

INTERNET-DRAFT
Obsoletes: RFC [4051](#)
Intended Status: Informational
Expires: May 2008

Network Working Group
Donald E. Eastlake 3rd
Motorola Laboratories
November 2007

Additional XML Security Uniform Resource Identifiers (URIs)

<[draft-eastlake-additional-xmlsec-uris-00.txt](#)>

Status of This Document

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with [Section 6 of BCP 79](#).

Distribution of this document is unlimited. Comments should be sent to the author.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

Abstract

This document expands and updates the list of URIs intended for use with XML Digital Signatures, Encryption, Canonicalization, and Key Management specified in [RFC 4051](#). These URIs identify algorithms and types of information.

INTERNET-DRAFT

Additional XML Security URIs

Acknowledgements

The following were the alphabetic acknowledgements listed in [\[RFC4051\]](#): Glenn Adams, Merlin Hughes, Gregor Karlinger, Brian LaMachia, Shiho Moriai, Joseph Reagle, Russ Housley, and Joel Halpern.

Table of Contents

Status of This Document.....	1
Abstract.....	1
Acknowledgements.....	2
Table of Contents.....	2
1 . Introduction.....	4
1.1 Terminology.....	4
2 . Algorithms.....	5
2.1 DigestMethod Algorithms.....	5
2.1.1 MD5.....	5
2.1.2 SHA-224.....	6
2.1.3 SHA-384.....	6
2.2 SignatureMethod Message Authentication Code Algorithms.....	6
2.2.1 HMAC-MD5.....	7
2.2.2 HMAC SHA Variations.....	7
2.2.3 HMAC-RIPEMD160.....	8
2.3 SignatureMethod Public Key Signature Algorithms.....	8
2.3.1 RSA-MD5.....	8
2.3.2 RSA-SHA256.....	9
2.3.3 RSA-SHA384.....	9
2.3.4 RSA-SHA512.....	10
2.3.5 RSA-RIPEMD160.....	10
2.3.6 ECDSA-SHA* and RIPEMD160.....	10
2.3.7 ESIGN-SHA1.....	11
2.4 Minimal Canonicalization.....	11
2.5 Transform Algorithms.....	11

2.5.1	XPointer	12
2.6	EncryptionMethod Algorithms	12
2.6.1	ARCFOUR Encryption Algorithm	12
2.6.2	Camellia Block Encryption	13
2.6.3	Camellia Key Wrap	13
2.6.4	PSEC-KEM	14
3	KeyInfo	15
3.1	PKCS #7 Bag of Certificates and CRLs	15
3.2	Additional RetrievalMethod Type Values	15

Table of Contents Continued

4	URI Index	16
5	Fragment Index	18
6	IANA Considerations	19
7	Security Considerations	19
	Normative References	20
	Informative References	21
	Changes from RFC 4051	23
	Disclaimer	24
	Additional IPR Provisions	24
	Author's Address	25
	Expiration and File Name	25

INTERNET-DRAFT

Additional XML Security URIs

1. Introduction

XML Digital Signatures, Canonicalization, and Encryption have been standardized by the W3C and by the joint IETF/W3C XMLDSIG working group [[W3C](#)]. All of these are now W3C Recommendations and IETF Informational or Standards Track documents. They are available as follows:

IETF level -----	W3C REC -----	Topic -----
[RFC3275] Draft Std	[XMLDSIG]	XML Digital Signatures
[RFC3076] Info	[CANON]	Canonical XML 1.0
- - - - -	[XMLENC]	XML Encryption
[RFC3741] Info	[XCANON]	Exclusive XML Canonicalization 1.0

All of these standards and recommendations use URIs [[RFC3986](#)] to identify algorithms and keying information types. This document is 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 required remove of any algorithms for which there was not

demonstrated interoperability from the main standards document. This required removal of the Minimal Canonicalization algorithm, in which there appears to be continued interest, to be dropped from the standards track specification. It was included in [\[RFC4051\]](#) and is included here.

[1.1](#) Terminology

Notwithstanding that this is an Informational document, standards track type terms [\[RFC2119\]](#) are used in specifying the use of some of the URIs as follows:

The key words "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](#).

