SOAP Message Security: Minimalist Profile (MProf)

Working Draft 1.5, Friday, 07 March 2003

Document identifier:

{draft}-{ WSS: SOAP Message Security }-{Minimalist Profile}-*{1.5}]* (Word) (PDF)

Location:

http://www.oasis-open.org/committees/wss

Editor:

Anthony Nadalin (drsecure@us.ibm.com)

Contributors:

Steve Anderson, OpenNetwork

Phillip Hallam-Baker, VeriSign

Alex Deacon, VeriSign

Frederick Hirsch, Nokia

Maryann Hondo, IBM

Takayuki Ito, IBM

Chris Kaler, Microsoft

Hiroshi Maruyama, IBM

Anthony Nadalin, IBM

Taiga Nakamura, IBM

Rob Philpott, RSA Security

Elliot Waingold, Microsoft

Abstract:

This document defines a subset of OASIS WSS: SOAP Message Security. This subset is intended to minimize the resource requirements of its implementation and maximize the performance, while keeping the interoperability with the base specification.

Status

This is an interim draft. Please send comments to the editors.

Committee members should send comments on this specification to the mailto:wss@lists.oasis-open.org list. Others should subscribe to and send comments to the wss-comment@lists.oasis-open.org list. To subscribe, visit http://lists.oasis-open.org/ob/adm.pl.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Security Services TC web page (**http://www.oasis-open.org/who/intellectualproperty.shtml**).

Table of Contents

Introduction

The OASIS Web Services Security : SOAP Message Security specification defines a very flexible way to secure SOAP messages and exchange security information using SOAP messages.

When WSS: SOAP Message Security is implemented on resource-limited platforms such as PDA's, it is desirable that constraints be imposed on the use of the core specification to ensure that the messages received by resource-limited platforms can be efficiently processed by them. A naïve implementation of WSS: SOAP Message Security can be very inefficient. The overhead of creating and verifying XML Signature with its associated "NodeSet" semantics could be a major source of this inefficiency. For example, if within the signature, signed data is referenced using an Xpath expression or and XSLT transform, then it will be very hard to verify the signature without first generating a NodeSet structure, which can be very costly. In many such cases, it is necessary to explicitly create a NodeSet data structure in memory and manipulate it according to the specified transforms. Canonicalization can be another source of inefficiency because of character encoding conversion and repeated serialization and deserialization required in the specification.

The goal of this document is to define a subset of WSS: SOAP Message Security that (1) allows stream processing on the receiving side of signed and encrypted messages, (2) requires no explicit canonicalization on the receiving side, and (3) requires no signature transforms requiring NodeSet evaluation. The subset, called MProf, is 100% compatible the WSS: SOAP Message Security specification. All the messages in this profile are valid WSS: SOAP Message Security messages.

This specification is intended to be applicable to both SOAP 1.1 and SOAP 1.2.

1 Notations and Terminology

This section specifies the notations, namespaces, and terminology used in this specification.

1.1 Notational Conventions

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

When describing abstract data models, this specification uses the notational convention used by the XML Infoset. Specifically, abstract property names always appear in square brackets (e.g., [some property]).

When describing concrete XML schemas, this specification uses the notational convention of WSS: Minimalist Profile (MProf). Specifically, each member of an element's [children] or [attributes] property is described using an XPath-like notation (e.g., /x:MyHeader/x:SomeProperty/@value1). The use of {any} indicates the presence of an element wildcard (<xs:any/>). The use of @{any} indicates the presence of an attribute wildcard (<xs:anyAttribute/>)

This specification is designed to work with the general SOAP message structure and message processing model, and should be applicable to any version of SOAP. The current SOAP 1.2 namespace URI is used herein to provide detailed examples, but there is no intention to limit the applicability of this specification to a single version of SOAP.

