XML Consortium

OASIS Web Serivces Securityの概要と今後の状況

2005年6月7日
XMLコンソーシアム セキュリティ部会
西村 利浩　富士通株式会社）

アジェンダ

- OASIS WSS TCの全体的な状況
- 各仕様解説(エンハンス予定機能も含めて)

TCの状況

- OASIS Web Services Security Technical Committee

Web Services Security TC

- OASIS Web Services Security Technical Committee (WSS TC)
  - 2002/7/23にTC発足のアナウンス
  - 実際の活動は2002/9/4,5のF2Fから
- 目的
  - Webサービスのセキュリティの技術基盤を作る
- メンバー
  - 31の組織、団体からのメンバーと個人メンバーと合わせて46人がVoting Member/Memberとして活動中(2005/6/1現在)
  - 日本企業では日立と富士通がTC発足当初から参加
- 活動形態
  - メーリングリストによる議論と隔週の電話会議による議論
  - 2003/6の第3回F2F以降、F2F meetingはしていない
  - メーリングリストの内容、会議の議事録はすべて公開
  - ドラフト仕様書、問題点リストなども公開

WSS TCのページ

Charter
設立綱領

Membership
メンバー一覧

Email Archive
メーリングリストに流れたメール

Documents
ドラフトなど

Minutes
電話会議、F2Fの議事録

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss

成果物(OASIS Standard)

- OASIS Web Services Security 1.0 (WS-Security 2004) [2004年4月承認]
  - SOAP Message Security (コア仕様)
  - Username Token Profile
  - X.509 Token Profile
- 追加トークンプロファイル[2004年12月承認]
  - SAML Token Profile
  - Rights Expression Language (REL) Token Profile
- 最初の3仕様に対する正誤表(Errata)もCommittee Draftとして公開

V1.1の状況

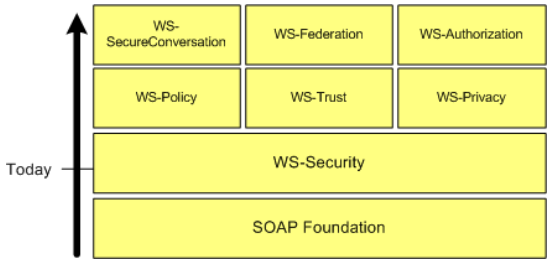
- V1.0終了後作業開始
- V1.0向けに検討されていたSAML V2 Token、Kerberos Token、SwAのプロファイルもV1.1に合わせて標準化予定
- V1.1には以下の7つの仕様が含まれる予定
  - SOAP Message Security (コア仕様)
  - Username Token Profile
  - X.509 Token Profile
  - SAML Token Profile
  - Rights Expression Language (REL) Token Profile
  - Kerberos Token Profile
  - SOAP Messages with Attachments (SwA) Profile
- 6月14日の電話会議でCommittee Draftとして承認するための投票を予定
- さらに2回の電話会議でPublic Review開始を目標
- 追加機能に関する相互運用性テストも計画

未検討の仕様(1)

- 提案されたが現在は検討されていない仕様
  - Minimalist Profile (MProf)
    - PDAなどリソースが限られているプラットフォームでWSSを利用する際に実装されるべきサブセットを規定
    - 2003年3月に提案されたが進展なし
    - 今後議論される可能性あり
  - XCBF Token Profile
    - OASIS StandardとなっているXCBF (XML Common Biometric Format)のデータをトークンとして扱うプロファイル
    - 2002年11月にXCBF TCの議長(当時)から提案
    - エディタ不在でWSS TCのページからも削除

未検討の仕様(2)

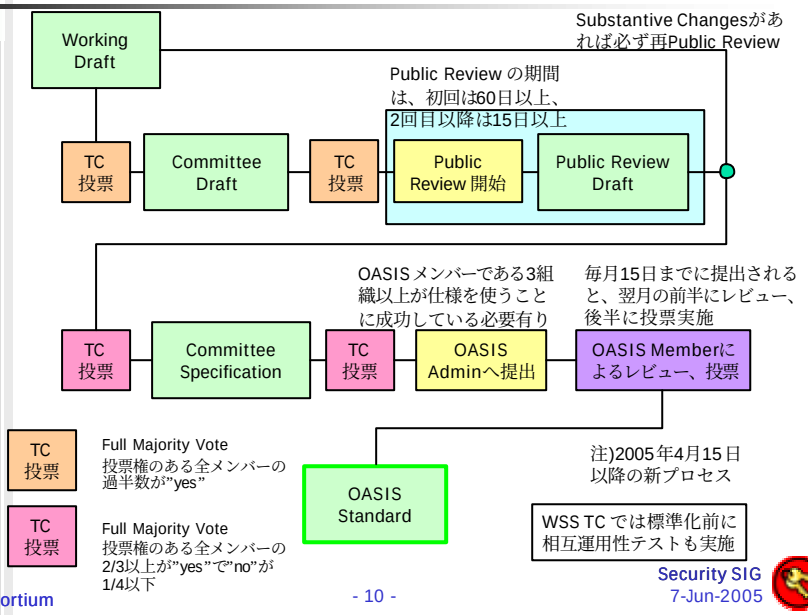
- IBMとMicrosoftのホワイトペーパーに出ていた仕様群



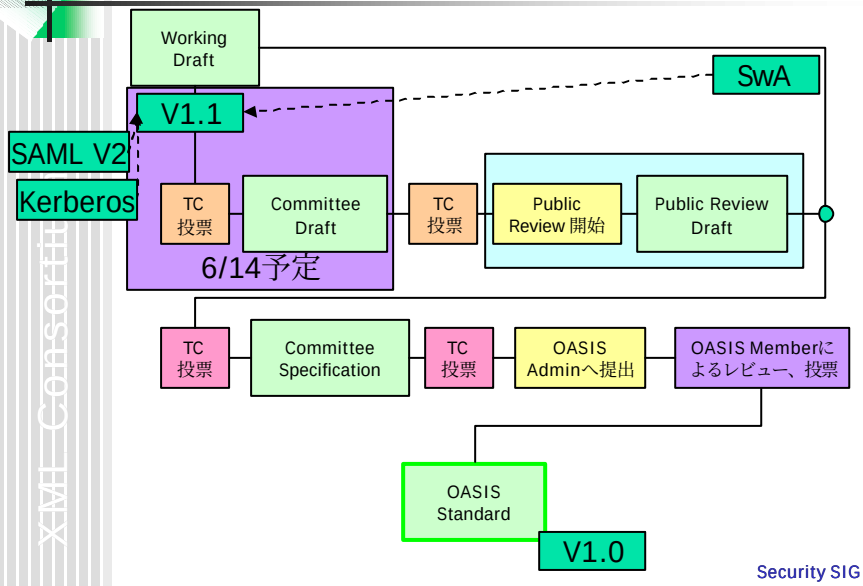
Security in a Web Services World: A Proposed Architecture and Roadmap
(http://msdn.microsoft.com/library/en-us/dnwssecur/html/securitywhitepaper.asp)