[2](#). Algorithms

The URI [\[RFC3986\]](#) being dropped from the standard due to the transition from Proposed Standard to Draft Standard is included in [Section 2.4](#) below with its original

<http://www.w3.org/2000/09/xmlsig#>

prefix so as to avoid changing the XMLSIG standard's namespace.

Additional algorithms in [\[RFC4051\]](#) were given URIs that start with

<http://www.w3.org/2001/04/xmlsig-more#>

while further algorithms added in this document are given URIs that

start with

<http://www.w3.org/2007/05/xmldsig-more#>

An "xmldsig-more" URI does not imply any official W3C status for these algorithms or identifiers nor does it imply that they are only useful in digital signatures. Currently, dereferencing such URIs may or may not produce a temporary placeholder document. Permission to use these URI prefixes has been given by the W3C.

[2.1](#) DigestMethod Algorithms

These algorithms are usable wherever a DigestMethod element occurs.

[2.1.1](#) MD5

Identifier:

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

The MD5 algorithm [[RFC1321](#)] takes no explicit parameters. An example of an MD5 DigestAlgorithm element is:

```
<DigestAlgorithm
  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 [[RFC2045](#)] encoding of this bit string viewed as a 16-octet octet stream.

[2.1.2](#) SHA-224

Identifier:

<http://www.w3.org/2001/04/xmldsig-more#sha224>

The SHA-224 algorithm [FIPS180-2?] [[RFC4634](#)] takes no explicit parameters. An example of a SHA-224 DigestAlgorithm element is:

```
<DigestAlgorithm
  Algorithm="http://www.w3.org/2001/04/xmldsig-more#sha224" />
```

A SHA-224 digest is a 224 bit string. The content of the DigestValue element shall be the base64 [RFC2045] encoding of this string viewed as a 28-octet stream. Because it takes roughly the same amount of effort to compute a SHA-224 message digest as a SHA-256 digest and terseness is usually not a criteria in XML application, consideration should be given to the use of SHA-256 as an alternative.

[2.1.3](#) SHA-384

Identifier:

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

The SHA-384 algorithm [FIPS 180-2] takes no explicit parameters. An example of a SHA-384 DigestAlgorithm element is:

```
<DigestAlgorithm
  Algorithm="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, consideration should be given to the use of SHA-512 as an alternative.

[2.2](#) SignatureMethod Message Authentication Code Algorithms

Note: Some text in this section is duplicated from [RFC3275] for the convenience of the reader. [RFC 3275](#) is normative in case of conflict.

[2.2.1](#) HMAC-MD5

Identifier:

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

The HMAC algorithm [[RFC2104](#)] 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 [[RFC2045](#)] 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 [[RFC2104](#)] would be

kpRyejY4uxwT9I74FYv8nQ==

Schema Definition:

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

DTD:

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

The Schema Definition and DTD immediately above are copied from [[RFC3275](#)].

Although 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-sha224>

INTERNET-DRAFT

Additional XML Security URIs

<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-224, SHA-256, SHA-384, and SHA-512 [FIPS180-2?] [RFC4634] 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

These algorithms are distinguished from those in [Section 2.2](#) above in that they use public key methods. That is to say, the verification key is different from and not feasibly derivable from the signing key.

[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 [RFC3447]. 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

[RFC2045] encoding of the octet string computed as per [RFC3447] [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 [RFC3447] [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

Vertical bar ("|") represents concatenation. "01", "FF", and "00" are fixed octets of the corresponding hexadecimal value and the asterisk ("*") after "FF" indicates repetition. "hash" is the MD5 digest of the data. "prefix" is the ASN.1 BER MD5 algorithm designator prefix required in PKCS #1 [RFC3447], 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 enough times that the value of the quantity being CRYPTed is exactly 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>

This implies the PKCS#1 v1.5 padding algorithm [RFC3447] as described in [section 2.3.1](#) but with the ASN.1 BER SHA-256 algorithm designator prefix. An example of use is

<SignatureMethod

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

[2.3.3](#) RSA-SHA384

Identifier:

