Web Services Security X.509 Certificate Token Profile 1.1

OASIS Committee Specification, 14 November 2005

OASIS Identifier:

`wss-v1.1-spec-cs-X509-token-profile`

Document Location:

`http://docs.oasis-open.org/wss/oasis-wss-x509-token-profile-1.1`

Technical Committee:

Web Service Security (WSS)

Chairs:

Kelvin Lawrence, IBM
Chris Kaler, Microsoft

Editors:

Anthony Nadalin, IBM
Chris Kaler, Microsoft
Ronald Monzillo, Sun
Phillip Hallam-Baker, Verisign

Abstract:

This document describes how to use X.509 Certificates with the Web Services Security: SOAP Message Security specification [WS-Security] specification.

Status:

Committee members should send comments on this specification to the 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 WS-Security TC web page (http://www.oasis-open.org/committees/wss/ipr.php).

Notices

This section is non-normative.

Table of Contents

1 Introduction (Non-Normative)

This specification describes the use of the X.509 authentication framework with the Web Services Security: SOAP Message Security specification [WS-Security].

An X.509 certificate specifies a binding between a public key and a set of attributes that includes (at least) a subject name, issuer name, serial number and validity interval. This binding may be subject to subsequent revocation advertised by mechanisms that include issuance of CRLs, OCSP tokens or mechanisms that are outside the X.509 framework, such as XKMS.

An X.509 certificate may be used to validate a public key that may be used to authenticate a SOAP message or to identify the public key with a SOAP message that has been encrypted.

Note that Sections 2.1, 2.2, all of 3, and indicated parts of 5 are normative. All other sections are non-normative.

2 Notations and Terminology (Normative)

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

2.1 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 a convention where 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/>).

2.2 Namespaces

Namespace URIs (of the general form "some-URI") represents some application-dependent or context-dependent URI as defined in RFC 3986 [URI]. This specification is designed to work with the general SOAP [SOAP11, SOAP12] message structure and message processing model, and should be applicable to any version of SOAP. The current SOAP 1.1 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.

The namespaces used in this document are shown in the following table (note that for brevity, the examples use the prefixes listed below but do not include the URIs – those listed below are assumed).

```
http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd
http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
1.0.xsd
http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd
```

The following namespace prefixes are used in this document:

Prefix	Namespace
S11	`http://schemas.xmlsoap.org/soap/envelope/`

S12	`http://www.w3.org/2003/05/soap-envelope`
ds	`http://www.w3.org/2000/09/xmldsig#`
xenc	`http://www.w3.org/2001/04/xmlenc#`
wsse	`http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd`
wsse11	`http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd`
wsu	`http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd`

Table 1- Namespace prefixes

URI fragments defined in this specification are relative to the following base URI unless otherwise stated:

```
http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-
profile-1.0
```

The following table lists the full URI for each URI fragment referred to in this specification.

URI Fragment	Full URI
#Base64Binary	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary
#STR-Transform	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#STR-Transform
#PKCS7	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#PKCS7
#X509v3	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3
#X509PKIPathv1	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509PKIPathv1
#X509SubjectKeyIdentifier	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509SubjectKeyIdentifier

2.3 Terminology

This specification adopts the terminology defined in Web Services Security: SOAP Message Security specification [WS-Security].

Readers are presumed to be familiar with the definitions of terms in the Internet Security Glossary [Glossary].

3 Usage (Normative)

This specification describes the syntax and processing rules for the use of the X.509 authentication framework with the Web Services Security: SOAP Message Security specification [WS-Security]. For the purposes of determining the order of preference of reference types, the use of IssuerSerial within X509Data should be considered to be a form of Key Identifier

3.1 Token types

This profile defines the syntax of, and processing rules for, three types of binary security token using the URI values specified in Table 2.

If the `ValueType` attribute is missing, the receiver may interpret it either based on a prior agreement or by parsing the content.

Token	ValueType URI	Description
Single certificate	#X509v3	An X.509 v3 certificate capable of signature-verification at a minimum
Single certificate	#x509v1	An X.509 v1 certificate capable of signature-verification at a minimum.
Certificate Path	#X509PKIPathv1	An ordered list of X.509 certificates packaged in a PKIPath
Set of certificates and CRLs	#PKCS7	A list of X.509 certificates and (optionally) CRLs packaged in a PKCS#7 wrapper

Table 2 – Token types

3.1.1 X509v3 Token Type

The type of the end-entity that is authenticated by a certificate used in this manner is a matter of policy that is outside the scope of this specification.

