SAML Token Profile Version 1.0

Working Group Draft

2006-01-20

This version:
http://www.ws-i.org/Profiles/SAMLTokenProfile-1.0-2006-01-20.html

Latest version:
http://www.ws-i.org/Profiles/SAMLTokenProfile-1.0.html

Editors:
Abbie Barbir, Nortel Networks
Martin Gudgin, Microsoft
Michael McIntosh, IBM
K. Scott Morrison, Layer 7

Administrative contact:
secretary@ws-i.org

Abstract

This document defines the WS-I SAML Token Profile 1.0, based on a non-proprietary Web services specification, along with clarifications and amendments to that specification which promote interoperability.

Status of this Document

This document is a Working Group Draft; it has been accepted by the Working Group as reflecting the current state of discussions. It is a work in progress, and should not be considered authoritative or final; other documents may supersede this document.

Notice

Feedback

The Web Services-Interoperability Organization (WS-I) would like to receive input, suggestions and other feedback ("Feedback") on this work from a wide variety of industry participants to improve its quality over time.

By sending email, or otherwise communicating with WS-I, you (on behalf of yourself if you are an individual, and your company if you are providing Feedback on behalf of the company) will be deemed to have granted to WS-I, the members of WS-I, and other parties that have access to your Feedback, a non-exclusive, non-transferable, worldwide, perpetual, irrevocable, royalty-free license to use, disclose, copy, license, modify, sublicense or otherwise distribute and exploit in any manner whatsoever the Feedback you provide regarding the work. You acknowledge that you have no expectation of confidentiality with respect to any Feedback you provide. You represent and warrant that you have rights to provide this Feedback, and if you are providing Feedback on behalf of a company, you represent and warrant that you have the rights to provide Feedback on behalf of your company. You also acknowledge that WS-I is not required to review, discuss, use, consider or in any way incorporate your Feedback into future versions of its work. If WS-I does incorporate some or all of your Feedback in a future version of the work, it may, but is not obligated to include your name (or, if you are identified as acting on behalf of your company, the name of your company) on a list of contributors to the work. If the foregoing is not acceptable to you and any company on whose behalf you are acting, please do not provide any Feedback.

Feedback on this document should be directed to wsi_secprofile_comment@lists.ws-i.org.

Table of Contents

1. Introduction

This document defines the WS-I SAML Token Profile 1.0 (hereafter, "Profile"), consisting of a set of non-proprietary Web services specifications, along with clarifications to and amplifications of those specifications which promote interoperability.

Section 1, *"Introduction,"* introduces the Profile and describes its relationship to other, existing profiles.

Section 2, *"Document Conventions,"* describes notational conventions utilized by this Profile.

Section 3, *"Profile Conformance,"* explains what it means to be conformant to the Profile.

Each subsequent section addresses a component of the Profile, and consists of two parts: an overview detailing the component specifications and their extensibility points, followed by subsections that address individual parts of the component specifications. Note that there is no relationship between the section numbers in this document and those in the referenced specifications.

1.1 Relationship to other Profiles

This Profile adds an additional security token type for use with Basic Security Profile 1.0.

2. Document Conventions

This document follows conventions common to all WS-I profiles. These are described in the following sections.

2.1 Security Considerations

The Profile will draw attention to security considerations; however, these are informational only and should be treated as non-normative. Adherence to these considerations does not guarantee security.

Security considerations are presented as follows:

Cnnnn *Statement text here.*

where "nnnn" is replaced by a number that is unique among the considerations in the Profile, thereby forming a unique consideration identifier.

2.2 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 RFC2119.

Normative statements of requirements in the Profile (i.e., those impacting conformance, as outlined in "Conformance Requirements") are presented in the following manner:

Rnnnn *Statement text here.*

where "nnnn" is replaced by a number that is unique among the requirements in the Profile , thereby forming a unique requirement identifier.

Requirement identifiers can be considered to be namespace qualified, in such a way as to be compatible with QNames from Namespaces in XML. If there is no explicit namespace prefix on a requirement's identifier (e.g., "R9999" as opposed to "bp10:R9999"), it should be interpreted as being in the namespace identified by the conformance URI of the document section it occurs in. If it is qualified, the prefix should be interpreted according to the namespace mappings in effect, as documented below.

