素は EncryptedData 要素に置き換えられます．このように署名・暗号化されたメッセージが Web サービスへ送信されます．

WS-Security に関する情報は，Security ヘッダの中に記述されますが，中でも重要なのはセキュリティトークンです．セキュリティトークンは，リクエスタと Web サービス間で共有しているセキュリティコンテキストを指すための抽象的な表現です．ここでのセキュリティトークンは，署名に使った鍵を指しています．

Web サービス側で WS-Security が適用されたメッセージを受け取ると，署名検証と暗号化された要素の復号化が行われます．署名検証のためには，最初にセキュリティトークンを参照し，署名検証のためのリクエスタ公開鍵を特定します．この鍵を使い，署名検証を行います．一方，Web サービスは自分の秘密鍵でセッション鍵を復号化し，このセッション鍵を使って暗号化された要素の復号化を行います．検証に成功すれば署名情報は不要になりますので SOAP ヘッダから削除し，暗号化されていた要素を復号化された要素に置き換えれば，元の SOAP メッセージを取得することができます．

セッション鍵を復号できる Web サービスでなければ，セキュリティトークンが指している鍵を特定できないので，暗号化されたメッセージを復号することができないということになります．そのため，リクエスタと Web サービス間に中間ノードが存在しても，元の SOAP メッセージが読まれることはありません．

では次に，WS-Security メッセージの詳細を見ていきましょう．

WS-Securityメッセージ構造

図 -5 は RSA 暗号方式を利用した場合の Security ヘッダの構造を示しています．

WS-Security の署名・暗号化では，XML 署名，XML 暗号化の仕様で決められている方法をそのまま利用しています．WS-Security の仕様で規定されていることは，署名や暗号化された要素をどのように Security ヘッダに入れるかというメッセージ構造です．したがって，本稿では署名・暗号化に関する要素の詳細な解説は XML 署名・XML 暗号化の解説を参照していただくこととし，メッセージ構造を中心に解説いたします．

