INTERNET-DRAFT Donald D. Eastlake 3rd
Motorola
Expires: July 2002 January 2002.

Additional XML Security URIs

<draft-eastlake-xmldsig-uri-02.txt>

Status of This Document

Distribution of this draft is unlimited. It is intended to become an Informational RFC and will probably also be published as a W3C Note. Comments should be sent to the author or the XMLDSIG working group <w3c-ietf-xmldsig@w3.org>.

Copyright Notice

Abstract

A number of algorithm and keying information identifying URIs intended for use with XML Digital Signatures and XML Encryption are defined.

Acknowledgements

Glenn Adams, Merlin Hughs, Brian LaMachia, Joseph Reagle

Table of Contents

1. Introduction

XML Digital Signatures have been standardized by the joint IETF/W3C XMLDSIG working group. The Proposed Standard is specified in [RFC 3075] and a Draft Standard version is pending before the IESG

[XMLDSIG-D]. Canonical XML, which is used by many digital signatures, has been standardized by the W3C and is documented in Informational [RFC 3076]. In addition, XML Encryption [XMLENC] and Exclusive XML Canonicalization [Exclusive] are currently being standardized by the W3C.

All of these standards and recommendations use URIs to identify algorithms and keying information types. This document is intended to be a convenient reference list of URIs and descriptions for algorithms in which there is substantial interest but which can not or have not been included in the main documents for some reason. Note in particular that raising XML digital signature to Draft Standard in the IETF requires remove of any algorithms for which there is not demonstrated interoperability from the main standards document. This requires removal of the Minimal Canonicalization algorithm, in which there appears to be continued interest, to be dropped from the standards track specification. It is included here.

2. Algorithm URIs

The URI being dropped from the standard due to the transition from Proposed Standard to Draft Stanard is included herein with its original

    http://www.w3.org/2000/09/xmldsig#

prefix so as to avoid changing the XMLDSIG standard's namespace. Additional non-proprietary algorithms, particularly those based on USA Government and W3C standards, are given URIs that start with

    http://www.w3.org/2001/04/xmldsig-more

as are some URIs from the on-going XML Encryption standardization effort. An "xmldsig-more" URI does not imply any official W3C status for these algorithms or identifiers. Currently, dereferencing such URIs may produce a temporary placeholder document. Permission to use these URIs has been given by the W3C.

2.1 DigestMethod Algorithms

2.1.1 MD5

Identifier:
    http://www.w3.org/2001/04/xmldsig-more#md5

The MD5 algorithm [RFC 1321] takes no explicit parameters. An example of an MD5 DigestAlgorithm element is:

```
<DigestMethod
    Algorithm="http://www.w3.org/2001/04/xmldsig-more#md5"/>
```

An MD5 digest is a 128-bit string. The content of the DigestValue element shall be the base64 [RFC 2045] encoding of this bit string viewed as a 16-octet octet stream.

2.1.2 SHA-384

Identifier:
    http://www.w3.org/2001/04/xmldsig-more#sha384

The SHA-384 algorithm [SHA-384] takes no explicit parameters. An example of a SHA-384 DigestAlforithm element is:

```
<DigestMethod
    Algorith="http://www.w3.org/2001/04/xmldsig-more#sha384"/>
```

A SHA-384 digest is a 384 bit string. The content of the DigestValue element shall be the base64 [RFC2045] encoding of this string viewed as a 48-octet stream. Because it takes roughly the same amount of effort to compute a SHA-384 message digest as a SHA-512 digest and terseness is usually not a criteria in XML application, use of SHA-512 as an alternative is recommended.

2.2 SignatureMethod Message Authentication Code Algorithms

Some text in this section is duplicated from RFC 3075 for the convenience of the reader.