Some requirements clarify the referenced specification(s), but do not place additional constraints upon implementations. For convenience, clarifications are annotated in the following manner: C

Some requirements are derived from ongoing standardization work on the referenced specification(s). For convenience, such forward-derived statements are annotated in the following manner: xxxx, where "xxxx" is an identifier for the specification (e.g., "WSDL20" for WSDL Version 2.0). Note that because such work was not complete when this document was published, the specification that the requirement is derived from may change; this information is included only as a convenience to implementers.

This specification uses a number of namespace prefixes throughout; their associated URIs are listed below. Note that the choice of any namespace prefix is arbitrary and not semantically significant.

- **soap** - "http://schemas.xmlsoap.org/soap/envelope/"
- **wsi** - "http://www.ws-i.org/schemas/conformanceClaim"
- **ds** - "http://www.w3.org/2000/09/xmldsig#""
- **xenc** - "http://www.w3.org/2001/04/xmlenc#"
- **c14n** - "http://www.w3.org/2001/10/xml-exc-c14n#"
- **wsse** - "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
- **wsu** - "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"

- **b10** - "http://www.ws-i.org/Profiles/Basic/2003-08/BasicProfile-1.0a.htm"
- **bp11** - "http://members.ws-i.org/dman/Document.phx/Private+Folders/Community+Folder/Working+Groups/WSBasic+Profile/Profile/v1.1/BasicProfile-1.1-WGAD.html?versionID=1""
- **saml** - "oasis:names:tc:SAML:1.0:assertion"
- **samlp** - "oasis:names:tc:SAML:1.0:protocol"

2.3 Profile Identification and Versioning

This document is identified by a name (in this case, SAML Token Profile) and a version number (here, 1.0). Together, they identify a particular *profile instance*.

Version numbers are composed of a major and minor portion, in the form "major.minor". They can be used to determine the precedence of a profile instance; a higher version number (considering both the major and minor components) indicates that an instance is more recent, and therefore supersedes earlier instances.

Instances of profiles with the same name (e.g., "Example Profile 1.1" and "Example Profile 5.0") address interoperability problems in the same general scope (although some developments may require the exact scope of a profile to change between instances).

One can also use this information to determine whether two instances of a profile are backwards-compatible; that is, whether one can assume that conformance to an earlier profile instance implies conformance to a later one. Profile instances with the same name and major version number (e.g., "Example Profile 1.0" and "Example Profile 1.1") MAY be considered compatible. Note that this does not imply anything about compatibility in the other direction; that is, one cannot assume that conformance with a later profile instance implies conformance to an earlier one.

3 Profile Conformance

Conformance to the Profile is defined by adherence to the set of *requirements* defined for a specific *target*, within the *scope* of the Profile. This section explains these terms and describes how conformance is defined and used.

3.1 Conformance Requirements

Requirements state the criteria for conformance to the Profile. They typically refer to an existing specification and embody refinements, amplifications, interpretations and clarifications to it in order to improve interoperability. All requirements in the Profile are considered normative, and those in the specifications it references that are in-scope (see "Conformance Scope") should likewise be considered normative. When requirements in the Profile and its referenced specifications contradict each other, the Profile 's requirements take precedence for purposes of Profile conformance.

Requirement levels, using RFC2119 language (e.g., MUST, MAY, SHOULD) indicate the nature of the requirement and its impact on conformance. Each requirement is individually identified (e.g., R9999) for convenience.

For example;

> R9999 *WIDGETs SHOULD be round in shape.*

This requirement is identified by "R9999", applies to the target WIDGET (see below), and places a conditional requirement upon widgets; i.e., although this requirement must be met to maintain conformance in most cases, there are some situations where there may be valid reasons for it not being met (which are explained in the requirement itself, or in its accompanying text).

Each requirement statement contains exactly one requirement level keyword (e.g., "MUST") and one conformance target keyword (e.g., "MESSAGE"). The conformance target keyword appears in bold text (e.g. "**MESSAGE**"). Other conformance targets appearing in non-bold text are being used strictly for their definition and NOT as a conformance target. Additional text may be included to illuminate a requirement or group of requirements (e.g., rationale and examples); however, prose surrounding requirement statements must not be considered in determining conformance.

Definitions of terms in the Profile are considered authoritative for the purposes of determining conformance.