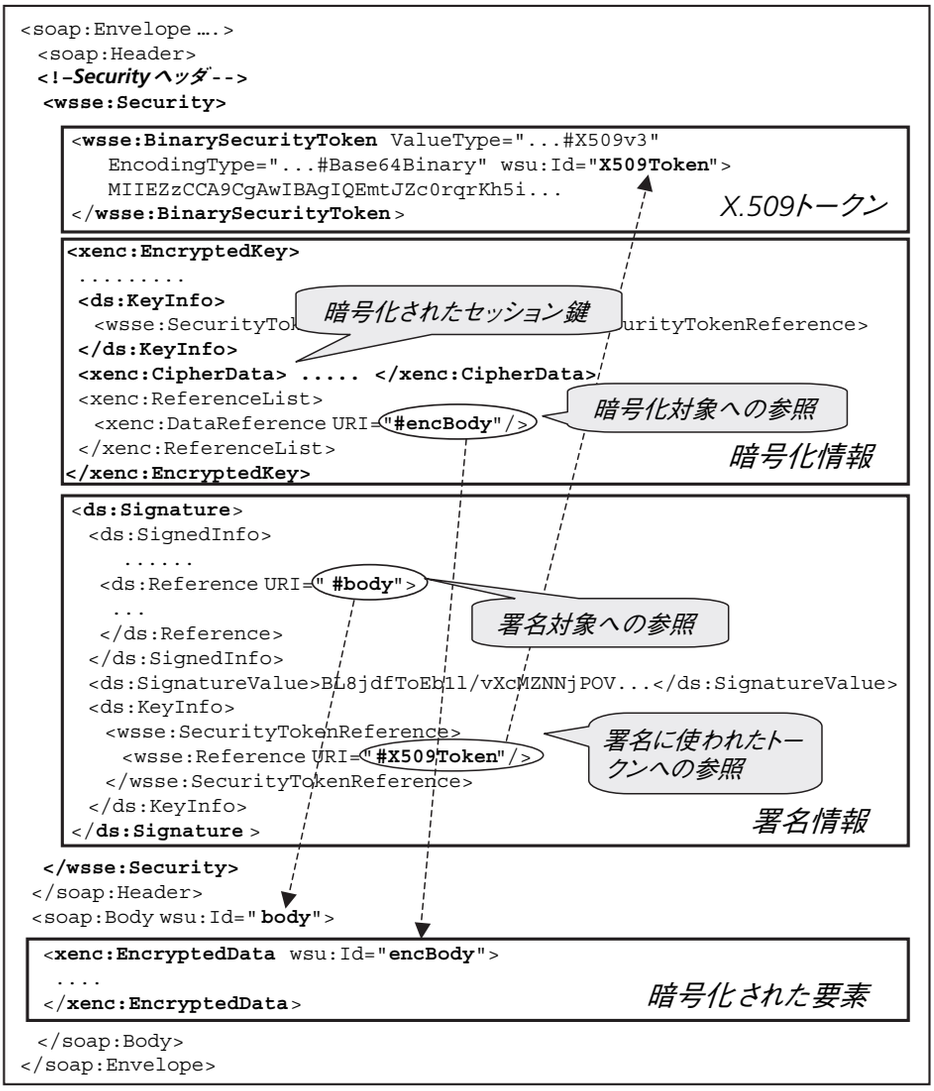

図 -5　WS-Security メッセージ構造

セキュリティトークン

セキュリティトークンは，ID ／パスワードや X.509 証明書，Kerberos チケット，共有鍵など，セキュリティに関連する情報を XML 形式で表現したものです．セキュリティトークンは鍵として参照されるほかに，リクエスタの認証情報として使われます．

リスト 1 を見てください．これは ID ／パスワードを表現するための Username トークンです．ID が Username 要素，パスワードが Password 要素で表現されます．リクエスタから Username トークンが送られると，Web サービスでは ID ／パスワードの検証によるリクエスタの認証が行われます．

ここでは分かりやすさのために，パスワードはそのまま Username トークンに入れられています．しかし実際の運用では，パスワードを保護するために Username トークンを暗号化することが推奨されています．

```
<wsse:UsernameToken>
  <wsse:Username>Alice</wsse:Username>
  <wsse:Password ....>
     weweYI3M....
  </wsse:Password>
</wsse:UsernameToken>
```

リスト 1.　Username トークンの例

セキュリティトークンにはいくつかの種類があり，表現しているセキュリティ情報ごとに具体的な XML 表現が決められています．図 -5 では署名・暗号化の鍵の表現としてリスト 2 に示したバイナリセキュリティトークンが使われています．これは，X.509 証明書などバイナリ形式の鍵を表現するためのセキュリティトークンです．BinarySecurityToken 要素には，バイナリ形式の鍵

をエンコードした値が挿入されます．エンコード形式はEncodingType 属性で指定され，リスト 2 では Base64 エンコードが指定されています. また, このセキュリティトークンが実際にどのような情報を表現しているものかを示すために，ValueType 属性が使われます．リスト 2 では，ValueType 属性に X.509v3 証明書を表す URI が指定されており，X.509v3 証明書をエンコードしたものであることが分かるようになっています.

```
<wsse:BinarySecurityToken
   ValueType="...#X509v3"
   EncodingType="...#Base64Binary">
    MIIEZzCCA9CgAwIBAg5i...
</wsse:BinarySecurityToken>
```

リスト 2.　X.509 トークンの例

署名情報

署名に関連する情報は Signature 要素で与えられます．この要素は XML 署名仕様で定められているものです．この下には SignedInfo 要素，SignatureValue 要素，KeyInfo 要素が入ります.

SignedInfo 要素では，この署名で使われているアルゴリズムの情報や署名対象が書かれています．また，この子要素である Reference 要素により，署名対象の要素が参照されています．図 -5 の例では，Reference 要素の URI 属性の値は“#body”です．これは，Id 属性の値に“body”を持つ要素，つまり Body 要素が署名対象であるということです.

SignatureValue 要素では，XML 署名の署名値が指定されます．本稿では署名検証方法の詳細は省略しますが，この値が改ざんされていないことを検証することで，署名検証が行われます.

KeyInfo 要素は署名に使われている鍵を指定する要素です．この図 -5 の例では，SecurityTokenReference 要素により，鍵として使われているセキュリティトークンを参照しています．ここでは，SecurityTokenReference 要素の子要素である Reference 要素の URI 属性の値が“#X509Token”ですので，Id 属性の値に“X509Token”を持つセキュリティトークンが署名の鍵として使われているということを表しています.

暗号化情報

暗号化に関連する要素も，署名と同じく Security ヘッダに入れられます．ここには EncryptedKey 要素が入れられ，暗号化に使われたセッション鍵が暗号化されて入っています．また，暗号化対象の要素は暗号化された要素に置き換えられます.

EncryptedKey 要素の子要素の KeyInfo 要素では，Web サービスの公開鍵を指すセキュリティトークンが参照されます．この公開鍵で暗号化されたセッション鍵は，CipherData 要素に入れられています．この例の KeyInfo 要素は，署名の場合のようにメッセージに挿入されているセキュリティトークンを内部参照するのではなく，外部の鍵を参照しています.