- このホワイトペーパー自体もWSS TCには提出されず
- WS-Security以外の6つの仕様は標準化団体に未提出
- WS-Authorization、WS-Privacyは仕様の公開も未
- (参考:MicrosoftはWS-Policyをメタデータ仕様として位置付け)

[参考]OASISの標準化プロセス



[参考]OASISの標準化プロセス



WS-SecurityかWSSか?(1)

- IBM、Microsoft、VeriSign の3 社による仕様の名称
  - Web Services Security (WS-Security) Version 1.0
- OASISでのTC設立綱領
  - Name of the TC:
    - OASIS Web Services Security Technical Committee (WSS)
  - The OASIS Web Services Security TC will:
    - 1. Accept as input the Web Services Security (WS-Security) specification published by IBM, Microsoft, and VeriSign on April 11th 2002 [1] and subsequent submitted documents.
    - 2. Produce as output a specification for Web Services Security.
- 最初のF2F以降
  - 仕様書名についての議論があり「Web Services Security: xxxxxx」という形にすることは決まったが、略称に関しては特別な議論はない (大体は「WSS」を使用)

WS-SecurityかWSSか?(2)

- 2004/2/24の電話会議での議論
  - WS-SecurityはTCへの初期入力となった仕様で、現在の仕様へと発展した
  - 現在の仕様はオリジナルのWS-Securityとは相互運用性がなく、それはWeb Services Securityと呼ばれる。略称はWSSである
  - 外部で現仕様をWS-Securityと呼ぶこともあるが、TCがそれをコントロールすることはできない
- 2004/3/9の電話会議での議論
  - 仕様書名とWS-Securityの関係について混乱を招くというOASIS事務局からの指摘を踏まえ以下の結論
    - 仕様書の題名に、カッコ付きで「(WS-Security 2004)」を追加
    - 仕様書に、オリジナルのWS-Securityに対して新たな作業をした成果だという記述を追加
      - This OASIS specification is the result of significant new work by the WSS Technical Committee and supersedes the input submissions, Web Service Security (WS-Security) Version 1.0 April 5, 2002 and Web Services Security Addendum Version 1.0 August 18, 2002.

WS-SecurityかWSSか?(3)

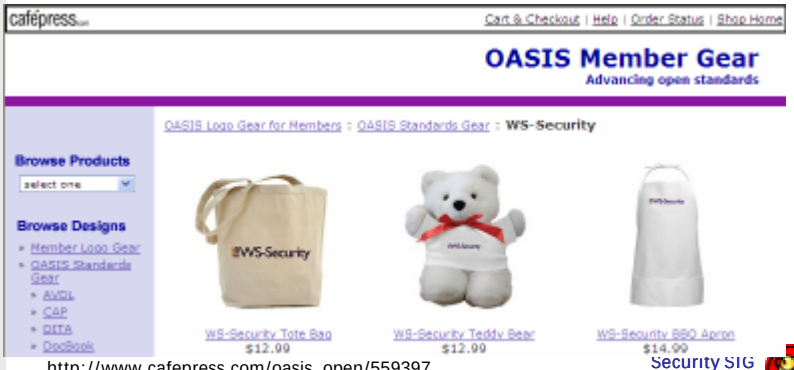
http://www.oasis-open.org/news/oasis_news_04_19_04.php

WS-SecurityかWSSか?(4)

http://www.oasis-open.org/news/oasis_news_04_20_05.php

WS-SecurityかWSSか?(5)

- 結局・・・
  - TCの名前の略称はWSS TC
  - 仕様の名前の略称はTC外ではWS-Securityが優勢(WSSでも可)
  - ただし、「WS-Security V1.0」は使わないのが懸命



http://www.cafepress.com/oasis_open/559397

仕様解説

- Web Services Security

仕様書構成

- Web Services Security:

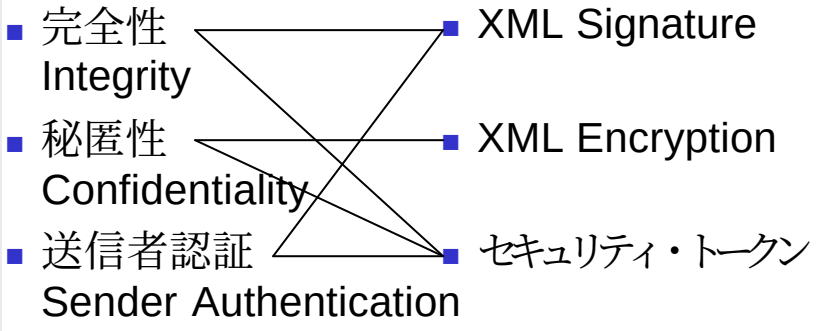
種類	仕様名	V1.0	V1.1
コア	SOAP Message Security	○	○
トークン・プロファイル	Username Toke Profile	○	○
	X.509 Certificate Token Profile	○	○
	SAML Token Profile	○追加	○
	Rights Expression Language (REL) Token Profile	○追加	○
	Kerberos Token Profile		○
他のプロファイル	SOAP Messages with Attachments (SwA) Profile		○

仕様解説

- Web Services Security:
  SOAP Message Security 1.0
  (WS-Security 2004)
  OASIS Standard 200401, March 2004

提供されるセキュリティ機能と手段

- SOAPメッセージに対する以下のセキュリティを提供



Securityヘッダ

- セキュリティ関連情報は<wsse:Security>ヘッダ・ブロックに記述

```
<S:Envelope>
  <S:Header>
    <wsse:Security>
      署名関連情報
      暗号化関連情報
      セキュリティトークン
      タイムスタンプ
    </wsse:Security>
  </S:Header>
  <S:Body>
  ...
  </S:Body>
</S:Envelope>
```


セキュリティ・トークン

- 申告の集まり(1つ以上)を表現
- 署名・暗号化の「鍵」を示すためにも利用される

- 利用者名トークン
  - 利用者名の提示
  - <wsse:UsernameToken>要素で指定
- バイナリ・セキュリティ・トークン
  - バイナリ符号化された（非 XML形式の）セキュリティ・トークンの提示
    - X.509証明書、Kerberos チケットなど
  - <wsse:BinarySecurityToken>要素で指定
- XML トークン
  - XML形式のセキュリティ・トークンの提示
    - SAML アサーション、REL ライセンスなど
  - <wsse:Security>ヘッダ・ブロックに直接挿入