Readers are presumed to be familiar with the terms in the Internet Security Glossary

1.2 Namespaces

The following namespaces are used in this document:

Prefix	Namespace
ds	http://www.w3.org/2000/09/xmldsig#
S	http://www.w3.org/2001/12/soap-envelope
xenc	http://www.w3.org/2001/04/xmlenc#
wsse	http://schemas.xmlsoap.org/ws/2002/xx/secext
wsu	http://schemas.xmlsoap.org/ws/2002/xx/utility

1.3 Schema and WSDL Files

The schema for this specification can be located at:

```
http://schemas.xmlsoap.org/ws/2003/02/mprof
```

1.4 Terminology

We introduce the following terms that are used throughout this document:

MProf – Shorthand notation of this specification, Minimalist Profile of Web Services Security : SOAP Message Security

Exc-C14N – Shorthand notation of Exclusive Canonicalization

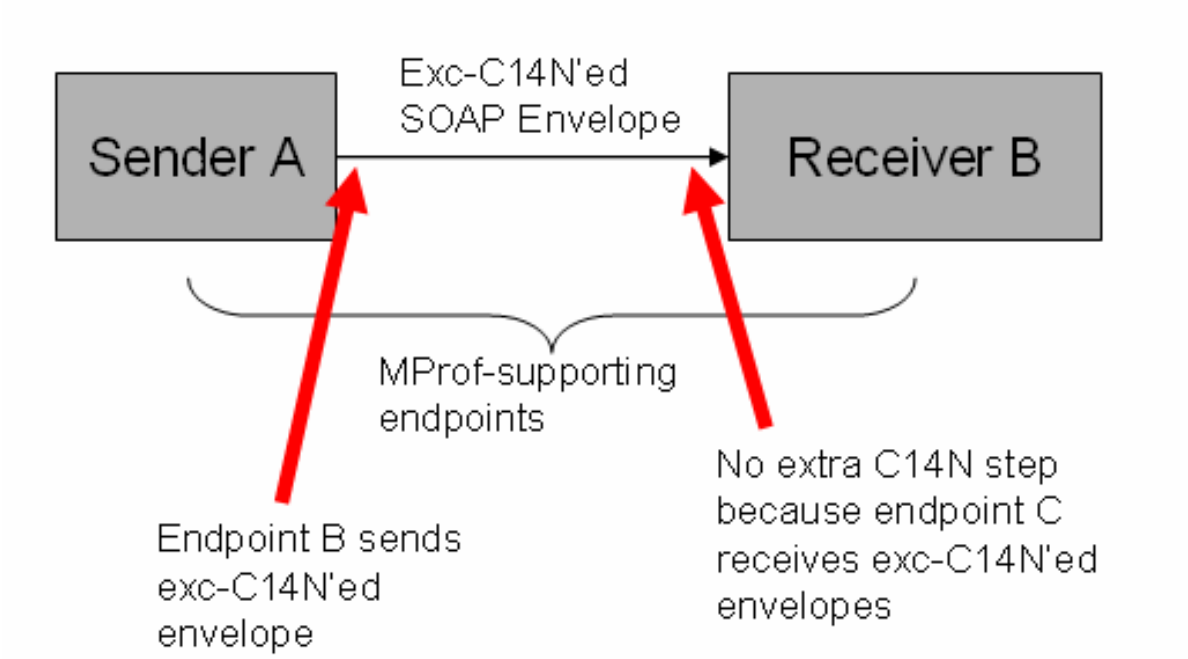
NodeSet – XPath NodeSet as defined in XPath

MProf transform – A transform algorithm that complies with this specification

Stable Exc-C14N element – An element in an exc-C14N'ed document that satisfies certain conditions (see Section 2.1.2).

2 MProf Messaging Model

In order to eliminate the necessity of performing canonicalization on the receiving side while preserving interoperability with other applications, the sender MUST prepare an already canonicalized message and intermediaries should preserve the canonicalization. We define a security header extension and processing rules that allow the receiver to know that the sender and intermediaries support MProf. This allows the receiver to expect all the incoming messages from this sender and intermediaries to be MProf messages.



When the sender and intermediaries "observe" MProf, the receiver can anticipate the received messages to have the following properties:

1. They are already canonicalized using Exclusive Canonicalization (we call it exc-C14N hereafter) and all the signed parts can be processed without computing exc-C14N explicitly. (Only exclusive canonicalization without comments may be used.)
2. There are no transforms that require explicit NodeSet computation.
3. The receiver knows elements to be signed in advance, so the receiver can start calculating the hash of the signed elements as soon as it encounters the start tag of the element. Therefore, the receiver can process these messages without expensive operations.