None of the requirements in the Profile , regardless of their conformance level, should be interpreted as limiting the ability of an otherwise conforming implementation to apply security countermeasures in response to a real or perceived threat (e.g., a denial of service attack).

3.2 Conformance Targets

Conformance targets identify what artifacts (e.g., SOAP message, WSDL description, UDDI registry data) or parties (e.g., SOAP processor, end user) requirements apply to.

This allows for the definition of conformance in different contexts, to assure unambiguous interpretation of the applicability of requirements, and to allow conformance testing of artifacts (e.g., SOAP messages and WSDL descriptions) and the behavior of various parties to a Web service (e.g., clients and service instances).

Requirements' conformance targets are physical artifacts wherever possible, to simplify testing and avoid ambiguity.

The following conformance targets are used in the Profile :

- **SECURE_ENVELOPE** - a SOAP envelope that contains sub-elements that have been subject to integrity and/or confidentiality protection. A message is considered conformant when all of its contained Artifacts are conformant with all Statements Targeted to those Artifacts as appropriate in the Basic Security Profile. Use of artifacts for which there are no statements in the Basic Security Profile does not affect conformance.
- **SECURE_MESSAGE** - protocol elements that have WSS Security applied to them. Protocol elements include a primary SOAP envelope and optionally associated SwA attachments.
- **SENDER** - software that generates a message according to the protocol(s) associated with it. A sender is considered conformant when all of the messages it produces are conformant and its behavior is conformant with all statements related to SENDER in the Basic Security Profile 1.0. (from Basic Profile 1.0)
- **RECEIVER** - software that consumes a message according to the protocol(s) associated with it. A receiver is considered conformant when it is capable of consuming conformant messages containing the artifacts that it supports and its behavior is conformant with all statements related to RECEIVER in the Basic Security Profile 1.0. A conformant receiver need not accept all possible conformant messages. A conformant receiver may choose not to support artifacts that provide unneeded or undesired functionality. When a receiver supports a specific artifact, and the Basic Security Profile 1.0 contains statements related to that artifact, a conformant receiver must accept all required conformant forms of that artifact. (from Basic Profile 1.0)
- **INSTANCE** - software that implements a wsdl:port or a uddi:bindingTemplate. (from Basic Profile 1.0)
- **SOAP_ENVELOPE** - an element named soap:Envelope, which has no parent element.
- **SOAP_HEADER** - an element named soap:Header, included as a child of the SOAP_ENVELOPE.
- **HEADER_ELEMENT** - an element included as a child of the SOAP_HEADER.
- **SECURITY_HEADER** - a HEADER_ELEMENT named wsse:Security.