トークン参照

- どこかに存在するセキュリティ・トークンを参照
- <wsse:SecurityTokenReference>要素で指定
  - <ds:KeyInfo>の子要素としても利用可

- 直接参照
  - URIを利用してトークンを直接参照
  - <wsse:Reference>子要素で指定
  - 同一文書内のトークンはID属性を利用
- 鍵識別子
  - トークンを表現するバイナリ符号化された鍵識別子でトークンを参照
  - <wsse:KeyIdentifier>子要素で指定
- 鍵名
  - セキュリティ・トークンの名称で参照
  - <ds:KeyName>要素を利用
- 埋め込まれた参照
  - 任意のセキュリティ・トークンを直接埋め込む
  - <wsse:Embedded>子要素で指定

署名

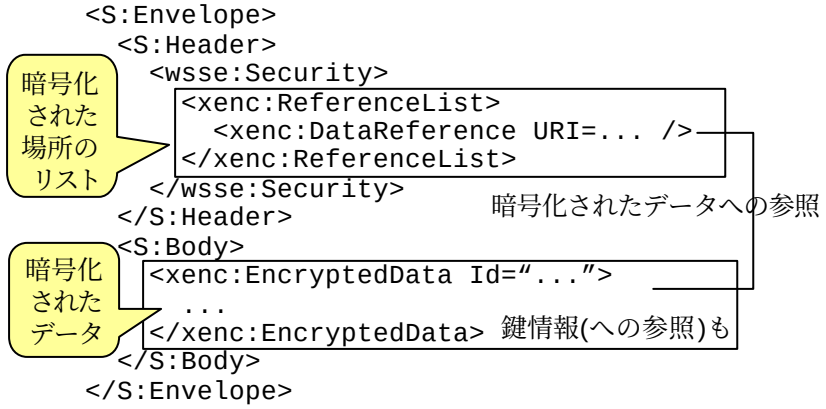
- XML Signatureの<ds:Signature>を<wsse:Security>ヘッダ・ブロックで利用

```
<S:Envelope>
  <S:Header>
    <wsse:Security>
      <wsse:BinarySecurityToken ...>
        ...
      </wsse:BinarySecurityToken>
      <ds:Signature>
          鍵情報(セキュリティ・トークン)への参照
          署名対象への参照
      </ds:Signature>
    </wsse:Security>
  </S:Header>
  <S:Body wsu:Id="body">
  ...
  </S:Body>
</S:Envelope>
```

署名情報

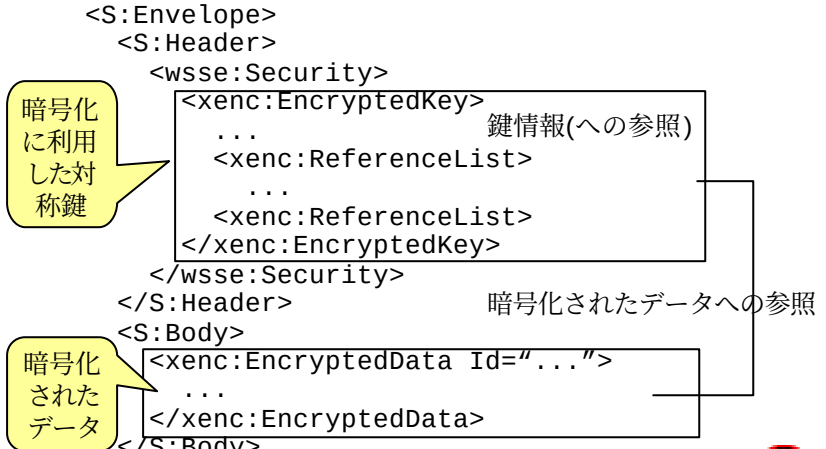
暗号化(1)

- XML Encryptionの<xenc:ReferenceList>を<wsse:Security>ヘッダ・ブロックで利用
- 暗号化されたデータは<xenc:EncryptedData>で表される



暗号化(2)

- XML Encryptionの<xenc:EncryptedKey>を<wsse:Security>で利用
- 暗号化されたデータは<xenc:EncryptedData>で表される



仕様解説

- Web Services Security:
  SOAP Message Security
  1.1でのエンハンス
- **(注: この内容は最終仕様確定時には変更となっている可能性があります)**

EncryptedData トークン

- <wsse:Security>ヘッダ・ブロック中のトークンの暗号化を許す
- <xenc:EncryptedData>が<wsse:Security>直下に置かれる
- トークンへの参照は<xenc:EncryptedData>要素に対してではなく、復号後のトークンに対してなされる

Encrypted Keyの参照

- V1.0では<xenc:EncryptedKey>要素は暗号化に利用した対称鍵を暗号化された形で同一メッセージ中で運ぶために利用
- 他の利用方法
  - EncryptedKeyを同一メッセージ内の暗号的操作(署名)のために利用
  - EncryptedKeyを2者間で交換されるその後のメッセージで暗号的操作のために利用
- EncryptedKeyを参照する仕組みを規定
  - 同一メッセージ中のEncryptedKeyの参照は、<xenc:EncryptedKey>のID属性を利用
  - 同一メッセージにないEncryptedKeyの参照は、<xenc:CipherValue>の値のSHA1値を鍵識別子として指定

署名確認

- 送信者が送信した署名付きメッセージを受信者が確かに受け取ったことを、送信者が確認できる手段を提供
- 受信者は返答時に<wsse:Security>ヘッダ・ブロックに<wsse11:SignatureConfirmation>要素を置く
  - 受信者は受信したメッセージ中の<ds:Signature>の<ds:SignatureValue>の値を<wsse11:SignatureConfirmation>のValue属性に入れて返す
  - もともとの署名が暗号化されていたなら、返答する<wsse11:SignatureConfirmation>も暗号化する

SOAPヘッダの暗号化

- 1つのSOAPヘッダ・ブロック全体を暗号化すると、mustUnderstand、actor (SOAP1.1)、role (SOAP1.2)属性などの属性が見えなくなってしまう
- <wsse11:EncryptedHeader>要素を規定
  - 元のヘッダ・ブロックの代わりに挿入
  - <xenc:EncryptedData>要素を1つ含む
  - mustUnderstand、actor、role属性を<xenc:EncryptedData>を参照する<wsse:Security>もしくは暗号化前のヘッダ・ブロックから引き継ぐ