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

This implies the PKCS#1 v1.5 padding algorithm [[RFC3447](#)] as described in [section 2.3.1](#) but with the ASN.1 BER SHA-384 algorithm designator prefix. 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 suggested 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>

This implies the PKCS#1 v1.5 padding algorithm [[RFC3447](#)] as described in [section 2.3.1](#) but with the ASN.1 BER SHA-512 algorithm designator prefix. An example of use is

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

[2.3.5](#) RSA-RIPEMD160

Identifier:

<http://www.w3.org/2001/04/xmldsig-more#rsa-ripemd160>

This implies the PKCS#1 v1.5 padding algorithm [[RFC3447](#)] as described in [section 2.3.1](#) but with the ASN.1 BER RIPEMD160 algorithm designator prefix. An example of use is

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

[2.3.6](#) ECDSA-SHA* and RIPEMD160

Identifiers

<http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha1>
<http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha224>
<http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256>
<http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha384>
<http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha512>
<http://www.w3.org/2007/05/xmldsig-more#ecdsa-ripemd160>

The Elliptic Curve Digital Signature Algorithm (ECDSA) [FIPS 186-2] is the elliptic curve analogue of the DSA (DSS) signature method. For a detailed specifications of how to use it with SHA hash functions and XML Digital Signature, please see [[X9.62](#)] and [[RFC4050](#)]. The

#ecdsa-ripemd160 fragment of the new namespace identifies a signature method processed in the same way as specified by the #ecdsa-sha1 fragment of this namespace with the exception that RIPEMD160 is used instead of SHA-1.

[2.3.7](#) ESIGN-SHA1

Identifier

<http://www.w3.org/2001/04/xmldsig-more#esign-sha1>
<http://www.w3.org/2001/04/xmldsig-more#esign-sha224>
<http://www.w3.org/2001/04/xmldsig-more#esign-sha256>
<http://www.w3.org/2001/04/xmldsig-more#esign-sha384>
<http://www.w3.org/2001/04/xmldsig-more#esign-sha512>

The ESIGN algorithm specified in [IEEE P1363a] is a signature scheme based on the integer factorization problem. It is much faster than

previous digital signature schemes so E-SIGN can be implemented on smart cards without special co-processors.

An example of use is

```
<SignatureMethod
  Algorithm="http://www.w3.org/2001/04/xmldsig-more#esign-sha1"
/>
```

[2.4](#) Minimal Canonicalization

Thus far two independent interoperable implementations of Minimal Canonicalization have not been announced. Therefore, when XML Digital Signature was advanced from Proposed Standard [[RFC3075](#)] to Draft Standard [[RFC3275](#)], Minimal Canonicalization was dropped from the standard track documents. However, there is still interest. For its definition, see [[RFC3075](#)] [Section 6.5.1](#).

For reference, its identifier remains:

<http://www.w3.org/2000/09/xmldsig#minimal>

[2.5](#) Transform Algorithms

Note that all CanonicalizationMethod algorithms 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=http://foobar.example)
      xpointer(//bar:Zab[@Id="foo"])
    </XPointer>
  </Transform>

```

Schema Definition:

```

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

```

DTD:

```

    <!ELEMENT XPointer (#PCDATA) >

```

Input to this transform 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 [[RFC3275](#)] for a same-document XPointer.

[2.6](#) EncryptionMethod Algorithms

This subsection gives identifiers and information for several EncryptionMethod Algorithms.

[2.6.1](#) ARCFOUR Encryption Algorithm

Identifier:

<http://www.w3.org/2001/04/xmldsig-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/xmldsig-more#arcfour">

```

```
<KeySize>40<KeySize>
</EncryptionMethod>
```

Note that Arcfour makes use of the generic KeySize parameter specified and defined in [\[XMLENC\]](#).