- **SIGNATURE** - an element named ds:Signature, included as a child of a SECURITY_HEADER.
- **SIG_KEY_INFO** - an element named ds:KeyInfo, included as a child of a SIGNATURE.
- **SIGNED_INFO** - an element named ds:SignedInfo, included as a child of a SIGNATURE.
- **SIGNATURE_METHOD** - an element named ds:SignatureMethod, included as a child of a SIGNED_INFO.
- **SIG_REFERENCE** - an element named ds:Reference, included as a child of a SIGNED_INFO.
- **SIG_TRANSFORMS** - an element named ds:Transforms, included as a child of a SIG_REFERENCE.
- **SIG_TRANSFORM** - an element named ds:Transform, included as a child of a SIG_TRANSFORMS.
- **CANONICALIZATION_METHOD** - an element named ds:CanonicalizationMethod, included as a child of a SIGNED_INFO or a wsse:TransformationParameters child of a SIG_TRANSFORM.
- **INCLUSIVE_NAMESPACES** - an element named xc14n:InclusiveNamespaces, include as a child of a SIG_TRANSFORM or a CANONICALIZATION_METHOD.
- **DIGEST_METHOD** - an element named ds:DigestMethod, included as a child of a SIG_TRANSFORM.
- **ENCRYPTED_KEY** - an element named xenc:EncryptedKey, included as a child of a SECURITY_HEADER.
- **ENC_KEY_INFO** - an element named ds:KeyInfo, included as a child of an ENCRYPTED_KEY or ENCRYPTED_DATA.
- **ENC_REFERENCE_LIST** - an element named xenc:ReferenceList, included as a child of a SECURITY_HEADER.
- **EK_REFERENCE_LIST** - an element named xenc:ReferenceList, included as a child of an ENCRYPTED_KEY.
- **ENC_KEY_REFERENCE** - an element named xenc:KeyReference, included as a child of an ENC_REFERENCE_LIST.
- **EK_KEY_REFERENCE** - an element named xenc:KeyReference, included as a child of an EK_REFERENCE_LIST.
- **ENC_DATA_REFERENCE** - an element named xenc:DataReference, included as a child of an ENC_REFERENCE_LIST.
- **EK_DATA_REFERENCE** - an element named xenc:DataReference, included as a child of an EK_REFERENCE_LIST.
- **ENCRYPTED_DATA** - an element named xenc:EncryptedData, referenced by an EK_REFERENCE_LIST or an ENC_REFERENCE_LIST.
- **ED_ENCRYPTION_METHOD** - an element named xenc:EncryptionMethod, included as a child of an ENCRYPTED_DATA.
- **EK_ENCRYPTION_METHOD** - an element named xenc:EncryptionMethod, included as a child of an ENCRYPTED_KEY.
- **SECURITY_TOKEN_REFERENCE** - an element named wsse:SecurityTokenReference, included as a descendant of a SECURITY_HEADER or ENCRYPTED_DATA.
- **STR_EMBEDDED** - an element named wsse:Embedded, included as a child of a SECURITY_TOKEN_REFERENCE.
- **STR_REFERENCE** - an element named wsse:Reference, included as a child of a SECURITY_TOKEN_REFERENCE.
- **STR_KEY_NAME** - an element named ds:KeyName, included as a child of a SECURITY_TOKEN_REFERENCE.
- **STR_KEY_IDENTIFIER** - an element named wsse:KeyIdentifier, included as a child of a SECURITY_TOKEN_REFERENCE.
- **STR_ISSUER_SERIAL** - an element named ds:X509IssuerSerial, included as a child of a child element named ds:X509Data of a SECURITY_TOKEN_REFERENCE.
- **INTERNAL_SECURITY_TOKEN** - a SECURITY_TOKEN defined in a security token profile, included as a child of a SECURITY_HEADER or STR_EMBEDDED.
- **EXTERNAL_SECURITY_TOKEN** - a SECURITY_TOKEN defined in a security token profile, not included as a descendant of a SOAP_ENVELOPE.
- **EXTERNAL_TOKEN_REFERENCE** - a SECURITY_TOKEN_REFERENCE that refers to an EXTERNAL_SECURITY_TOKEN.
- **USERNAME_TOKEN** - a SECURITY_TOKEN named wsse:UsernameToken.
- **NONCE** - an element named wsse:Nonce, included as a child of a USERNAME_TOKEN.
- **PASSWORD** - an element named wsse:Password, included as a child of a USERNAME_TOKEN.
- **BINARY_SECURITY_TOKEN** - a SECURITY_TOKEN named wsse:BinarySecurityToken.
- **X509_TOKEN** - a BINARY_SECURITY_TOKEN containing an X.509 certificate.
- **PKCS7_TOKEN** - a BINARY_SECURITY_TOKEN containing a PKCS#7 certificate chain.
- **PKIPATH_TOKEN** - a BINARY_SECURITY_TOKEN containing a PkiPath certificate chain.
- **KERBEROS_TOKEN** - a BINARY_SECURITY_TOKEN containing a GSS wrapped Kerberos v5 AP-REQ or a non-wrapped Kerberos v5 AP-REQ.
- **REL_TOKEN** - a SECURITY_TOKEN named rel:license.
- **SAML_TOKEN** - a SECURITY_TOKEN named saml:Assertion.
- **SECURITY_TOKEN** - an INTERNAL_SECURITY_TOKEN or EXTERNAL_SECURITY_TOKEN (e.g. USERNAME_TOKEN, X509_TOKEN, REL_TOKEN, SAML_TOKEN, KERBEROS_TOKEN, etc.).
- **TIMESTAMP** - an element named wsu:Timestamp, included as a child of a SECURITY_HEADER.
- **CREATED** - an element named wsu:Created, included as a child of a TIMESTAMP or USERNAME_TOKEN.
- **EXPIRES** - an element named wsu:Expires, included as a child of a TIMESTAMP or USERNAME_TOKEN.
- **MIME_HEADER** - a header field of a multipart entity, as defined by MIME.
- **MIME_BODY** - the body of a multipart entity, as defined by MIME.
- **MIME_PART** - the MIME_BODY and all MIME_HEADERs associated with a single multipart entity, as defined by MIME.
- **INTERNAL_SAML_TOKEN** - an INTERNAL_SECURITY_TOKEN that is a SAML_TOKEN.
- **EXTERNAL_SAML_TOKEN** - an EXTERNAL_SECURITY_TOKEN that is a SAML_TOKEN.
- **SAML_SUBJECT_CONFIRMATION** - an element named saml:SubjectConfirmation, included in a SAML_TOKEN
- **SAML_SC_KEY_INFO** - an element named ds:KeyInfo, included as a child of a SAML_SUBJECT_CONFIRMATION
- **SAML_AUTHORITY_BINDING** - an element named saml:AuthorityBinding, included as a child of an STR_KEY_IDENTIFIER