仕様解説

- トークン・プロファイルの説明の前に

トークン・プロファイルの主な内容

- トークンの説明
- トークンのタイプを表すURI
- 鍵識別子のタイプを表すURI
- <wsse:Security>ヘッダ・ブロックへのトークンの置き方
  - そのまま置く(XML トークン)
  - <wsse:BinarySecurityToken>として置く
- トークンの参照方法
  - 直接参照
  - 鍵識別子による参照
  - 鍵名による参照
  - 埋め込まれた参照
  - その他?
- 署名、暗号の鍵として利用(<ds:KeyInfo>から参照)する際の解釈
- 利用例
- エラーコード
- 脅威モデルと対抗手段
- その他注意事項、トークン固有の説明など

- 注)すべてのプロファイルですべての項目を規定しているわけではない

仕様解説

- Web Services Security
  UsernameToken Profile 1.0
  OASIS Standard 200401, March 2004

UsernameToken

- WSS:SOAP Message Security(コア)仕様で導入された<wsse:UsernameToken>要素
- SOAPメッセージ送信者のユーザ名を示すために利用可能
- <wsse:Security>ヘッダ・ブロックの直下に置く
- WSSコアでは
  - 利用者名を指定する<wsse:Username>子要素のみを規定
  - 任意の属性、要素を許す拡張性を提供

ユーザ名とパスワード

- UsernemeToken Profileでは、<wsse:UsernameToken>にパスワードを含めることができるよう拡張
  →SOAPメッセージ送信者の認証
- <wsse:Username>要素以外に次の要素を子要素として規定
  - <wsse:Password>　パスワード
  - <wsse:Nonce>　ナンス(ランダムデータ)
  - <wsu:Created>　生成時刻
- パスワードの型式としてはテキスト(#PasswordText)とダイジェスト(#PasswordDigest)がある。ダイジェストの場合は次式で計算
  - Password_Digest = Base64 (SHA-1 (nonce + created + password))
  - →nonceとcreatedを利用することでリプレイ攻撃を阻止

トークン参照

- <wsse:UsernameToken>は鍵識別子や鍵名による参照はできない
- <wsse:UsernameToken>を<ds:KeyInfo>から参照することによって、パスワードから生成される鍵を示すことも可能であるが、その方法については規定しない

仕様解説

- Web Services Security: UsernameToken Profile 1.1でのエンハンス
- **(注: この内容は最終仕様確定時には変更となっている可能性があります)**

パスワードからの鍵の生成

- ユーザ名に関連付けられたパスワードから、MAC計算や暗号化に使われる鍵を生成する
- `<wsse:UsernameToken>`要素に次の要素を子要素として規定
  - `<wsse:Salt>` ランダムデータ
  - `<wsse:Iteration>` 計算回数
- 鍵は次のように計算を繰り返して生成
  - K1 = SHA1 (password + Salt)
  - K2 = SHA1 (K1)
    …
  - Kn = SHA1 (Kn-1)

仕様解説

- Web Services Security
  X.509 Certificate Token Profile
  OASIS Standard 200401, March 2004

X.509証明書

- ITU(国際電気通信連合)が勧告としているデジタル証明書の標準仕様
- 公開鍵と主体(Subject)との結びつきを認証局が証明する
- 完全性(署名)、秘匿性(暗号化)のための鍵としての利用と、送信者認証のための手段としての利用ができる
- X.509 Profileではトークンとして次の種類を規定
  - X509v3　　X.509 v3証明書
  - X509PKIPathv1　　X.509証明書のリスト
  - PKCS7　　X.509証明書とCRL(オプション)のリスト
- `<wsse:Security>`ヘッダ・ブロック内に直接置く場合はバイナリ・セキュリティ・トークン(`<wsse:BinarySecurityToken>`)を利用する

トークン参照

- 鍵識別子による参照
  - X.509 v3証明書のSubject Key Identifier拡張フィールドの値を利用して参照する
- URIによる参照
  - 同一文書内では`<wsse:BinarySecurityToken>`の`wsu:Id`属性を利用
- IssuerとSerial Numberによる参照
  - `<ds:X509IssuerSerial>`を利用する

署名で利用する際の注意点

- 1つの公開鍵に対して複数のX.509証明書を作ることが可能
  →証明書置き換え攻撃への対策として、証明書そのものもしくは証明書を一意に特定できる参照を署名に含めることを推奨
- 例) Subject Key Identifierは公開鍵のハッシュ値なので証明書が一意に特定できない可能性がある
  →鍵識別子(Subject Key Identifier)による参照を利用する場合はSTR Dereference Transformを使って証明書そのものに署名されるようにする

仕様解説

- Web Services Security:
  X.509 Certificate Token Profile
  1.1でのエンハンス
- **(注: この内容は最終仕様確定時には変更となっている可能性があります)**

Thumbprintによる参照

X.509 Thumbprint: X.509証明書全体をハッシュした値

- このThumbprintを鍵識別子として証明書を参照する

仕様解説

- Web Services Security:
  SAML Token Profile
  OASIS STANDARD, 01 Dec. 2004

SAMLアサーション

- Security Assertion Markup Language (SAML) V1.1 (2004年5月にOASIS Standard)
- SAMLアサーションは、Authorityによる主体(Subject)に関するセキュリティ関連情報の表明のXML表現
  - 認証アサーション
    「AliceをX時Y分にパスワードで認証した」
  - 認可決定アサーション
    「Aliceはファイルxxx.txtを閲覧ことを許されている」
  - 属性アサーション
    「Aliceのgroup属性はAdministratorである」
- SAML Token ProfileはSAML V1.1のアサーションをトークンとして利用する方法を規定
- メッセージ送信者の認証、認可、属性情報を伝えることができる
- SAMLアサーションを直接<wsse:Security>ヘッダ・ブロックに置く

SAMLアサーションの参照

- 鍵識別子による参照
  - SAMLアサーションのID(AssertionID属性)の値を利用
  - 同一メッセージ内にないアサーションを参照する場合にはIDに加え、<saml:AuthorityBinding>でアサーション入手先を指定
- 直接参照は規定しない
  - <saml:Assertion>要素にwsu:Id属性は追加できない
- 埋め込まれた参照は利用可能

SAMLアサーションの主体確認

- メッセージ受信者は、メッセージを送信(作成)したのがSAMLアサーションの主体であることを確認する必要がある
- WSS:SAML Profileでは、SAMLで規定された次の主体確認方法を利用
  - holder-of-key
    - アサーションの中に鍵の情報が埋め込まれる
    - メッセージ送信者はその鍵を利用してメッセージ内容に署名
    - holder-of-keyのアサーションを署名、暗号化の鍵として使用(<ds:KeyInfo>から参照)することも可能
  - sender-vouches
    - メッセージ受信者はメッセージ送信者と信頼関係をもち、受け取ったアサーションを信頼する
    - メッセージ内容とアサーションの改変の検出、メッセージ内容とアサーションの結びつけのため、送信者は両方に署名