2.2.1 HMAC-MD5

Identifier:
    http://www.w3.org/2001/04/xmldsig-more#hmac-md5

The HMAC algorithm [RFC 2104] takes the truncation length in bits as a parameter; if the parameter is not specified then all the bits of the hash are output. An example of an HMAC-MD5 SignatureMethod element is as follows:

```
<SignatureMethod
    Algorithm="http://www.w3.org/2001/04/xmldsig-more#hmac-md5">
    <HMACOutputLength>112</HMACOutputLength>
</SignatureMethod>
```

The output of the HMAC algorithm is ultimately the output (possibly truncated) of the chosen digest algorithm. This value shall be base64 [RFC 2405] encoded in the same straightforward fashion as the output of the digest algorithms. Example: the SignatureValue element for the HMAC-MD5 digest

    9294727A 3638BB1C 13F48EF8 158BFC9D

from the test vectors in [RFC 2104] would be

```
<SignatureValue>kpRyejY4uxwT9I74FYv8nQ==</SignatureValue>
```

Schema Definition:

```
<simpleType name="HMACOutputLengthType">
    <restriction base="integer"/>
</simpleType>
```

DTD:

```
<!ELEMENT HMACOutputLength (#PCDATA)>
```

The Schema Definition and DTD immediately above are copied from RFC 3075.

Although some cryptographic suspicions have recently been cast on MD5 for use in signatures such as RSA-MD5 below, this does not effect use of MD5 in HMAC.

2.2.2 HMAC SHA Variations

Identifiers:
    http://www.w3.org/2001/04/xmldsig-more#hmac-sha256

    http://www.w3.org/2001/04/xmldsig-more#hmac-sha384
    http://www.w3.org/2001/04/xmldsig-more#hmac-sha512

SHA-256, SHA-384, and SHA-512 [SHA-256] can also be used in HMAC as described in section 2.2.1 above for HMAC-MD5.

2.2.3 HMAC-RIPEMD160

Identifier:
    http://www.w3.org/2001/04/xmldsig-more#hmac-ripemd160

RIPEMD-160 [RIPEMD-160] can also be used in HMAC as described in section 2.2.1 above for HMAC-MD5.

2.3 SignatureMethod Public Key Signature Algorithms

2.3.1 RSA-MD5

Identifier:
http://www.w3.org/2001/04/xmldsig-more#rsa-md5

This implies the PKCS#1 v1.5 padding algorithm described in [RFC 2437].

An example of use is

```
<SignatureMethod
    Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-md5"/>
```

The SignatureValue content for an RSA-MD5 signature is the base64 [RFC 2405] encoding of the octet string computed as per [RFC 2437], section 8.1.1.

Signature generation for the RSASSA-PKCS1-v1_5 signature scheme. As specified in the EMSA-PKCS1-V1_5-ENCODE function in [RFC 2437, section 9.2.1], the value input to the signature function MUST contain a pre-pended algorithm object identifier for the hash function, but the availability of an ASN.1 parser and recognition of OIDs is not required of a signature verifier. The PKCS#1 v1.5 representation appears as:

```
CRYPT (PAD (ASN.1 (OID, DIGEST (data))))
```

Note that the padded ASN.1 will be of the following form:


```
01 | FF* | 00 | prefix | hash
```

where "|" is concatentation, "01", "FF", and "00" are fixed octets of the corresponding hexadecimal value, "hash" is the MD5 digest of the data, and "prefix" is the ASN.1 BER MD5 algorithm designator prefix required in PKCS #1 [RFC 2437], that is,

```
hex 30 20 30 0c 06 08 2a 86 48 86 f7 0d 02 05 05 00 04 10
```

This prefix is included to make it easier to use standard cryptographic libraries. The FF octet MUST be repeated the maximum number of times such that the value of the quantity being CRYPTed is one octet shorter than the RSA modulus.

Due to increases in computer processor power and advances in cryptography, use of RSA-MD5 is NOT RECOMMENDED.

2.3.2 RSA-SHA256

Identifier:
http://www.w3.org/2001/04/xmldsig-more#rsa-sha256

An example of use is

```
<SignatureMethod
    Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"
    />
```

[I think the SHA-256/384/512 RSA signature algorithms should use PKCS#1 v2, i.e., OAEP.]