[2.6.2](#) Camellia Block Encryption

Identifiers:

<http://www.w3.org/2001/04/xmldsig-more#camellia128-cbc>
<http://www.w3.org/2001/04/xmldsig-more#camellia192-cbc>
<http://www.w3.org/2001/04/xmldsig-more#camellia256-cbc>

Camellia is an efficient and secure block cipher with the same interface as the AES [\[Camellia\]](#) [\[RFC3713\]](#), that is 128-bit block size and 128, 192, and 256 bit key sizes. In XML Encryption Camellia is used in the same way as the AES: It is used in the Cipher Block Chaining (CBC) mode with a 128-bit initialization vector (IV). The resulting cipher text is prefixed by the IV. If included in XML output, it is then base64 encoded. An example Camellia EncryptionMethod is as follows:

```
<EncryptionMethod
  Algorithm=
    "http://www.w3.org/2001/04/xmldsig-more#camellia128-cbc"
/>
```

[2.6.3](#) Camellia Key Wrap

Identifiers:

<http://www.w3.org/2001/04/xmldsig-more#kw-camellia128>
<http://www.w3.org/2001/04/xmldsig-more#kw-camellia192>
<http://www.w3.org/2001/04/xmldsig-more#kw-camellia256>

Camellia [\[Camellia\]](#) [\[RFC3713\]](#) key wrap is identical to the AES key wrap algorithm [\[RFC3394\]](#) specified in the XML Encryption standard with "AES" replaced by "Camellia". As with AES key wrap, the check value is 0xA6A6A6A6A6A6A6A6.

The algorithm is the same whatever the size of the Camellia key used in wrapping, called the key encrypting key or KEK. The implementation of Camellia is OPTIONAL. However, if it is supported, the same implementation guidelines as to which combinations of KEK size and wrapped key size should be required to be supported and which are

optional to be supported should be followed. That is to say, if Camellia key wrap is supported, they wrapping 128-bit keys with a 128-bit KEK and wrapping 256-bit keys with a 256-bit KEK are REQUIRED and all other combinations are OPTIONAL.

An example of use is:

```
<EncryptionMethod
  Algorithm=
    "http://www.w3.org/2001/04/xmldsig-more#kw-camellia128"
/>
```

[2.6.4](#) PSEC-KEM

Identifier:

<http://www.w3.org/2001/04/xmldsig-more#psec-kem>

The PSEC-KEM algorithm, specified in [[18033-2](#)], is a key encapsulation mechanism using elliptic curve encryption.

An example of use is:

```
<EncryptionMethod
  Algorithm="http://www.w3.org/2001/04/xmlenc#psec-kem">
  <ECParameters>
    <Version>version</Version>
    <FieldID>id</FieldID>
    <Curve>curve</Curve>
    <Base>base</Base>
    <Order>order</Order>
    <Cofactor>cofactor</Cofactor>
  </ECParameters>
</EncryptionMethod>
```

See [[18033-2](#)] for information on the parameters above.

[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 [[RFC2315](#)] "signedData" can also be used as a bag of certificates and/or certificate revocation lists (CRLs). The PKCS7signedData element is defined to accommodate such structures within KeyInfo. The binary PKCS #7 structure is base64 [[RFC2045](#)] encoded. Any signer information present is ignored. The following is an example [[RFC3092](#)], eliding 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 attribute because they are not unambiguously parseable.

Identifiers:

<http://www.w3.org/2001/04/xmldsig-more#KeyValue>
<http://www.w3.org/2001/04/xmldsig-more#RetrievalMethod>
<http://www.w3.org/2001/04/xmldsig-more#KeyName>

<http://www.w3.org/2001/04/xmldsig-more#rawX509CRL>
<http://www.w3.org/2001/04/xmldsig-more#rawPGPKeyPacket>
<http://www.w3.org/2001/04/xmldsig-more#rawSPKISexp>
<http://www.w3.org/2001/04/xmldsig-more#PKCS7signedData>
<http://www.w3.org/2001/04/xmldsig-more#rawPKCS7signedData>

[4.](#) URI Index

The following is an index by URI of the algorithm and KeyInfo URIs defined in this document and in the standards (plus the one KeyInfo child element name defined in this document). The "Sec/Doc" column has the section of this document or, if not specified in this document, the standards document where the item is specified.