3.3 Conformance Scope

The scope of the Profile delineates the technologies that it addresses; in other words, the Profile only attempts to improve interoperability within its own scope. Generally, the Profile 's scope is bounded by the specifications referenced by it.

The Profile's scope is further refined by extensibility points. Referenced specifications often provide extension mechanisms and unspecified or open-ended configuration parameters; when identified in the Profile as an extensibility point, such a mechanism or parameter is outside the scope of the Profile , and its use or non-use is not relevant to conformance.

Note that the Profile may still place requirements on the use of an extensibility point. Also, specific uses of extensibility points may be further restricted by other profiles, to improve interoperability when used in conjunction with the Profile .

Because the use of extensibility points may impair interoperability, their use should be negotiated or documented in some fashion by the parties to a Web service; for example, this could take the form of an out-of-band agreement.

The Profile's scope is defined by the referenced specifications in Appendix A, as refined by the extensibility points in Appendix B.

3.4 Claiming Conformance

Claims of conformance to the Profile can be made using the following mechanisms, as described in Conformance Claim Attachment Mechanisms, when the applicable Profile requirements associated with the listed targets have been met:

The CCAM URI may change before final publication.

The conformance claim URI for this Profile is "http://ws-i.org/profiles/basic-security/saml-token/1.0" .

4. SAML Token Profile

This section of the Profile incorporates the following specifications by reference:

- Web Services Security: SAML Token Profile

4.1 Requirements from SAML Token Profile

The following specifications (or sections thereof) are referred to in this section of the Profile :

- Web Services Security: SAML Token Profile

Web Services Security: SAML Token Profile contains various statements containing the RFC2119 'MUST' keyword. This Profile restates some of those statements:

4.1.1 References to SAML assertions from SAML assertions prohibited

This requirement rules out the possibility of a SAML assertion refering to itself, an undesirable occurence as it essentially makes the assertion self certifying. In addition a reference to another SAML assertion is also ruled out, this is undesirable as SAML does not have a transative trust model.

R6601 *Any* **SAML_SC_KEY_INFO** *MUST NOT contain a reference to a SAML_TOKEN.* C

For example,

INCORRECT:

```
<!-- This example is incorrect because the ds:KeyInfo in the SAML assertion contains a reference to another such assertion thus conflicting wi
<wsse:Security xmlns:wsse='http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd'
               xmlns:wsu='http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd'
               xmlns:xenc='http://www.w3.org/2001/04/xmlenc#'
               xmlns:ds='http://www.w3.org/2000/09/xmldsig#' >
  <saml:Assertion xmlns:saml='urn:oasis:names:tc:SAML:1.0:assertion'
                  MajorVersion='1' MinorVersion='1'
                  AssertionID='uuid:006ab385-35e0-41b1-b0f5-ccef5090c1b0'
                  Issuer='http://example.org/issuer' IssueInstant='2004-11-04T21:01:50Z' >
    . . .
    <saml:AuthenticationStatement AuthenticationMethod='urn:oasis:names:tc:SAML:1.0:am:password' AuthenticationInstant='2004-11-04T21:01:50Z' 
      <saml:Subject>
        . . .
        <saml:SubjectConfirmation>
          . . .
          <ds:KeyInfo xmlns:ds='http://www.w3.org/2000/09/xmldsig#' >
            <wsse:SecurityTokenReference>
              <wsse:Reference URI='uuid:a9afffbe-a0fb-4789-8b54-299782c3c0ac'
                              ValueType='http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0#SAMLAssertionID' />
            </wsse:SecurityTokenReference>
          </ds:KeyInfo>
        </saml:SubjectConfirmation>
      </saml:Subject>
    </saml:AuthenticationStatement>
        . . .
  </saml:Assertion>
</wsse:Security>
```

4.1.2 References to SAML assertions by KeyIdentifier

These requirements restate various statements from the base specification related to references to SAML assertions that use wsse:KeyIdentifiers.