3.1.2 X509PKIPathv1 Token Type

The `X509PKIPathv1` token type MAY be used to represent a certificate path.

3.1.3 PKCS7 Token Type

The `PKCS7` token type MAY be used to represent a certificate path. It is RECOMMENDED that applications use the PKIPath object for this purpose instead.

The order of the certificates in a PKCS#7 data structure is not significant. If an ordered certificate path is converted to PKCS#7 encoded bytes and then converted back, the order of the certificates may not be preserved. Processors SHALL NOT assume any significance to the order of the certificates in the data structure. See [PKCS7] for more information.

3.2 Token References

In order to ensure a consistent processing model across all the token types supported by WSS: SOAP Message Security, the `<wsse:SecurityTokenReference>` element SHALL be used to specify all references to X.509 token types in signature or encryption elements that comply with this profile.

A `<wsse:SecurityTokenReference>` element MAY reference an X.509 token type by one of the following means:

- Reference to a Subject Key Identifier
  The `<wsse:SecurityTokenReference>` element contains a `<wsse:KeyIdentifier>` element that specifies the token data by means of a X.509 SubjectKeyIdentifier reference. A subject key identifier may only be used to reference an X.509v3 certificate."

- Reference to a Binary Security Token
  The `<wsse:SecurityTokenReference>` element contains a `wsse:Reference>` element that references a local `<wsse:BinarySecurityToken>` element or a remote data source that contains the token data itself.

- Reference to an Issuer and Serial Number
  The `<wsse:SecurityTokenReference>` element contains a `<ds:X509Data>` element that contains a `<ds:X509IssuerSerial>` element that uniquely identifies an end entity certificate by its X.509 Issuer and Serial Number.

3.2.1 Reference to an X.509 Subject Key Identifier

The `<wsse:KeyIdentifier>` element is used to specify a reference to an X.509v3 certificate by means of a reference to its X.509 SubjectKeyIdentifier attribute. This profile defines the syntax of, and processing rules for referencing a Subject Key Identifier using the URI values specified in Table 3 (note that URI fragments are relative to `http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0`).

Subject Key Identifier	ValueType URI	Description
Certificate Key Identifier	`#X509SubjectKeyIdentifier`	Value of the certificate's X.509 SubjectKeyIdentifier

Table 3 – Subject Key Identifier

The `<wsse:SecurityTokenReference>` element from which the reference is made contains the `<wsse:KeyIdentifier>` element. The `<wsse:KeyIdentifier>` element MUST have a `ValueType` attribute with the value `#X509SubjectKeyIdentifier` and its contents MUST be the value of the certificate's X.509v3 SubjectKeyIdentifier extension, encoded as per the `<wsse:KeyIdentifier>` element's `EncodingType` attribute. For the purposes of this specification, the value of the SubjectKeyIdentifier extension is the contents of the KeyIdentifier octet string, excluding the encoding of the octet string prefix.

3.2.2 Reference to a Security Token

The `<wsse:Reference>` element is used to reference an X.509 security token value by means of a URI reference.

The URI reference MAY be internal in which case the URI reference SHOULD be a bare name XPointer reference to a `<wsse:BinarySecurityToken>` element contained in a preceding message header that contains the binary X.509 security token data.

3.2.3 Reference to an Issuer and Serial Number

The `<ds:X509IssuerSerial>` element is used to specify a reference to an X.509 security token by means of the certificate issuer name and serial number.

The `<ds:X509IssuerSerial>` element is a direct child of the <ds:X509Data> element that is in turn a direct child of the `<wsse:SecurityTokenReference>` element in which the reference is made

3.3 Signature

Signed data MAY specify the certificate associated with the signature using any of the X.509 security token types and references defined in this specification.

An X.509 certificate specifies a binding between a public key and a set of attributes that includes (at least) a subject name, issuer name, serial number and validity interval. Other attributes may specify constraints on the use of the certificate or affect the recourse that may be open to a relying party that depends on the certificate. A given public key may be specified in more than one X.509 certificate; consequently a given public key may be bound to two or more distinct sets of attributes.