The initial "http://www.w3.org/" part of the URI is not included:

URI	Sec/Doc	Type
---	-----	----
2000/09/xmldsig#base64	[RFC3275]	Transform
2000/09/xmldsig#dsa-sha1	[RFC3275]	SignatureMethod
2000/09/xmldsig#enveloped-signature	[RFC3275]	Transform
2000/09/xmldsig#hmac-sha1	[RFC3275]	SignatureMethod
2000/09/xmldsig#minimal	2.4	Canonicalization
2000/09/xmldsig@rsa-sha1	[RFC3275]	SignatureMethod
2000/09/xmldsig#sha1	[RFC3275]	DigestAlgorithm
2001/04/xmldsig-more#arcfour	2.6.1	EncryptionMethod
2001/04/xmldsig-more#camellia128-cbc	2.6.2	EncryptionMethod
2001/04/xmldsig-more#camellia192-cbc	2.6.2	EncryptionMethod
2001/04/xmldsig-more#camellia256-cbc	2.6.2	EncryptionMethod
2001/04/xmldsig-more#ecdsa-sha1	2.3.6	SignatureMethod
2001/04/xmldsig-more#ecdsa-sha224	2.3.6	SignatureMethod
2001/04/xmldsig-more#ecdsa-sha256	2.3.6	SignatureMethod
2001/04/xmldsig-more#ecdsa-sha384	2.3.6	SignatureMethod
2001/04/xmldsig-more#ecdsa-sha512	2.3.6	SignatureMethod

2001/04/xmldsig-more#esign-sha1	2.3.7	SignatureMethod
2001/04/xmldsig-more#esign-sha224	2.3.7	SignatureMethod
2001/04/xmldsig-more#esign-sha256	2.3.7	SignatureMethod
2001/04/xmldsig-more#esign-sha384	2.3.7	SignatureMethod
2001/04/xmldsig-more#esign-sha512	2.3.7	SignatureMethod
2001/04/xmldsig-more#hmac-md5	2.2.1	SignatureMethod
2001/04/xmldsig-more#hmac-ripemd160	2.2.3	SignatureMethod
2001/04/xmldsig-more#hmac-sha224	2.2.2	SignatureMethod
2001/04/xmldsig-more#hmac-sha256	2.2.2	SignatureMethod
2001/04/xmldsig-more#hmac-sha384	2.2.2	SignatureMethod
2001/04/xmldsig-more#hmac-sha512	2.2.2	SignatureMethod
2001/04/xmldsig-more#KeyName	3.2	Retrieval type
2001/04/xmldsig-more#KeyValue	3.2	Retrieval type
2001/04/xmldsig-more#kw-camellia128	2.6.3	EncryptionMethod
2001/04/xmldsig-more#kw-camellia192	2.6.3	EncryptionMethod
2001/04/xmldsig-more#kw-camellia256	2.6.3	EncryptionMethod
2001/04/xmldsig-more#md5	2.1.1	DigestAlgorithm
2001/04/xmldsig-more#PKCS7signedData	3.2	Retrieval type
2001/04/xmldsig-more#psec-kem	2.6.4	EncryptionMethod
2001/04/xmldsig-more#rawPGPKeyPacket	3.2	Retrieval type