仕様解説

- Web Services Security: SAML Token Profile 1.1でのエンハンス
- **(注: この内容は最終仕様確定時には変更となっている可能性があります)**

SAML 2.0 Token Profile

- 2004年12月
  SAML Token ProfileがOASIS Standardとして承認
- 2005年3月
  SAML 2.0がOASIS Standardとして承認
- 当初はSAML 2.0の標準化と並行してWSS 1.0向けのSAML 2.0 Pofileが作られていたが、WSS 1.1(SAML Token Profile 1.1)に統合される

SAML V1.1とV2.0の違い

- アサーション識別子を示す属性の名前の変更
  - AssertionID→ID
- 主体とステートメントの関係の変更
  - 1.1ではアサーション内のステートメント毎に主体を指定できた
    →2.0ではアサーション内には最大1つの主体
- AuthorityBindingの削除
  - AuthorityBindingを削除
  - 代わりに、HTTP requestでIDを指定することによりアサーションを入手できるエンドポイント(SAML URI Binding)のサポートすることが必須
    - https://saml.example.edu/assertion-authority?ID=abcde
- Attesting Entityの識別子
  - アサーションのサブジェクトとは違うエンティティがアサーションを提示することが見込まれる際に、そのエンティティを識別する情報(IPアドレス)を指定できるようになった

SAML V2.0アサーションの参照

- 鍵識別子による参照
  - SAMLアサーションのAssertionID属性ではなくID属性の値を利用
  - 同一メッセージ内にあるアサーションを参照する場合のみ利用してもよい
- 直接参照
  - 同一メッセージ内にないアサーションを参照する場合にSAML URI Bindingで規定されるURIで参照する
- 埋め込まれた参照は可能

仕様解説

- Web Services Security
  Rights Expression Language (REL)
  Token Profile
  OASIS Standard: 19 December 2004

RELライセンス

- ISO/IEC 21000-5:2004 Information technology
  -- Multimedia framework (MPEG-21)
  -- Part 5: Rights Expression Language
- 主体(principal)がリソースにおけるアクションを実行する権利をもっているかどうかを示すためのXML表現
- メッセージ送信者が、メッセージとライセンスを結び付けて送付することができる
- REL ライセンスを直接<wsse:Security>ヘッダ・ブロックに置く

RELライセンスの参照

- 直接参照(URI参照)
  - <r:license>要素はwsu:Id属性を追加できるので同一メッセージ内の参照はこれを利用する
  - ライセンス文書が置いてあるリモートの場所をURIとして指定して参照してもよい
  - <r:license>要素のlicenseId属性はURI値であるので、これを利用して参照してもよい

ライセンスの主体確認

- REL Token Profileでは、主体(Principal)は<r:keyHolder>によるもののみ利用可能
- <r:keyHolder>内で指定される鍵を使ってメッセージ内容に署名することによって、メッセージ送信者とライセンスを結びつける
- <r:keyHolder>内で指定される鍵(公開鍵)を使ってデータや鍵を暗号化することによって、ライセンス保持者のみがデータを見るようにすることができる

仕様解説

- Web Services Security
  Kerberos Token Profile
- **(注: この内容は最終仕様確定時には変更となっている可能性があります)**

TCでの状況

- 2004年7月27日付けWorking Draft 05
- 2005年5月にIBM、Microsoft、DataPowerの3社が相互運用性テスト
- 2005年5月16日付けWorking Draft 06
- WSS 1.1に合わせて標準化の方向

Kerberos

- MITで開発された、信頼する第三者機関による認証プロトコル
- Kerberos Profileでは、Kerberos V5のAP-REQメッセージ(およびAP-REQメッセージをGSS-APIで規定される方法でラップしたもの)をトークンとしてSOAPメッセージと一緒に送付する方法を規定
- AP-REQにはセッション鍵が含まれており
  - Kerberosセッション鍵(その鍵から導出される鍵)でHMACを利用することによってメッセージ認証
  - Kerberosセッション鍵(その鍵から導出される鍵)を使って暗号化(対称鍵暗号)
- Kerberos Tokenはバイナリ・セキュリティ・トークンとして<wsse:Security>ヘッダ・ブロックに置かれる

AP-REQ (KRB_AP_REQ)

- クライアントがチケット発行サービス(TGS: Ticket Granting Service)からチケットを受け取った後で、サーバへ送信するメッセージ
- チケット発行サービスから発行されたチケットと認証子(Authenticator)を含む
- チケット
  - クライアント名、ネットワークアドレス、サーバ名、クライアントサーバ間で利用されるセッション鍵を含む
  - サーバの秘密鍵で暗号化される
- 認証子
  - クライアント名、タイムスタンプ、予備のセッション鍵を含む
  - クライアントサーバ間のセッション鍵で暗号化される

Kerberos Tokenの参照

- 直接参照
  - `<wsse:BinarySecurityToken>`の`wsu:Id`属性を使って同じメッセージ内のトークンを参照
- 鍵識別子による参照
  - `<wsse:BinarySecurityToken>`に置かれるデータ(符号化する前のもの)のSHA-1ハッシュ値で指定
- 鍵名による参照は使われない

仕様解説

- Web Services Security
  SOAP Messages with Attachments
  (SwA) Profile
- **(注: この内容は最終仕様確定時には変更となっている可能性があります)**

TCでの状況

- 2004年10月、11月に2度のInterop
  - 一度はActional、Sun、IBM、Oracle
  - 二度目はActional、Sun、Oracle
- 2004年12月6日付けDraft 15
- 2004年12月23日付けCommittee Draft 01
- 2005年1月12日～2月11日 Public Review
- 2005年5月25日付けDraft 20

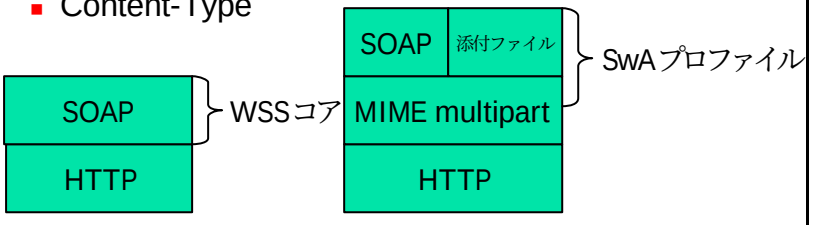
- WSS 1.1に合わせて標準化の方向

SOAP Messages with Attachments

- W3C Note 11 December 2000
- SOAPメッセージと一緒にアタッチメント(添付文書)を送るための仕様
- MIMEを利用
- WS-I Attachments Profile 1.0で採用
- SOAP 1.1のみが対象
- (SOAP 1.2での添付文書の仕様は、SOAP Message Transmission Optimization Mechanism (MTOM)が2005年1月にW3C Recommendation)