Another possible scenario is that even if the sender is a non-MProf endpoint, an intermediary may be able to convert ordinary WSS: SOAP Message Security messages into MProf messages if they satisfy certain properties described in this document.

2.1 MProf awareness

This document defines a new `wsse:MProf` header. This header is specified whenever a sender of a message asserts that the message is conformant with the profile and that the conformance MUST be maintained to the specified actor/role. For example, if the actor/role is 'next' then that actor/role MAY alter the message so as not to be conformant. If, however, there is a header targeted at the ultimate receiver then the message MUST maintain conformance until delivery at the ultimate receiver. The sender that is following this profile is REQUIRED to include the wsse:MProf header element on all messages. The mustUnderstand attribute MUST be set to the

value of "1" in `wsse:MProf` headers to ensure that recipients know that this is an MProf message and how to process it.

When an intermediary is the target of a MProf message it is required to process the message. If the intermediary does not support the MProf header the node MUST return a SOAP Fault to the sender and stop processing the received message.

Each MProf header MAY be signed, using the mechanisms defined in WS-Security to allow the recipient to verify that MProf processing is appropriate for the message as received from the previous party.

Any SOAP node that supports Web Services Security : SOAP Message Security should be able to receive MProf messages because MProf is a strict subset of Web Services Security : SOAP Message Security. However, some nodes may support MProf only. When such a SOAP node detects that a received message is a non-MProf compliant message, the node MUST return a SOAP Fault to the sender and stop processing the received message.

The following illustrates the syntax of this header:

```
<S:Envelope>
  <S:Header>
       <wsse:MProf Stable="..." S:mustUnderstand="1" >
       </wsse:MProf>
     ...
  </S:Header>
    ...
</S:Envelope>
```

The following describes the attributes and elements listed in the example above:

/wsse:MProf

This header element defines the method of messaging used. Use of the header indicates sender adherence to the requirements of this profile for the SOAP message.

/wsse:MProf/{any}

This is an extensibility mechanism to allow additional information based on a schema to be passed in an MProf element.

/wsse:Mprof/@Stable

This required QName attribute indicates the type of "canonicalization stability" to which this message conforms. Two forms of stability are described in subsequent specifications.

/wsse:Mprof/@S:mustUnderstand="1"

This required attribute indicates that recipient MUST understand the semantics of this profile in order to process the message.

/wsse:Mprof/@{any}

This is an extensibility mechanism to allow additional information based on a schema to be added as a MProf header attributes.

Note: This specification requires requires use of "0" and "1" for MustUnderstand, which is consistent with the WS-I Basic Profile.

2.2 MProf Specification

MProf messages MUST satisfy each of the conditions described below in the subsections.

2.2.1 Conditions on the <ds:Reference> element

In order to avoid computational intensive NodeSet operations, MProf messages are not allowed to have generic transformations in their <ds:Signature> elements. They are allowed to have optional XML Decryption transforms provided by the sender to enable simplified receiver processing, when signatures and encryption are combined. In addition, each <Reference> element is required to have an explicit transformation algorithm as the last transform, which is either exc-C14N (<ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>) or one of the MProf transforms defined later in this section. No other transforms are permitted.

Any <ds:Reference> element in an MProf message MUST have one of the following forms:

1. An intra-document URI, where the <ds:Reference> element has an URI="…" fragment attribute that has an intra-document URI value (i.e., in the form of "#id") whose referent appears after the enclosing <ds:Signature> element. Furthermore, the <ds:Reference> element has a <ds:Transforms> element, which has an optional decryption transform followed by <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>.
2. A 'same-document' reference, where the <ds:Reference> element has an empty URI attribute, that is, URI="". XPointer alternatives are not allowed. Furthermore, the <ds:Reference> element has a <ds:Transforms> element, which has an optional decryption transform followed by <ds:Transform Algorithm="*tfmalg*"/> where tfmalg is one of MProf transforms (described below).
3. An attachment URI, where the <ds:Reference> element has a cid: URI value in the case of multipart MIME. [SOAP].

In the same document reference (case 2), the transform algorithm MUST satisfy the following conditions:

- The input of the transform algorithm is the NodeSet of the entire SOAP envelope.
- The output of the transform algorithm is an octet string that is a subsequence of the exc-C14N form of the entire envelope. A subsequence of a string is a string obtained by removing one or more characters from the original string.