2.3.3 RSA-SHA384

Identifier:
    http://www.w3.org/2001/04/xmldsig-more#rsa-sha384

An example of use is

```
<SignatureMethod
    Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha384"
    />
```

Because it takes about the same effort to calculate a SHA-384 message digest as it does a SHA-512 message digest, it is recommended that RSA-SHA512 be used in preference to RSA-SHA384 where possible.

2.3.4 RSA-SHA512

Identifier:
    http://www.w3.org/2001/04/xmldsig-more#rsa-sha512

An example of use is

```
<SignatureMethod
    Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha512"
    />
```

2.3.5

Identifier:
    http://www.w3.org/2001/04/xmldsig-more/rsa-ripemd160

This siganture method uses PKCS#1 padding as described in section 2.3.1. An example of use is

```
<SignatureMethod
 Algorithm="http://www.w3.org/2001/04/xmldsig-more/rsa-ripemd160"
 />
```

2.4 Minimal Canonicalization

At this time two independent interoperable implementations of Minimal

Canonicalization have not been announced. Therefore, when XML Digital Siganture is advanced from Proposed Standard to Draft Standard, it must be dropped from the standard track documents. However, there is still interest and indicates of possible future use for Minimal Canonicalization. For its definition, see [RFC 3075], Section 6.5.1.

For reference, it's identifier remains:
http://www.w3.org/2000/09/xmldsig#minimal

2.5 Transform Algorithms

Note that all CanonicalizationMethod algorithms listed can also be used as Transform algorithms.

2.5.1 XPointer

Identifier:
http://www.w3.org/2001/04/xmldsig-more/xptr

This transform algorithm takes an [XPointer] as an explicit parameter. An example of use is:

```
<Transform
    Algorithm="http://www.w3.org/2001/04/xmldsig-more/xptr">
    <XPointer
        xmlns="http://www.w3.org/2001/04/xmldsig-more/xptr">
        xpointer(id("foo")) xmlns(bar=urn:baz)
            xpointer(//bar:Zab[@Id="foo"])
    </XPointer>
</Transform>
```

Schema Definition:

```
<element name="XPointer" type="string"/>
```

DTD:

```
<!ELEMENT XPointer (#PCDATA)>
```

Input to this transfrom is an octet stream (which is then parsed into XML).

Output from this transform is a node set; the results of the XPointer are processed as defined in the XMLDSIG specification [RFC 3075] for a same-document XPointer.

2.6 ARCFOUR Encryption Algorithm

Identifier:
    http://www.w3.org/2001/04/xmldsgi-more#arcfour

ARCFOUR is a fast, simple stream encryption algorithm that is compatible with RSA Security's RC4 algorithm. An example EncryptionMethod element using ARCFOUR is

```
<EncryptionMethod
 Algorithm="http://www.w3.org/2001/04/xmldsgi-more#arcfour">
 <KeySize>40</KeySize>
</EncryptionMethod>
```


3. KeyInfo

In section 3.1 below a new KeyInfo element child is specified while in section 3.2 additional KeyInfo Type values for use in RetrievalMethod are specified.

3.1 PKCS #7 Bag of Certificates and CRLs

A PKCS #7 [RFC 2315] "signedData" can also be used as a bag of certificates and/or certificate revocation lists (CRLs). The PKCS7signedData element is defined to accomodate such structures within KeyInfo. The binary PKCS #7 strucuture is base64 [RFC 2405] encoded. Any signer information present is ignored. The following is a example, elliding the base64 data:

```
<foo:PKCS7signedData
    xmlns:foo="http://www.w3.org/2001/04/xmldsig-more">
    ...
</foo:PKCS7signedData>
```

3.2 Additional RetrievalMethod Type Values

The Type attribute of RetrievalMethod is an optional identifier for the type of data to be retrieved. The result of de-referencing a RetrievalMethod reference for all KeyInfo types with an XML structure is an XML element or document with that element as the root. The various "raw" key information types return a binary value. Thus they require a Type attibute because they are not unambiguously parseable.