保護するレイヤ

- SwAプロファイルでは、必要であればコンテンツに関するMIMEヘッダ情報も保護できるようにする
  - Content-Description
  - Content-Disposition
  - Content-ID
  - Content-Location
  - Content-Type



アタッチメントの参照

- SOAP Messages with Attachments仕様ではアタッチメントを参照する方法として2つの方法を定義
  - Content-ID MIMEヘッダの値を利用して"cid:foo"という形式のURLを利用
  - Conetnt-Location MIMEヘッダのURLを利用
- SwAプロファイルではCID形式のURLのみを利用

MIME Part Reference Transforms

- RFC 2392の定義では、MIMEアタッチメントのURI参照にはアタッチメントに関連するMIMEヘッダが含まれる
- どの部分が処理対象かを明確にするために、MIME参照のためのTransformを規定
  - Attachment-Content-Signature-Transform
    - MIMEパートのコンテンツのみ+正規化
  - Attachment-Complete-Signature-Transform
    - MIMEパートのコンテンツとContent-*ヘッダ+正規化
  - Attachment-Ciphertext-Transform
    - MIMEパートのコンテンツのみ

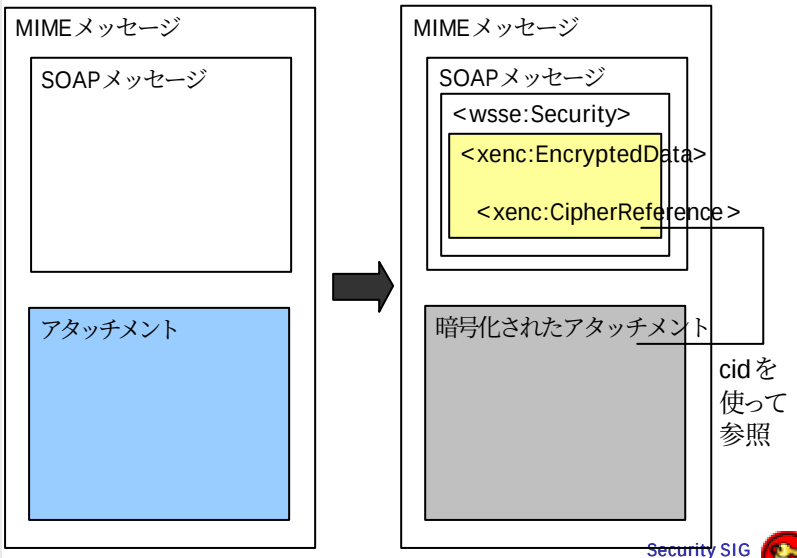
署名

- MIMEヘッダの正規化
  - Attachment-Complete-Signature-Transformを使う場合には、署名付与、署名検証の前に正規化が必要(SwAプロファイル内で規定)
- MIMEコンテンツの正規化
  - MIME type毎に正規化
  - XMLの場合はExclusive XML Canonicalization
- アタッチメント挿入攻撃
  - すべてのアタッチメントを署名対象にすることで、アタッチメントの削除は検出できるが挿入は検出できない
  - すべてのアタッチメントが署名対象であることを受信者が何らかの方法で知っていれば挿入も検出できる→このプロファイルの範囲外

暗号化

- アタッチメントのコンテンツ、もしくはコンテンツと一部ヘッダを暗号化し、暗号文をもともとのコンテンツと置き換える
- `<xenc:EncryptedData>`要素を`<wsse:Security>`ヘッダ・ブロックに置き、`<xenc:CipherReference>`でアタッチメント内の暗号文を参照する
  (ここでAttachment-Ciphertext-Transformを利用する)
- `<xenc:EncryptedData>`のType属性で、暗号化されたデータがコンテンツのみかヘッダも含むのかをURIにより指定する(TransformのURIとは別)
- WSSコアで規定している`<wsse:Security>`直下の `<xenc:ReferenceList>`、 `<xenc:EncryptedKey>`要素から `<wsse:Security>`に置かれた `<xenc:EncryptedData>`は、参照されてもよい

暗号化(イメージ)



まとめ

- Web Services Security仕様の標準化を進めているOASIS WSS TCの全体的な状況を説明した
- OASIS StandardとなっているWeb Services Security V1.0 (WS-Security 2004)と、現在検討されているV1.1でのエンハンス予定機能の概略を説明した
  - エンハンス予定機能については今後変更となる可能性がある点に注意のこと

付録

- WS-Security 2004の説明 (2005年1月13日第6回XMLコンソーシアムDayの資料より)

WS-Security 2004

- 概要
  - メッセージの完全性と秘匿性の為の、SOAPメッセージングの拡張
    - 広範囲のセキュリティ・モデルと暗号技術に適用して利用可能
  - セキュリティ・トークンとメッセージ内容の関連付け
    - セキュリティ・トークンの形式に依存しない、汎用的な仕組み

WS-Security 2004

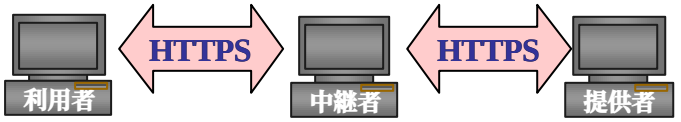
- TOC
  - 目標と要件
  - 記法と用語
  - メッセージ保護の仕組み
    - メッセージ・セキュリティ・モデル
  - ID参照
  - Securityヘッダ
  - セキュリティ・トークン
  - トークン参照
  - 署名
  - 暗号化
  - セキュリティ タイムスタンプ
  - エラー処理
  - セキュリティの考慮
  - 相互運用の注記
  - プライバシの考慮

目標と要件

- 目標
  - 安全なSOAPメッセージの交換
    - セキュリティ・トークンの受け渡し
    - メッセージの完全性
    - メッセージの秘匿性
- 要件
  - 幅広いセキュリティ・モデルのサポート
    - 複数のセキュリティ・トークン形式
    - 複数の信頼ドメイン
    - 複数の署名形式
    - 複数の暗号技術
    - End-to-Endのメッセージコンテントセキュリティ