It is therefore necessary to ensure that a signature created under an X.509 certificate token uniquely and irrefutably specifies the certificate under which the signature was created.

Implementations SHOULD protect against a certificate substitution attack by including either the certificate itself or an immutable and unambiguous reference to the certificate within the scope of the signature according to the method used to reference the certificate as described in the following sections.

3.3.1 Key Identifier

The `<wsse:KeyIdentifier>` element does not guarantee an immutable and unambiguous reference to the certificate referenced. Consequently implementations that use this form of reference within a signature SHOULD employ the `STR` Dereferencing Transform within a reference to the signature key information in order to ensure that the referenced certificate is signed, and not just the ambiguous reference. The form of the reference is a bare name reference as defined by the XPointer specification [XPointer].

The following example shows a certificate referenced by means of a KeyIdentifier. The scope of the signature is the `<ds:SignedInfo>` element which includes both the message body (#body) and the signing certificate by means of a reference to the `<ds:KeyInfo>` element which references it (#keyinfo). Since the `<ds:KeyInfo>` element only contains a mutable reference to the certificate rather than the certificate itself, a transformation is specified which replaces the reference to the certificate with the certificate. The `<ds:KeyInfo>` element specifies the signing key by means of a `<wsse:SecurityTokenReference>` element which contains a `<wsse:KeyIdentifier>` element which specifies the X.509 subject key identifier of the signing certificate.

```
<S11:Envelope xmlns:S11="...">
   <S11:Header>
      <wsse:Security
           xmlns:wsse="..."
           xmlns:wsu="...">
         <ds:Signature
              xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <ds:SignedInfo>…
               <ds:Reference URI="#body">…</ds:Reference>
               <ds:Reference URI="#keyinfo">
                  <ds:Transforms>
                     <ds:Transform  Algorithm="...#STR-Transform">
                        <wsse:TransformationParameters>
                          <ds:CanonicalizationMethod Algorithm="…"/>
                        </wsse:TransformationParameters>
                     </ds:Transform>
                  </ds:Transforms>…
               </ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>HFLP…</ds:SignatureValue>
            <ds:KeyInfo Id="keyinfo">
               <wsse:SecurityTokenReference>
                  <wsse:KeyIdentifier EncodingType="...#Base64Binary"
                       ValueType="...#X509SubjectKeyIdentifier">
                     MIGfMa0GCSq…
                  </wsse:KeyIdentifier>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
         </ds:Signature>
      </wsse:Security>
   </S11:Header>
   <S11:Body wsu:Id="body"
        xmlns:wsu=".../">
      …
   </S11:Body>
</S11:Envelope>
```

3.3.2 Reference to a Binary Security Token

The signed data SHOULD contain a core bare name reference (as defined by the XPointer specification [XPointer]) to the`<wsse:BinarySecurityToken>` element that contains the security token referenced, or a core reference to the external data source containing the security token.

The following example shows a certificate embedded in a `<wsse:BinarySecurityToken>` element and referenced by URI within a signature. The certificate is included in the `<wsse:Security>` header as a `<wsse:BinarySecurityToken>` element with identifier `binarytoken`. The scope of the signature defined by a `<ds:Reference>` element within the `<ds:SignedInfo>` element includes the signing certificate which is referenced by means of the URI bare name pointer `#binarytoken`. The `<ds:KeyInfo>` element specifies the signing key by means of a `<wsse:SecurityTokenReference>` element which contains a `<wsse:Reference>` element which references the certificate by means of the URI bare name pointer `#binarytoken`.

```
<S11:Envelope xmlns:S11="...">
   <S11:Header>
      <wsse:Security
           xmlns:wsse="..."
           xmlns:wsu="...">
         <wsse:BinarySecurityToken
              wsu:Id="binarytoken"
              ValueType="…#X509v3"
              EncodingType="…#Base64Binary">
            MIIEZzCCA9CgAwIBAgIQEmtJZc0…
         </wsse:BinarySecurityToken>
         <ds:Signature
              xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <ds:SignedInfo>…
               <ds:Reference URI="#body">…</ds:Reference>
               <ds:Reference URI="#binarytoken">…</ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>HFLP…</ds:SignatureValue>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                  <wsse:Reference URI="#binarytoken" />
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
         </ds:Signature>
      </wsse:Security>
   </S11:Header>
   <S11:Body wsu:Id="body"
        xmlns:wsu="...">
      …
   </S11:Body>
</S11:Envelope>
```

3.3.3 Reference to an Issuer and Serial Number

The signed data SHOULD contain a core bare name reference (as defined by the XPointer specification [XPointer]) to the `<ds:KeyInfo>` element that contains the security token reference.