R6602 *Any* **STR_KEY_IDENTIFIER** *that references a SAML_TOKEN MUST include a ValueType attribute.* C

R6603 *Any* **STR_KEY_IDENTIFIER** *ValueType attribute that references SAML_TOKEN MUST have a value of "http://docs.oasis-open.org /wss/oasis-wss-saml-token-profile-1.0#SAMLAssertionID"* C

R6604 *Any* **STR_KEY_IDENTIFIER** *that references a SAML_TOKEN MUST NOT include an EncodingType attribute.* C

R6605 *Any* **STR_KEY_IDENTIFIER** *that references a SAML_TOKEN MUST have a value encoded as an xs:string.* C

For example,

CORRECT:

```
<wsse:SecurityTokenReference xmlns:wsse='http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd'>
  <wsse:KeyIdentifier ValueType='http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0#SAMLAssertionID' >
  uuid:006ab385-35e0-41b1-b0f5-ccef5090c1b0
  </wsse:KeyIdentifier>
</wsse:SecurityTokenReference>
```

INCORRECT:

```
<!-- This example is incorrect because the wsse:KeyIdentifier element is missing a ValueType attribute thus conflicting with R6602 -->
<wsse:SecurityTokenReference xmlns:wsse='http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd'>
  <wsse:KeyIdentifier>uuid:006ab385-35e0-41b1-b0f5-ccef5090c1b0</wsse:KeyIdentifier>
</wsse:SecurityTokenReference>
```

INCORRECT:

```
<!-- This example is incorrect because the wsse:KeyIdentifier element has an incorrect value for the ValueType attribute thus conflicting with
<wsse:SecurityTokenReference xmlns:wsse='http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd'>
  <wsse:KeyIdentifier ValueType='http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0#SAML'>
  uuid:006ab385-35e0-41b1-b0f5-ccef5090c1b0
  </wsse:KeyIdentifier>
</wsse:SecurityTokenReference>
```

INCORRECT:

```
<!-- This example is incorrect because the wsse:KeyIdentifier has an EncodingType attribute thus conflicting with R6604 -->
<wsse:SecurityTokenReference xmlns:wsse='http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd'>
  <wsse:KeyIdentifier ValueType='http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0#SAMLAssertionID'
```

```
                      EncodingType='xs:string' >
  uuid:006ab385-35e0-41b1-b0f5-ccef5090c1b0
  </wsse:KeyIdentifier>
</wsse:SecurityTokenReference>
```

4.1.3 References to External SAML Assertions

These requirements restate various statements from the base specification related to references to SAML assertions that are outside a SECURE_ENVELOPE.

R6606 *Any* **STR_KEY_IDENTIFIER** *that references an EXTERNAL_SAML_TOKEN MUST contain a SAML_AUTHORITY_BINDING.* C

R6607 *Any AuthorityKind attribute of a* **SAML_AUTHORITY_BINDING** *MUST have a value of samlp:AssertionIdReference.* C

R6608 *Any* **STR_KEY_IDENTIFIER** *that references an INTERNAL_SAML_TOKEN MUST NOT contain a SAML_AUTHORITY_BINDING.* C

Appendix A: Referenced Specifications

The following specifications' requirements are incorporated into the Profile by reference, except where superseded by the Profile:

- Web Services Security: SAML Token Profile

Appendix B: Extensibility Points

This section identifies extensibility points, as defined in "Scope of the Profile ," for the Profile 's component specifications.

These mechanisms are out of the scope of the Profile ; their use may affect interoperability, and may require private agreement between the parties to a Web service.