SSLとの比較

- SSL: Point-to-Point
  - トランスポート層のセキュリティ
  - 中継者へのセキュリティ確保が困難



- WSS: End-to-End
  - メッセージコンテントのセキュリティ
  - 中継者にもセキュリティを確保

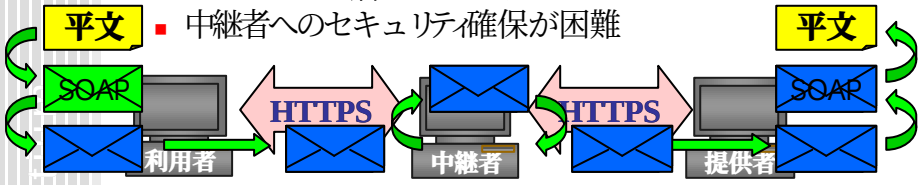
SSLとの比較

- SSL: Point-to-Point
  - トランスポート層のセキュリティ
  - 中継者へのセキュリティ確保が困難

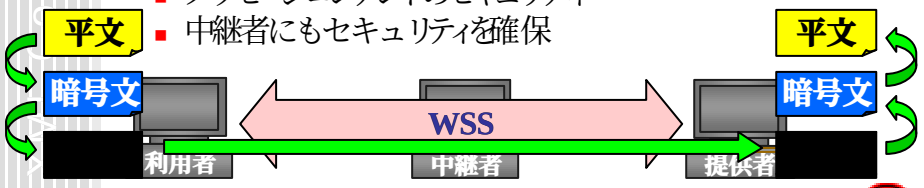
- WSS: End-to-End
  - メッセージコンテントのセキュリティ
  - 中継者にもセキュリティを確保

SSLとの比較

- SSL: Point-to-Point
  - トランスポート層のセキュリティ
  - 中継者へのセキュリティ確保が困難



- WSS: End-to-End
  - メッセージコンテントのセキュリティ
  - 中継者にもセキュリティを確保



メッセージセキュリティモデル

- セキュリティ上の脅威
  - 敵対者によるメッセージの改変・読み取り
  - 不正なセキュリティ申告
- SOAPメッセージの保護と認証
  - デジタル署名と組み合わせたセキュリティ・トークンの利用
  - XML-Signatureの利用（メッセージ完全性）
  - XML Encryptionの利用（メッセージ秘匿性）

ID参照

- 他のメッセージ要素への参照
  - 署名の参照
  - 署名とセキュリティ・トークンの関連付け
- メッセージの完全なスキーマを知らなくても要素の識別・参照が可能
  - グローバル「wsu:Id」属性
  - XML-Signature要素上のローカルID属性
  - XML Encryption要素上のローカルID属性

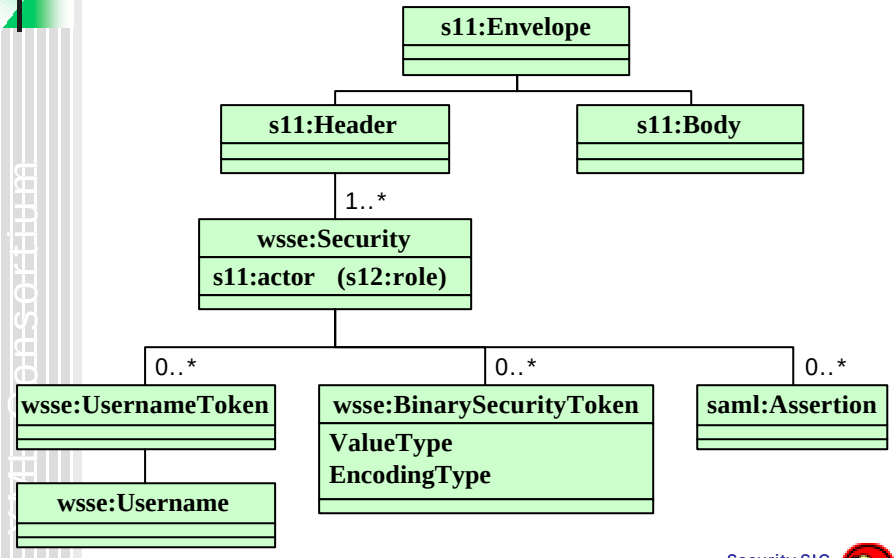
Securityヘッダ

- SOAP actor/role形式での特定の受信者を対象としたセキュリティ関連情報を付与
  - 複数のセキュリティ・トークン
  - 複数のセキュリティ・トークン参照
  - 複数の電子署名
  - 複数の暗号化情報
  - 0または1つのタイムスタンプ
- <wsse:Security>ヘッダ要素で指定
  - 複数のSecurityヘッダ部で同じactor/roleを指定することは出来ない
  - 1つのSecurityヘッダ部だけがactor/roleを省略できる
- 子要素を追加する場合は、既存の子要素の前に挿入する
  - メッセージを作成した際の処理順序が表わされる
  - 子要素間での前方参照がないため、受信側でヘッダ部に現れる順序に処理できることが保証される
  - 鍵運搬要素は、鍵を利用する要素より前に配置する

セキュリティ・トークン

- 利用者名 トークン
  - 利用者名の提示
  - <wsse:UsernameToken>要素で指定
- バイナリ・セキュリティ・トークン
  - バイナリ符号化された（非XML形式の）セキュリティ・トークンの提示
    - X.509証明書、Kerberos チケット など
  - <wsse:BinarySecurityToken>要素で指定
- XML トークン
  - XML形式のセキュリティ・トークンの提示
    - SAML アサーション、RELライセンス など
  - <wsse:Security> ヘッダに直接挿入