A transform algorithm satisfying these conditions is called *MProf transform*. For example, a transform that returns a concatenation of <wsu:Timestamp> header and the <S:Body> element in the exc-C14N'ed envelope is an MProf transform. The use of an MProf transform ensures that the receiving node can calculate the digest of the referent without applying any canonicalization. Appendix A of this specification defines a set well-known MProf transforms that SHOULD be supported by compliant implementations.

2.2.2 Restricted Stability form of XML Canonicalization

An MProf message MUST maintain exc-C14N state at the message's top level in order to avoid extraneous exclusive canonicalization operations. The message MUST also be idempotent with regard to exclusive canonicalization operations on signed parts. We define a restricted stable form of *exc-C14N* (wsse:StableRestricted) as an exc-C14N representation of an element that does not change when the entire document is exc-C14N'ed. This requires at least two things:

- The element's representation complies with exc-C14N (character encoding, white space normalization, etc).
- All the namespace prefixes explicitly used in the element MUST be declared within the element.
- If required by an XML processor, then if the same namespace URI is declared at an ancestor element, the namespace prefixes MAY be different.

A receiving SOAP endpoint does not need to compute exc-C14N on a signed element if it is a stable exc-C14N'ed element, because it is already exc-C14N'ed. A stable exc-C14N'ed element is a completely legal well-formed XML element so non-MProf-aware endpoint should have no problem processing it. In the following example where we assume that the whole envelope has been already exc-C14N'ed, elements <S:Envelope> and <tru:StockSymbol> are both stable exc-C14N'ed element because all the necessary namespace declarations are self-contained.

```
<S:Envelope xmlns:S="http://… ">
  ...
<S:Body>
  <tru:StockSymbol xmlns:tru="http://fabrikam123.com/payloads">
      QQQ
  </tru:StockSymbol>
</S:Body>
</S:Envelope>
```

On the other hand, <S:Body> is not stable because the namespace prefix S is not declared within this element's scope. If this element is to be signed, it's exc-C14N form will have an xmlns:S="http://... " declaration in the <S:Body> tag. This is not desirable since it uses the same prefix as the S:Envelope declaration.

Therefore, all the signed parts of an MProf message MUST be stable exc-C14N elements. This requirement has consequences on three places in a Web Services Security : SOAP Message Security message.

- The <ds:SignedInfo> element MUST be a stable exc-C14N'ed element.
- The value of a <ds:CanonicalizationMethod> element in a <ds:Signature> element MUST always be http://www.w3.org/2001/10/xml-exc-c14n#

- Any element that is referred to by a <ds:Reference> element through an intra-document reference in the URI attribute MUST be a stable exc-C14N'ed element. Note that this condition allows a signature on an entire element (a node and all of its descendant nodes) only when wsu:id is used for referring to the signed portion. When signing a node and its proper subset of the descendant nodes and all of its associated attributes, an MProf transform needs to be used.

In order to provide a stable body element for signing, after exc-c14n the previous example would be recast as follows:

```
<S:Envelope xmlns:S="http://… ">
  ...
<Sb:Body xmlns:Sb="…">
  <tru:StockSymbol xmlns:tru="http://fabrikam123.com/payloads">
      QQQ
  </tru:StockSymbol>
</Sb:Body>
</S:Envelope>
```

2.2.3 Safe Stability form of XML Canonicalization

An MProf message MUST maintain exc-C14N state at the message's top level in order to avoid extraneous exclusive canonicalization operations. The message MUST also be idempotent with regard to exclusive canonicalization operations on signed parts. We define a safe stable form of *exc-C14N* (wsse:StableSafe) as an exc-C14N representation of an element that does not change when the entire document is exc-C14N'ed. This requires at least two things:

- The element's representation complies with exc-C14N (character encoding, white space normalization, etc).
- All the namespace prefixes explicitly used in the QName element and/or attributes values MUST be declared within the element.

A receiving SOAP endpoint does not need to compute exc-C14N on a signed element if it is a safe stable form of exc-C14N'ed element, because it is already exc-C14N'ed. A stable exc- C14N'ed element is a completely legal well-formed XML element so non-MProf-aware endpoint should have no problem processing it.