ReferenceList 要素では，暗号化対象の要素への参照を持っています．図 -5 の例では，DataReference 要素の URI 属性の値が“#encBody”ですので，Id 属性の値に“encBody”を持つ要素が暗号化された要素であることが分かります．暗号化された要素は EncryptedData 要素で置き換えられます.

EncryptedData 要素を復号化するには，EncryptedKey 要素を復号化する必要があります．このための秘密鍵を持っている Web サービスだけが CipherData 要素を復号化し，セッション鍵を取得でき，EncryptedData 要素を復号化することができることになります.

Webサービス連携におけるセキュリティ

最初に，Web サービスセキュリティでは End-to-End で Web サービスのセキュリティを確保することができると説明しました．そのための Web サービスセキュリティの特徴として，異なるセキュリティ機構を統合して扱うことができるということが挙げられます．具体的な例を用いて見てみましょう.

図 -6 を見てください．ここでは，複数の Web サービスが連携して動作する旅行予約サービスを考えます．旅行予約サービスは，旅行代理店サービスから呼び出される飛行機予約サービス，ホテル予約サービスなどの複数のサービスから構成されています.

旅行代理店サービスがメッセージを署名・暗号化して送信する場合，飛行機予約サービスが採用している鍵を使う必要があります．しかし，旅行代理店サービスと他のサービスは独立したサービスですから，それぞれのセキュリティ機構が異なることは容易にあり得ます．すなわち，図 -6 のように，飛行機予約サービスはセキュリティ機構として PKI を採用しているが，ホテル予約サービスは Kerberos を採用している，という場合です．ここで，旅行代理店サービスも独自のセキュリティ機構を採用しているとすると，それぞれのサービスが採用して

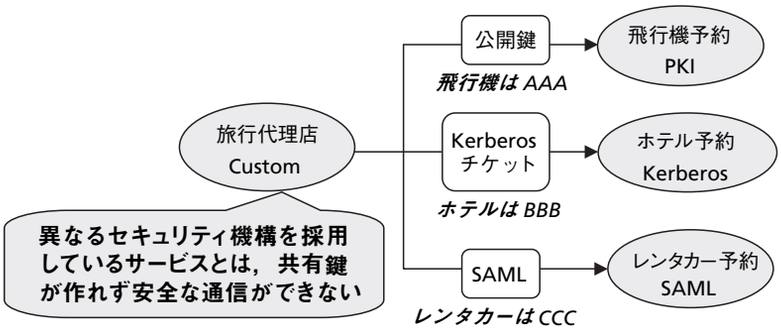

図 -6
異なるセキュリティ機構に基づく Web サービス

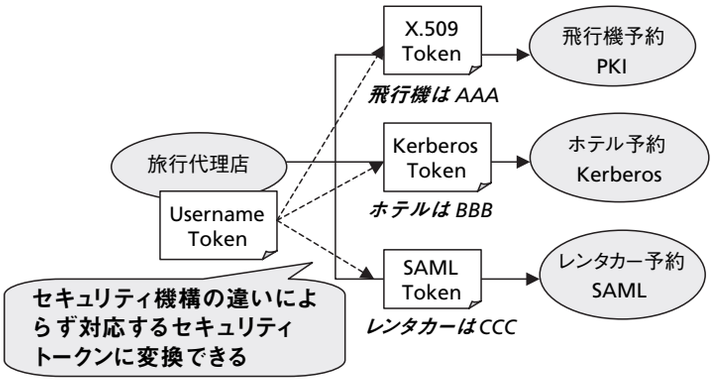

図 -7
異なるセキュリティ機構の Web サービスセキュリティによる統合

いるセキュリティ機構ごとに鍵の形式が異なるため，他のサービスとの共通鍵を使うことができません．したがって，このままでは異なるセキュリティ機構を採用しているサービスとは安全な通信ができなくなってしまいます．

このようなサービスに対し，Web サービスセキュリティを適用することを考えてみましょう．Web サービスセキュリティでは，鍵の情報を抽象化してセキュリティトークンとして表現します．セキュリティトークンは，PKI では X.509 トークン，Kerberos では Kerberos トークンというように，セキュリティ機構ごとにセキュリティトークンの表現が決められています．各 Web サービスへ署名・暗号化したメッセージを送るには，受信側のサービスが採用しているセキュリティ機構に対応したセキュリティトークンを使って署名・暗号化する必要があります．セキュリティトークンは異なる種類のセキュリティトークンに変換することができるため，受信側の Web サービスが対応しているセキュリティトークンに変換し，それを使って署名・暗号化することで安全にメッセージを交換することが可能になります．