2001/04/xmldsig-more#rawPKCS7signedData	3.2	Retrieval type
2001/04/xmldsig-more#rawSPKISexp	3.2	Retrieval type
2001/04/xmldsig-more#rawX509CRL	3.2	Retrieval type
2001/04/xmldsig-more#RetrievalMethod	3.2	Retrieval type
2001/04/xmldsig-more#rsa-md5	2.3.1	SignatureMethod
2001/04/xmldsig-more#rsa-sha256	2.3.2	SignatureMethod
2001/04/xmldsig-more#rsa-sha384	2.3.3	SignatureMethod
2001/04/xmldsig-more#rsa-sha512	2.3.4	SignatureMethod
2001/04/xmldsig-more#rsa-ripemd160	2.3.5	SignatureMethod
2001/04/xmldsig-more#sha224	2.1.2	DigestAlgorithm
2001/04/xmldsig-more#sha384	2.1.3	DigestAlgorithm
2001/04/xmldsig-more#xptra	2.5.1	Transform
2001/04/xmldsig-more:PKCS7signedData	3.1	KeyInfo child
2001/04/xmlenc#aes128-cbc	[XMLENC]	EncryptionMethod
2001/04/xmlenc#aes192-cbc	[XMLENC]	EncryptionMethod
2001/04/xmlenc#aes256-cbc	[XMLENC]	EncryptionMethod
2001/04/xmlenc#dh	[XMLENC]	AgreementMethod
2001/04/xmlenc#kw-aes128	[XMLENC]	EncryptionMethod
2001/04/xmlenc#kw-aes192	[XMLENC]	EncryptionMethod
2001/04/xmlenc#kw-aes256	[XMLENC]	EncryptionMethod
2001/04/xmlenc#ripemd160	[XMLENC]	DigestAlgorithm

2001/04/xmlenc#rsa-1_5	[XMLENC]	EncryptionMethod
2001/04/xmlenc#rsa-oaep-mbg1p	[XMLENC]	EncryptionMethod
2001/04/xmlenc#sha256	[XMLENC]	DigestAlgorithm
2001.04/xmlend#sha512	[XMLENC]	DigestAlgorithm
2001/04/xmlenc#tripledes-cbc	[XMLENC]	EncryptionMethod
2007/05/xmldsig-more#ecdsa-ripemd160	2.3.6	SignatureMethod
TR/1999/REC-xpath-19991116	[XPATH]	Transform
TR/1999/REC-xslt-19991116	[XSLT]	Transform
TR/2001/06/xml-excl-c14n#	[XCANON]	Canonicalization
TR/2001/06/xml-excl-c14n#WithComments	[XCANON]	Canonicalization
TR/2001/REC-xml-c14n-20010315	[CANON]	Canonicalization
TR/2001/REC-xml-c14n-20010315#WithComments	[CANON]	Canonicalization
TR/2001/REC-xmlschema-1-20010502	[Schema]	Transform

[5.](#) Fragment Index

The following is an index as in [Section 4](#) but sorted by the fragment portion of the URI:

Prefix	Fragment	Sec/Doc	Type
-----	-----	-----	----

TBD

6. IANA Considerations

None.

As it is easy for people to construct their own unique URIs [[RFC3986](#)] and, if appropriate, to obtain a URI from the W3C, it is not intended

that any additional "http://www.w3.org/2007/05/xmlsig-more#" URIs be created beyond those enumerated in this RFC. (W3C Namespace stability rules prohibit the creation of new URIs under "http://www.w3.org/2000/09/xmlsig#" and URIs under "http://www.w3.org/2001/04/#xmlsig-more" were frozen with the publication of [[RFC4051](#)].)

[7](#). Security Considerations

Due to computer speed and cryptographic advances, the use of MD5 as a DigestMethod or in the RSA-MD5 SignatureMethod is NOT RECOMMENDED. The cryptographic 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.

Additional security considerations are given in connection with the description of some algorithms in the body of this document.

Normative References

[18033-2] - "Information technology -- Security techniques -- Encryption algorithms -- Part 3: Asymmetric ciphers", ISO/IEC 18033-2, October 2002.

[Camellia] - "Camellia: A 128-bit Block Cipher Suitable for Multiple Platforms - Design and Analysis -", K. Aoki, T. Ichikawa, M. Matsui, S. Moriai, J. Nakajima, T. Tokita, In Selected Areas in Cryptography, 7th Annual International Workshop, SAC 2000, August 2000, Proceedings, Lecture Notes in Computer Science 2012, pp. 39-56, Springer-Verlag, 2001.

[FIPS 180-2] - "Secure Hash Standard", (SHA-1/256/384/512) US Federal Information Processing Standard, Draft, not yet issued.

[FIPS 180-2change] - "FIPS 180-2, Secure Hash Standard Change Notice 1", adds SHA-224 to [FIPS 180-2].

[FIPS 186-2] - "Digital Signature Standard", National Institute of Standards and Technology, 2000.