Appendix C: Acknowledgements

This document is the work of the WS-I Basic Security Profiles Working Group, whose members have included:

Jan Alexander (Microsoft Corporation), Steve Anderson (BMC), Paula Austel (IBM), Siddharth Bajaj (Verisign), Frank Balluffi (Deutsche Bank), Abbie Barbir (Nortel), David Baum (Kantega AS), Randy Bias (Grand Central Communications), Tim Bond (webMethods, Inc.), Heidi Buelow (Quovadx), David Burdett (Commerce One, Inc.), Ted Burghart (Hitachi, Ltd.) Symon Chang (TIBCO, Inc.), Richard Chennault, (Kaiser Permanente), Dipak Chopra (SAP AG), Jamie Clark (OASIS), Edward Cobb (BEA Systems, Inc.), David Cohen (Merrill Lynch), Brett Cooper, (Accenture), Ugo Corda (SeeBeyond Technology), Paul Cotton (Microsoft Corporation), Suresh Damodaran, (Rosettanet), Mark Davis (Intel), Alex Deacon (Verisign), Thomas DeMartini (ContentGuard, Inc.), Blake Dournaee (Intel), Rob Drew (Charlse Schwab), Gregory Elkins (Reed Elsevier), Mark Ericson (Mindreef), Jon Oyvind Eriksen (Kantega AS), Chris Ferris (IBM), Bob Freund, (Hitachi), Edwin Goei (Sun Microsystems), Grant Goodale (Reactivity, Inc.), Marc Goodner (SAP AG), Phil Goodwin (Sun Microsystems), Marc Graveline (Cognos, Inc.), Eric Gravengaard (Reactivity, Inc.), Thomas Gross (IBM), Martin Gudgin (Microsoft Corporation), Marc Hadley (Sun Microsystems), Mark Hapner (Sun Microsystems), Nathan Harris (Kaiser Permanente), Bret Hartman (IBM), Frederick Hirsch (Nokia), Jason Hogg (Microsoft Corporation), Maryann Hondo (IBM), Lawrence Hsiung (Quovadx), Tony Huber (Commerce Quest), Jim Hughes (Hewlett-Packard), Michael Hui (Computer Associates), Brian Jackson (Avanade, Inc.), Steve Jenisch (SAS Institute), Erik Johnson (Epicor), Chris Kaler (Microsoft Corporation), Anish Karmarkar (Oracle Corporation), Dana Kaufman, (Forum Systems), Manveen Kaur (Sun Microsystems), Slava Kavsan (RSA Security), Paul Knight (Nortel Networks), Chris Kurt (Microsoft Corporation), Kelvin Lawrence (IBM), Hal Lockhart (BEA Systems), Brad Lund (Intel Corporation), Jim Luth (OPC Foundation), Paul Madsen (Entrust, Inc.), Eve Maler (Sun Microsystems), Skip Marler (Parasoft), Axl Mattheus (Sun Microsystems), Michael McIntosh (IBM), Craig Milhiser, (Ascential), Chris Miller (Accenture), Prateek Mishra (Oracle Corporation) Dale Moberg (Cyclone Commerce), Ron Monzillo (Sun Microsystems), K. Scott Morrison, (Layer 7) Tim Moses (Entrust, Inc.), Tony Nadalin (IBM), Nataraj Nagaratnam (IBM), Andrew Nash (RSA Security), Hsin Ning (Bestning Technologies), Eisaku Nishiyama (Hitachi, Ltd.), Mark Nottingham (BEA Systems, Inc.), TJ Pannu (ContentGuard, Inc.), Martine Pean (Quovadx), Robert Philpott (RSA Security), Dave Prout (BT), Joe Pruitt (F5 Networks, Inc.), Eric Rejkovic (Oracle Corporation), Matt Recupito (Accenture), Jason Rouault (Hewlett-Packard), Rich Salz (IBM), Matt Sanchez (Webify Solutions, Inc.), Jerry Schwarz (Oracle Corporation), Senthil Sengodan (Nokia), Shawn Sharp (Cyclone Commerce), Aslak Siira (F5 Networks, Inc.), David Solo (Citigroup, Inc.), Davanum Srinivas (Computer Associates), Raghavan Srinivas (Sun Microsystems), John Stanton (Defense Information Systems Agency), Andrew Stone (Accenture), Julie Surer (MITRE), Wes Swenson (Forum Systems), Dino Vitale (Citigroup, Inc.), Jonathan Wenocur (IBM), Pete Wenzel (Sun Microsystems), Ian White (Micro Focus)