The following example shows a certificate referenced by means of its issuer name and serial number. In this example the certificate is not included in the message. The scope of the signature defined by the `<ds:SignedInfo>` element includes both the message body (#body) and the key information element (`#keyInfo`). The `<ds:KeyInfo>` element contains a `<wsse:SecurityTokenReference>` element which specifies the issuer and serial number of the specified certificate by means of the `<ds:X509IssuerSerial>` element.

```
<S11:Envelope xmlns:S11="...">
   <S11:Header>
      <wsse:Security
           xmlns:wsse="..."
           xmlns:wsu="...">
         <ds:Signature
                xmlns:ds="...">
            <ds:SignedInfo>…
               <ds:Reference URI="#body"></ds:Reference>
               <ds:Reference URI="#keyinfo"></ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>HFLP…</ds:SignatureValue>
            <ds:KeyInfo Id="keyinfo">
               <wsse:SecurityTokenReference>
                  <ds:X509Data>
                     <ds:X509IssuerSerial>
                        <ds:X509IssuerName>
                           DC=ACMECorp, DC=com
                        </ds:X509IssuerName>
                        <ds:X509SerialNumber>12345678</ds:X509SerialNumber>
                     </ds:X509IssuerSerial>
                  </ds:X509Data>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
         </ds:Signature>
      </wsse:Security>
   </S11:Header>
   <S11:Body wsu:Id="body"
        xmlns:wsu="...">
      …
   </S11:Body>
</S11:Envelope>
```

3.4 Encryption

Encrypted keys or data MAY identify a key required for decryption by identifying the corresponding key used for encryption by means of any of the X.509 security token types or references specified herein.

Since the sole purpose is to identify the decryption key it is not necessary to specify either a trust path or the specific contents of the certificate itself.

The following example shows a decryption key referenced by means of the issuer name and serial number of an associated certificate. In this example the certificate is not included in the message. The `<ds:KeyInfo>` element contains a `<wsse:SecurityTokenReference>` element which specifies the issuer and serial number of the specified certificate by means of the `<ds:X509IssuerSerial>` element.

```
<S11:Envelope
     xmlns:S11="..."
     xmlns:ds="..."
     xmlns:wsse="..."
     xmlns:xenc="...">
   <S11:Header>
      <wsse:Security>
         <xenc:EncryptedKey>
            <xenc:EncryptionMethod Algorithm="…"/>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                 <ds:X509Data>
                  <ds:X509IssuerSerial>
                     <ds:X509IssuerName>
                        DC=ACMECorp, DC=com
                     </ds:X509IssuerName>
                     <ds:X509SerialNumber>12345678</ds:X509SerialNumber>
                  </ds:X509IssuerSerial>
                 </ds:X509Data>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
            <xenc:CipherData>
               <xenc:CipherValue>…</xenc:CipherValue>
            </xenc:CipherData>
            <xenc:ReferenceList>
               <xenc:DataReference URI="#encrypted"/>
            </xenc:ReferenceList>
         </xenc:EncryptedKey>
      </wsse:Security>
   </S11:Header>
   <S11:Body>
      <xenc:EncryptedData Id="encrypted" Type="…">
         <xenc:CipherData>
            <xenc:CipherValue>…</xenc:CipherValue>
         </xenc:CipherData>
      </xenc:EncryptedData>
   </S11:Body>
</S11:Envelope>
```