Identifiers:
    http://www.w3.org/2000/09/xmldsig-more#KeyValue
    http://www.w3.org/2000/09/xmldsig-more#RetrievalMethod
    http://www.w3.org/2000/09/xmldsig-more#KeyName
    http://www.w3.org/2000/09/xmldsig-more#rawX509CRL
    http://www.w3.org/2000/09/xmldsig-more#rawPGPKeyPacket
    http://www.w3.org/2000/09/xmldsig-more#rawSPKISexp

http://www.w3.org/2000/09/xmldsig-more#PKCS7signedData
http://www.w3.org/2000/09/xmldsig-more#rawPKCS7signedData

4. IANA Considerations

None. (so far)

5. Security Considerations

Due to computer speed and cryptographic advances, the use of MD5 as a DigestMethod or in the RSA-MD5 SigantureMethod is NOT RECOMMENDED. The cryrptographic advances concerned do not effect the security of HMAC-MD5; however, there is little reason not to go for one of the SHA series of algorithms.

References

[Exclusive] - Exclusive XML Canonicalization Version 1.0, D. Eastlake, J. Reagle, 18 October 2001. <http://www.w3.org/Signature/Drafts/xml-exc-c14n.html>

[RFC 1321] - "The MD5 Message-Digest Algorithm", R. Rivest, April 1992.

[RFC 2104] - "HMAC: Keyed-Hashing for Message Authentication", H. Krawczyk, M. Bellare, R. Canetti, February 1997.

[RFC 2405] - "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed, N. Borenstein, November 1996.

[RFC 2437] - "PKCS #1: RSA Cryptography Specifications Version 2.0", B. Kaliski, J. Staddon, October 1998.

[RFC 2315] - "PKCS #7: Cryptographic Message Syntax Version 1.5", B. Kaliski, March 1998.

[RFC 3075] - "XML-Signature Syntax and Processing", D. Eastlake, J. Reagle, D. Solo, March 2001. <http://www.w3.org/TR/2000/CR-xmldsig-core-20001031>

[RFC 3076] - "Canonical XML Version 1.0", J. Boyer, March 2001. <http://www.w3.org/TR/2001/REC-xml-c14n-20010315>

[RFC 3092] - "Etymology of 'Foo'", D. Eastlake 3rd, C. Manros, E. Raymond, 1 April 2001.

[RIPEMD-160] - ISO/IEC 10118-3:1998, Information Technology - Security techniques - Hash-functions - Part3: Dedicated hash-functions, ISO, 1998.

[SHA-384] - US Federal Information Processing Standard 180-2, Secure Hash Standard, Draft.

[XMLDSIG-D] - XML - Signature Syntax and Processing, D. Eastlake, J. Reagle, D. Solo, August 2001. <http://www.w3.org/TR/2001/PR-xmldsig-core-20010820/> and corresponding internet-draft.

[XMLENC] - XML Encryption Syntax and Processing, J. Reagle, D. Eastlake, October 2001. <http://www.w3.org/TR/2001/WD-xmlenc-core-20011018/>

[XPointer] - "XML Pointer Language (XPointer) Version 1.0", W3C working draft, Steve DeRose, Eve Maler, Ron Daniel Jr., January 2001. <http://www.w3.org/TR/2001/WD-xptr-20010108>

Author's Address

Donald E. Eastlake 3rd
Motorola
155 Beaver Street
Milford, MA 01757 USA

Telephone: +1-508-634-2066 (h)
+1-508-851-8280 (w)
FAX: +1-508-851-8507 (w)
EMail: Donald.Eastlake@motorola.com

Expiration and File Name

This draft expires July 2002.

Its file name is draft-eastlake-xmldsig-uri-02.txt.