2.2.4 Message Canonicalization Conditions

The sender MUST canonicalize the SOAP message, including mustUnderstand, role and attribute formats and processing instruction and whitespace.

Specifically, the normative processing rules specified in SOAP Message Canonicalization must be implemented. That specification is incorporated by reference; the following extract is included here for convenience:

The following processing rules are required, as stated in SOAP Message Canonicalization:

- For child element information items of the SOAP Header element information item:
  - If the SOAP mustUnderstand attribute information item is present with a value of "0" or "false" then remove the mustUnderstand attribute information item.
  - If the SOAP role attribute information item is present with a value of "http://www.w3.org/2002/06/soap-envelope/role/ ultimateReceiver" or "" then remove the role attribute information item.
  - If the SOAP relay attribute information item is present with a value of "0" or "false" then remove the relay attribute information item.
- Processing instruction information items that are children of the SOAP Envelope , Header , Fault , Code , Subcode , Value , Reason , Text , Node and Role element information items are removed.
- Whitespace character information items that are children of the SOAP Envelope , Header , Fault , Code , Subcode , Value , Reason , Node and Role element information items are removed.

The handling of the processing for “true” or “1” values is handled differently here than in the SOAP Messaging Canonicalization Note, in order to be consistent with the WS-I Basic Profile (R1013):

- For child element information items of the SOAP Header element information item:
  - If the SOAP mustUnderstand attribute information item is present with a value of "1" or "true" then change its value to “1”.
- If the SOAP relay attribute information item is present with a value of "true" then change its value to "1".

2.2.5 Security Token Order Conditions

In MProf messages, if a security token is referenced in a signature or encryption element (<ds:Signature>, <xenc:EncryptedKey>, <xenc:ReferenceList>, or <xenc:EncryptedData>) and if the security token is carried in the same envelope, the security token MAY appear before it is referenced. In addition, this intra-envelope reference MUST be done through a wsu:Id attribute and <wsse:Reference URI="…"/> element. This condition allows implementations to have necessary key materials ready at the point they are needed. The receiving node MAY stop processing the message if the referenced token is not yet available when it is used (i.e., the token appears after it is used) and MAY return a SOAP Fault.

2.2.6 XML Digital Signature Constraints

The following constraints are applied to XML Digital Signatures when MProf messaging is used. Note that this only applies to signatures conveyed in the SOAP header. Note also that SignatureProperty elements *are* allowed.

2.2.7 KeyInfo Handling

KeyInfo as defined in the XML Signature recommendation offers a wide variety of extensible means for conveying key information, including the ability to specify a URI for retrieving key information, a variety of X.509 certificate representations, PGP, SPKI and other possibilities. In addition, WSS: SOAP Message Security defines a number of security tokens that may be referenced using a SecurityTokenReference.

The WSS: SOAP Message Security specification recommends that WSS: SOAP Message Security tokens be used but allows other KeyInfo usages for signatures.

The MProf REQUIRES use of Web Services Security : SOAP Message Security tokens and disallows arbitrary KeyInfo usages.

2.2.8 Algorithm Constraints

Use of the XML Signature HMAC algorithm is an option to consider both to reduce receiver processing requirements as well as to enable streaming data, but is not a requirement.

2.2.9 Manifests

XML Signature Manifests MUST NOT be used in MProf compliant signatures. Manifests are a mechanism to allow applications to avoid reference digest validation when it makes sense in the application context. In the SOAP messaging context this is not desired or required.

2.2.10 Processing Rules

The sender MUST perform XML and SOAP Message canonicalization before any signing operations are performed.

---

3 Example

The following example shows an MProf signed and encrypted message. Note that in order to improve readability, this example is not exactly exc-C14N'ed. In particular, white spaces within a tag and empty tags should be read as if they were exc-C14N'ed.