The following example shows a decryption key referenced by means of the Thumbprint of an associated certificate. In this example the certificate is not included in the message. The `<ds:KeyInfo>` element contains a `<wsse:SecurityTokenReference>` element which specifies the Thumbprint of the specified certificate by means of the `http://docs.oasis-`

open.org/wss/oasis-wss-soap-message-security-1.1#ThumbprintSHA1 attribute of the <wsse:KeyIdentifier> element.

```
<S11:Envelope
     xmlns:S11="..."
     xmlns:ds="..."
     xmlns:wsse="..."
     xmlns:xenc="...">
   <S11:Header>
      <wsse:Security>
         <xenc:EncryptedKey>
            <xenc:EncryptionMethod Algorithm="…"/>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                    <wsse:KeyIdentifier
                      ValueType="http://docs.oasis-open.org/wss/oasis-wss-
soap-message-security-1.1#ThumbPrintSHA1" >LKiQ/CmFrJDJqCLFcjlhIsmZ/+0=
                    </wsse:KeyIdentifier>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
            <xenc:CipherData>
               <xenc:CipherValue>…</xenc:CipherValue>
            </xenc:CipherData>
            <xenc:ReferenceList>
               <xenc:DataReference URI="#encrypted"/>
            </xenc:ReferenceList>
         </xenc:EncryptedKey>
      </wsse:Security>
   </S11:Header>
   <S11:Body>
      <xenc:EncryptedData Id="encrypted" Type="…">
         <xenc:CipherData>
            <xenc:CipherValue>…</xenc:CipherValue>
         </xenc:CipherData>
      </xenc:EncryptedData>
   </S11:Body>
</S11:Envelope>
```

3.5 Error Codes

When using X.509 certificates, the error codes defined in the WSS: SOAP Message Security specification [WS-Security] MUST be used.

If an implementation requires the use of a custom error it is recommended that a sub-code be defined as an extension of one of the codes defined in the WSS: SOAP Message Security specification [WS-Security].

4 Threat Model and Countermeasures (Non-Normative)

The use of X.509 certificate token introduces no new threats beyond those identified in WSS: SOAP Message Security specification [WS-Security].

Message alteration and eavesdropping can be addressed by using the integrity and confidentiality mechanisms described in WSS: SOAP Message Security [WS-Security]. Replay attacks can be addressed by using message timestamps and caching, as well as other application-specific tracking mechanisms. For X.509 certificates, identity is authenticated by use of keys, man-in-the-middle attacks are generally mitigated.

It is strongly RECOMMENDED that all relevant and immutable message data be signed.

It should be noted that a transport-level security protocol such as SSL or TLS [RFC2246] MAY be used to protect the message and the security token as an alternative to or in conjunction with WSS: SOAP Message Security specification [WS-Security].

5 References

The following are normative references

[Glossary]	Informational RFC 2828, *Internet Security Glossary*, May 2000. http://www.ietf.org/rfc/rfc2828.txt
[KEYWORDS]	S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, RFC 2119, Harvard University, March 1997, http://www.ietf.org/rfc/rfc2119.txt
[RFC2246]	T. Dierks, C. Allen., *The TLS Protocol Version, 1.0.* IETF RFC 2246 January 1999. http://www.ietf.org/rfc/rfc2246.txt
[SOAP11]	W3C Note, "SOAP: Simple Object Access Protocol 1.1," 08 May 2000.
[SOAP12]	W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging Framework", 23 June 2003.
[URI]	T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," RFC 3986, MIT/LCS, Day Software, Adobe Systems, January 2005.
[WS-Security]	A. Nadalin et al., Web Services Security: SOAP Message Security 1.1 (WS-Security 2004), OASIS Standard, http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.1.pdf.
[PKCS7]	*PKCS #7: Cryptographic Message Syntax Standard* RSA Laboratories, November 1, 1993. http://www.rsasecurity.com/rsalabs/pkcs/pkcs-7/index.html
[PKIPATH]	http://www.itu.int/rec/recommendation.asp?type=items&lang=e&parent=T-REC-X.509-200110-S!Cor1
[X509]	ITU-T Recommendation X.509 (1997 E): Information Technology - *Open Systems Interconnection - The Directory: Authentication Framework*, June 1997.

The following are non-normative references

[XML-ns]	T. Bray, D. Hollander, A. Layman. *Namespaces in XML. W3C Recommendation.* January 1999. http://www.w3.org/TR/1999/REC-xml-names-19990114
[XML Encrypt]	W3C Recommendation, "XML Encryption Syntax andProcessing," 10 December 2002
[XML Signature]	D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon. *XML-Signature Syntax and Processing*, W3C Recommendation, 12 February 2002.