[IEEE P1363a] - "Standard Specifications for Public Key Cryptography: Additional Techniques", October 2002.

[RFC1321] - Rivest, R., "The MD5 Message-Digest Algorithm", [RFC 1321](#), April 1992.

[RFC2045] - Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", [RFC 2045](#), November 1996.

[RFC2104] - Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", [RFC 2104](#), February 1997.

[RFC2119] - Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.

[RFC2315] - Kaliski, B., "PKCS #7: Cryptographic Message Syntax Version 1.5", [RFC 2315](#), March 1998.

[RFC3275] - Eastlake 3rd, D., Reagle, J., and D. Solo, "(Extensible Markup Language) XML-Signature Syntax and Processing", [RFC 3275](#), March 2002.

[RFC3394] - Schaad, J. and R. Housley, "Advanced Encryption Standard (AES) Key Wrap Algorithm", [RFC 3394](#), September 2002.

[RFC3447] - Jonsson, J. and B. Kaliski, "Public-Key Cryptography

INTERNET-DRAFT

Additional XML Security URIs

[RFC 3447](#), February 2003.

[RFC3713] - Matsui, M., Nakajima, J., and S. Moriai, "A Description of the Camellia Encryption Algorithm", [RFC 3713](#), April 2004.

[RFC3986] - Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, [RFC 3986](#), January 2005.

[RFC4050] - Blake-Wilson, S., Karlinger, G., Kobayashi, T., and Y. Wang, "Using the Elliptic Curve Signature Algorithm (ECDSA) for XML Digital Signatures", [RFC 4050](#), April 2005.

[RFC4634] - Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and HMAC-SHA)", [RFC 4634](#), July 2006.

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

[X9.62] - X9.62-200X, "Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)", Accredited Standards Committee X9, American National Standards Institute.

[XMLENC] - "XML Encryption Syntax and Processing", J. Reagle, D. Eastlake, December 2002. <<http://www.w3.org/TR/2001/RED-xmlenc-core-20021210/>>

[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>>

Informative References

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

[RFC3075] - Eastlake 3rd, D., Reagle, J., and D. Solo, "XML-Signature

Syntax and Processing", [RFC 3075](#), March 2001.

[RFC3076] - Boyer, J., "Canonical XML Version 1.0", [RFC 3076](#), March 2001.

[RFC3092] - Eastlake 3rd, D., Manros, C., and E. Raymond, "Etymology of "Foo"", [RFC 3092](#), April 1 2001.

[RFC3741] - Boyer, J., Eastlake 3rd, D., and J. Reagle, "Exclusive

XML Canonicalization, Version 1.0", [RFC 3741](#), March 2004.

[RFC4051] - Eastlake 3rd, D., "Additional XML Security Uniform Resource Identifiers (URIs)", [RFC 4051](#), April 2005.

[Schema] -

[W3C] - World Wide Web Consortium, <<http://www.w3.org>>.

[XCANON] - "Exclusive XML Canonicalization Version 1.0", D. Eastlake, J. Reagle, 18 July 2002. <<http://www.w3.org/TR/REC-xml-enc-c14n-20020718/>>.

[XMLDSIG] -

[XPath] -

[XSLT] -

INTERNET-DRAFT

Additional XML Security URIs

Changes from [RFC 4051](#)

Note to RFC Editor: This section should be deleted on publication.

1. Update numerous RFC and Internet-Draft references.
2. Add #ecdsa-ripemd160.
3. Incorporate [RFC 4051](#) errata.
4. Add URI and Fragment index sections.

INTERNET-DRAFT

Additional XML Security URIs

Disclaimer

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Additional IPR Provisions

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights 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; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in [BCP 78](#) and [BCP 79](#).

Copies of IPR disclosures made to the IETF Secretariat 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 implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in [BCP 78](#), and except as set forth therein, the authors retain all their rights.

Author's Address

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

Telephone: +1-508-786-7554 (w)
EMail: Donald.Eastlake@motorola.com

Expiration and File Name

This draft expires in May 2008.

Its file name is [draft-eastlake-additional-xmlsec-uris-00.txt](#)