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/"
  S:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<S:Header>
<wsse:MProf Stable="wsse:StableRestricted" S:mustUnderstand="1" >
</wsse:MProf>
<wsse:Security
```

```
 xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/07/secext">
 <EncryptedKey xmlns="http://www.w3.org/2001/04/xmlenc#">
  <EncryptionMethod
   Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
<KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
 <KeyName>CN=Interop Server</KeyName>
</KeyInfo>
 <CipherData><CipherValue>LeUKTi9…9KQ=</CipherValue></CipherData>
 <ReferenceList><DataReference URI="#i1"/></ReferenceList>
</EncryptedKey>
<wsse:BinarySecurityToken EncodingType="wsse:Base64Binary"
  wsu:Id="i3" ValueType="wsse:X509v3">MIIDnTC…1xeEq6X=
</wsse:BinarySecurityToken>
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
 <SignedInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
  <CanonicalizationMethod
    Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
  <SignatureMethod
    Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
  <Reference URI="#i4">
   <Transforms>
    <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
   </Transforms>
   <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
   <DigestValue>R9qBlYS6u8FWJprZ5iSCfbOISVA=</DigestValue>
  </Reference>
 </SignedInfo>
 <ds:SignatureValue>BuD+PDxEc9liWyiQ... kSRM=</ds:SignatureValue>
 <ds:KeyInfo>
  <wsse:SecurityTokenReference>
   <wsse:Reference URI="#i3"/>
  </wsse:SecurityTokenReference>
</ds:KeyInfo>
```

```
 </ds:Signature>
</wsse:Security>
</S:Header>
<S1:Body xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/07/utility"
   xmlns:S1="http://schemas.xmlsoap.org/soap/envelope/"
   wsu:Id="i4">
 <EncryptedData xmlns="http://www.w3.org/2001/04/xmlenc#"
    Id="i1" Type="http://www.w3.org/2001/04/xmlenc#Content">
  <EncryptionMethod
   Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
  <CipherData><CipherValue>DXnqS…PyFHzo7Pw==</CipherValue></CipherData>
 </EncryptedData>
</S1:Body>
</S:Envelope>
```

Note that the namespace prefix S1 is declared in the <S1:Body> element although the same namespace URI is already bound to another prefix, S. This is required for the <S1:Body> element to be a stable exc-C14N element.

4 Security Considerations

This specification introduces no addtioanl security concideractions over and above the ones defined in WSS: SOAP Message Security specification.

5 Acknowledgements

We would like to thank the following people for their contributions towards this specification:

Garry Ellison, Sun

Ron Monzillo, Sun

6 References

[DIME]

Henrik Frystyk Nielsen, Erik Christensen ,Joel Farrell , "WS-Attachments", June 17, 2002, DRAFT **http://www.ietf.org/internet-drafts/draft-nielsen-dime-soap-01.txt**

[KEYWORDS]

S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," RFC 2119, Harvard University, March 1997

[RFC2068]

IETF Standard, "Hypertext Transfer Protocol -- HTTP/1.1" January 1997

[SOAP]

W3C Note, "SOAP: Simple Object Access Protocol 1.1," 08 May 2000.

W3C Note, "SOAP Messages with Attachments", 11 December 2000 "http://www.w3.org/TR/2000/NOTE-SOAP-attachments-20001211""

[SOAPCanon]

W3C Note, SOAP Message Canonicalization (SM-C14N), 15 January 2003

[URI]

T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," RFC 2396, MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998.

[XML Signature]

W3C Recommendation, "XML Signature Syntax and Processing," 12 February 2002.

[Exclusive Canonicalization]

W3C Recommendation, "Exclusive XML Canonicalization Version 1.0," 18 July 2002.

[XML-Encrypt]

W3C Recommendation, "XML Encryption Syntax and Processing," 10 December 2002

W3C Recommendation, "Decryption Transform for XML Signature", 10 December 2002.

[Web Services Security : SOAP Message Security]

Web Services Security: SOAP Message Security

Web Services Security: Username Token Profile
Web Services Security: Kerberos Token Profile
Web Services Security: SAML Token Profile
Web Services Security X509 Token Profile
Web Services Security: XrML Token Profile

See Oasis Web Services Security page: http://www.oasis-open.org/committees/wss/

[Web Services Interoperability : Basic Profile]

487 Basic Profile Version 1.0, 12 February 2003

488

Appendix A. Well-Known MProf Transforms

(TBD)

Appendix B. Revision History

Rev	Date	What
01	07-Feb-03	Initial draft.
04	03-Mar-04	Final Draft.

Appendix C. Notices

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to implement this specification. Please address the information to the OASIS Executive Director.