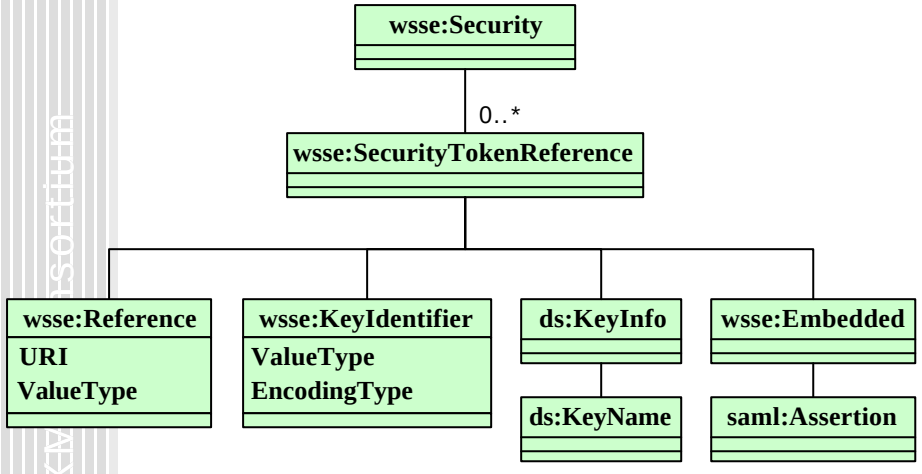
メッセージ構造



トークン参照

- どこかに存在するセキュリティ・トークンを参照
- <wsse:SecurityTokenReference>要素で指定
  - <ds:KeyInfo>の子要素としても利用可
- 直接参照
  - URIを利用して外部トークンを直接参照
  - <wsse:Reference>子要素で指定
- 鍵識別子
  - トークンを表現するバイナリ符号化された鍵識別子でトークンを参照
  - <wsse:KeyIdentifier>子要素で指定
- 鍵名
  - セキュリティ・トークンの名称で参照
  - <ds:KeyName>要素を利用
- 埋め込まれた参照
  - 任意のセキュリティ・トークンを直接埋め込む
  - <wsse:Embedded>子要素で指定

メッセージ構造



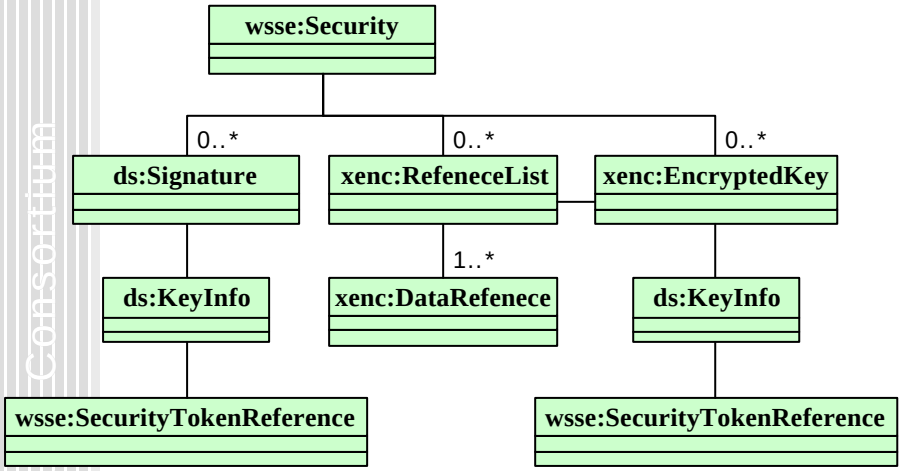
署名

- メッセージ完全性の実現
- セキュリティ・トークンの確認
- XML-Signature標準に対応
  - <ds:Signature>要素を利用
  - Detached形式で<wsse:Security>ヘッダに含める
  - Exclusive XML Canonicalization
  - SOAP Message Normalization
- STR Dereference Transform
  - <wsse:SecurityTokenReference>要素で参照されるセキュリティ・トークンへの署名手順を規定

暗号化

- メッセージ秘匿性の実現
- XML Encryption標準に対応
  - <xenc:RefernceList>要素を利用
  - <xenc:EncryptedKey>要素を利用

メッセージ構成



署名と暗号化の順序

- 処理した要素を順に前に挿入
  - 署名 ⇨ 暗号化

```
<wsse:Security>
    <xenc:EncryptedKey>….</xenc:EncryptedKey>
    <ds:Signature>….</ds:Signature>
</wsse:Security>
```

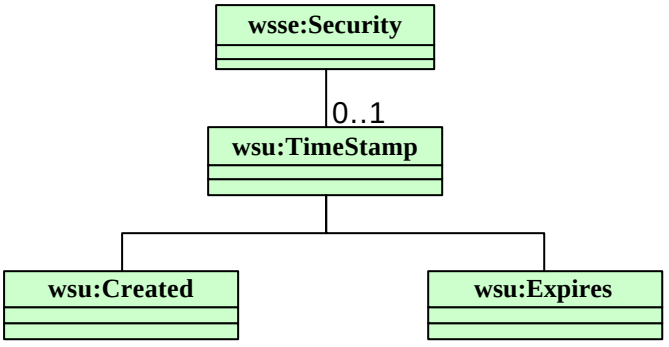
  - 暗号化 ⇨ 署名

```
<wsse:Security>
    <ds:Signature>….</ds:Signature>
    <xenc:EncryptedKey>….</xenc:EncryptedKey>
</wsse:Security>
```

セキュリティ・タイムスタンプ

- メッセージに含まれるセキュリティ情報の生成時刻および有効期限を表現
- <wsu:Timestamp>要素で指定
- 生成時刻
  - <wsu:Created>子要素で指定
- 有効期限
  - <wsu:Expires>子要素で指定

メッセージ構造



エラー処理

- SOAPのFault機構を利用
- wsse:UnsupportedSecurityToken
  - サポートされていないセキュリティ・トークン
- wsse:UnsupportedSecurityAlgorithm
  - サポートされていない署名または暗号方式
- wsse:InvalidSecurity
  - <wsse:Security>ヘッダ部のエラー
- wsse:InvalidSecurityToken
  - 無効なセキュリティ・トークン
- wsse:FailedAuthentication
  - 認証されないセキュリティ・トークン
- wsse:FailedCheck
  - 無効な署名 または 復号の失敗
- wsse:SecurityTokenUnavailable
  - 利用不可能なセキュリティトークン参照

セキュリティの考慮

- 一般的に考慮しなければならないこと
  - 新鮮さの保証
    - 再送の危険
    - 遅延したメッセージ
    - 安全な時刻同期を仮定したタイムスタンプに依存することの危険
  - デジタル署名と暗号の適切な利用
  - セキュリティ・トークンの保護
  - 証明書の検証
  - 外部の保護なしにパスワードを利用する危険
    - 辞書攻撃
    - パスワードから導出した鍵の危険性
  - 乱数の利用
  - 他のシステム構成要素を実装するセキュリティ機構との間の相互操作
  - 中間者攻撃
  - PKI攻撃

セキュリティの考慮

- 追加で考慮するべきこと
  - 再送攻撃への対応
    - 署名はそれだけでは認証を提供しない
    - 再送を検知できるような要素を署名対象に含める
      - タイムスタンプ、連続番号、有効期限、メッセージ相関関係
  - セキュリティ機構の組み合わせに起因する脆弱性
    - 電子署名を参考に平文推測攻撃で暗号文を解読
  - チャレンジ・レスポンスの利用
  - セキュリティ・トークンと鍵の保護（置き換え攻撃への対応）
    - 電子署名の利用
    - 署名と暗号に同じ鍵ペアを用いる
  - タイムスタンプとidの保護
    - 電子署名の利用