図 -7 のように，連携する複数のサービスがサポートしているセキュリティ機構が異なっているにもかかわらず，それらを統合して End-to-End セキュリティが確保できるということが，Web サービスセキュリティを用いる利点であるといえます．

セキュリティトークンの交換

それではどのようにセキュリティトークンが交換できるのでしょうか．「Web サービスのセキュリティ：アーキテクチャとロードマップの提案」には，セキュリティトークンサービスという別の Web サービスを介してセキュリティトークンを交換・取得するシナリオが例示されています．このシナリオを旅行予約サービスにあてはめたものが**図 -8** です．

旅行代理店サービスは，飛行機予約サービスとのメッセージ交換に必要なセキュリティトークンの発行を，セキュリティトークンサービスに依頼します．この際，旅行代理店は自分のセキュリティトークンを添付します．セキュリティトークンサービスでは，添付されたセキュリティトークンにより旅行代理店が認証されたあと，飛行機予約サービスで使われる新しいセキュリティトークンが発行されます．このように，セキュリティトークンサービスを介したセキュリティトークンの交換により，必要なセキュリティトークンを取得し，WS-Security を適用することができるのです．

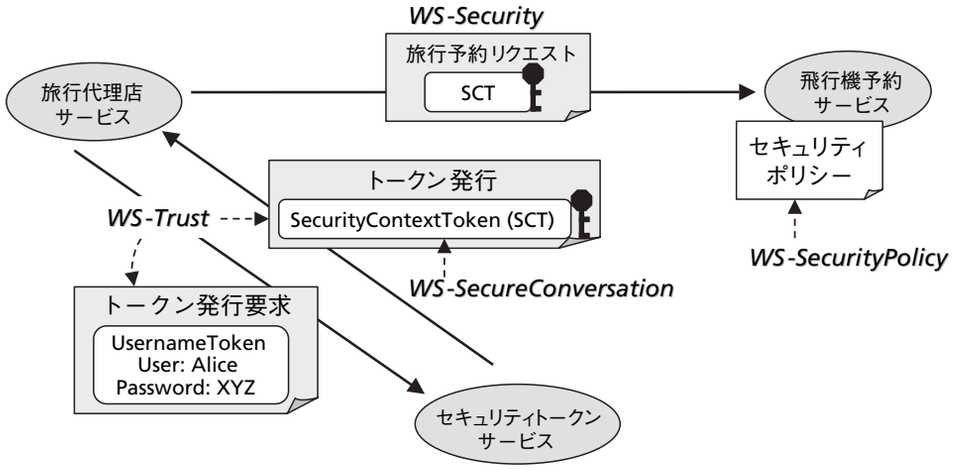

図 -8
セキュリティトークンサービスによるトークン発行要求

旅行代理店サービスとセキュリティトークンサービス間で交換される，セキュリティトークン発行依頼のためのメッセージ形式は，WS-Trust 仕様により決められています．また，WS-SecureConversation では，信頼が確立されたセッションで一時的に共有する共有鍵を表すセキュリティトークンの形式を定義しています．これをセキュリティコンテキストトークンと呼んでいます．

また，図 -8 で示したようなシナリオを可能にするには，飛行機予約サービスが要求するセキュリティトークンの種類や仲介に利用するセキュリティトークンサービスについて，旅行代理店サービスが知る必要があります．このようなサービスが要求しているセキュリティの条件は，セキュリティポリシーにより公開する必要があります．WS-SecurityPolicy は，Web サービスのポリシーを記述するためのフレームワークである WS-Policy を元に，セキュリティポリシーの記述方法を定めたものです．

次回はこれら 3 つの WS-SX 関連仕様についてご紹介します．

参考文献

1）Web Services Security : SOAP Message Security 1.1 (WS-Security 2004)．http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf
2）Web サービスのセキュリティ：アーキテクチャとロードマップの提案．http://www.ibm.com/jp/developerworks/webservices/020607/j_ws-secmap.pdf
3）OASIS WS-SX TC Webページ．http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-sx
4）WS-Trust 1.3 Committee Draft 01．http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-spec-cd-01.pdf
5）WS-SecureConversation 1.3 Committee Draft 01．http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/ws-secureconversation-1.3-spec-cd-01.pdf
6）WS-SecurityPolicy 1.2 Committee Draft 02．http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/ws-securitypolicy-1.2-spec-cd-02.pdf

（平成 19 年 4 月 27 日受付）

佐藤史子（正会員）
sfumiko@jp.ibm.com

2001 年東京工業大学理工学研究科基礎物理学専攻修了．同年日本 IBM（株）東京基礎研究所入所．現在は，Web サービスセキュリティ，サービスコンポーネントアーキテクチャのセキュリティ関連の研究に従事．