Appendix A: Acknowledgments

Current Contributors:

Michael	Hu	Actional
Maneesh	Sahu	Actional
Duane	Nickull	Adobe Systems
Gene	Thurston	AmberPoint
Frank	Siebenlist	Argonne National Laboratory
Hal	Lockhart	BEA Systems
Denis	Pilipchuk	BEA Systems
Corinna	Witt	BEA Systems
Steve	Anderson	BMC Software
Rich	Levinson	Computer Associates
Thomas	DeMartini	ContentGuard
Merlin	Hughes	Cybertrust
Dale	Moberg	Cyclone Commerce
Rich	Salz	Datapower
Sam	Wei	EMC
Dana S.	Kaufman	Forum Systems
Toshihiro	Nishimura	Fujitsu
Kefeng	Chen	GeoTrust
Irving	Reid	Hewlett-Packard
Kojiro	Nakayama	Hitachi
Paula	Austel	IBM
Derek	Fu	IBM
Maryann	Hondo	IBM
Kelvin	Lawrence	IBM
Michael	McIntosh	IBM
Anthony	Nadalin	IBM
Nataraj	Nagaratnam	IBM
Bruce	Rich	IBM
Ron	Williams	IBM
Don	Flinn	Individual
Kate	Cherry	Lockheed Martin
Paul	Cotton	Microsoft
Vijay	Gajjala	Microsoft
Martin	Gudgin	Microsoft
Chris	Kaler	Microsoft
Frederick	Hirsch	Nokia
Abbie	Barbir	Nortel
Prateek	Mishra	Oracle
Vamsi	Motukuru	Oracle
Ramana	Turlapi	Oracle
Ben	Hammond	RSA Security
Rob	Philpott	RSA Security
Blake	Dournaee	Sarvega
Sundeep	Peechu	Sarvega

Coumara	Radja	Sarvega
Pete	Wenzel	SeeBeyond
Manveen	Kaur	Sun Microsystems
Ronald	Monzillo	Sun Microsystems
Jan	Alexander	Systinet
Symon	Chang	TIBCO Software
John	Weiland	US Navy
Hans	Granqvist	VeriSign
Phillip	Hallam-Baker	VeriSign
Hemma	Prafullchandra	VeriSign

564 **Previous Contributors:**

Peter	Dapkus	BEA
Guillermo	Lao	ContentGuard
TJ	Pannu	ContentGuard
Xin	Wang	ContentGuard
Shawn	Sharp	Cyclone Commerce
Ganesh	Vaideeswaran	Documentum
Tim	Moses	Entrust
Carolina	Canales-Valenzuela	Ericsson
Tom	Rutt	Fujitsu
Yutaka	Kudo	Hitachi
Jason	Rouault	HP
Bob	Blakley	IBM
Joel	Farrell	IBM
Satoshi	Hada	IBM
Hiroshi	Maruyama	IBM
David	Melgar	IBM
Kent	Tamura	IBM
Wayne	Vicknair	IBM
Phil	Griffin	Individual
Mark	Hayes	Individual
John	Hughes	Individual
Peter	Rostin	Individual
Davanum	Srinivas	Individual
Bob	Morgan	Individual/Internet2
Bob	Atkinson	Microsoft
Keith	Ballinger	Microsoft
Allen	Brown	Microsoft
Giovanni	Della-Libera	Microsoft
Alan	Geller	Microsoft
Johannes	Klein	Microsoft
Scott	Konersmann	Microsoft
Chris	Kurt	Microsoft
Brian	LaMacchia	Microsoft
Paul	Leach	Microsoft
John	Manferdelli	Microsoft
John	Shewchuk	Microsoft
Dan	Simon	Microsoft

Hervey	Wilson	Microsoft
Jeff	Hodges	Neustar
Senthil	Sengodan	Nokia
Lloyd	Burch	Novell
Ed	Reed	Novell
Charles	Knouse	Oblix
Vipin	Samar	Oracle
Jerry	Schwarz	Oracle
Eric	Gravengaard	Reactivity
Andrew	Nash	Reactivity
Stuart	King	Reed Elsevier
Martijn	de Boer	SAP
Jonathan	Tourzan	Sony
Yassir	Elley	Sun
Michael	Nguyen	The IDA of Singapore
Don	Adams	TIBCO
Morten	Jorgensen	Vordel

Appendix B: Revision History

Rev	Date	By Whom	What