

INTERNET-DRAFT

Expires: March 2005

Donald E. Eastlake 3rd  
Motorola Laboratories  
September 2004

Additional XML Security URIs

-----  
<[draft-eastlake-xmldsig-uri-09.txt](#)>

Donald E. Eastlake 3rd

Status of This Document

By submitting this Internet-Draft, I certify that any applicable patent or other IPR claims of which I am aware have been disclosed, or will be disclosed, and any of which I become aware will be disclosed, in accordance with [RFC 3668](#).

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 a "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>

Copyright (C) 2004 The Internet Society. All Right Reserved.

Abstract

A number of URIs intended for use with XML Digital Signatures, Encryption, and Canonicalization are defined. These URIs identify algorithms and types of keying information.

Acknowledgements

Glenn Adams, Merlin Hughs, Gregor Karlinger, Brian LaMachia, Shiho Moriai, Joseph Reagle, Russ Housley, and Joel Halpern.

INTERNET-DRAFT

Additional XML Security URIs

## Table of Contents

Status of This Document.....	<a href="#">1</a>
Abstract.....	<a href="#">1</a>
Acknowledgements.....	<a href="#">1</a>
Table of Contents.....	<a href="#">2</a>
<a href="#">1</a> . Introduction.....	<a href="#">3</a>
<a href="#">2</a> . Algorithms.....	<a href="#">4</a>
<a href="#">2.1</a> DigestMethod Algorithms.....	<a href="#">4</a>
<a href="#">2.1.1</a> MD5.....	<a href="#">4</a>
<a href="#">2.1.2</a> SHA-224.....	<a href="#">4</a>
<a href="#">2.1.3</a> SHA-384.....	<a href="#">5</a>
2.2 SignatureMethod Message Authentication Code Algorithms.....	5
<a href="#">2.2.1</a> HMAC-MD5.....	<a href="#">5</a>
<a href="#">2.2.2</a> HMAC SHA Variations.....	<a href="#">6</a>
<a href="#">2.2.3</a> HMAC-RIPEMD160.....	<a href="#">7</a>
<a href="#">2.3</a> SignatureMethod Public Key Signature Algorithms.....	<a href="#">7</a>
<a href="#">2.3.1</a> RSA-MD5.....	<a href="#">7</a>
<a href="#">2.3.2</a> RSA-SHA256.....	<a href="#">8</a>
<a href="#">2.3.3</a> RSA-SHA384.....	<a href="#">8</a>
<a href="#">2.3.4</a> RSA-SHA512.....	<a href="#">9</a>
<a href="#">2.3.5</a> RSA-RIPEMD160.....	<a href="#">9</a>
<a href="#">2.3.6</a> ECDSA-SHA*.....	<a href="#">9</a>
<a href="#">2.3.7</a> ESIGN-SHA1.....	<a href="#">9</a>
<a href="#">2.4</a> Minimal Canonicalization.....	<a href="#">10</a>
<a href="#">2.5</a> Transform Algorithms.....	<a href="#">10</a>
<a href="#">2.5.1</a> XPointer.....	<a href="#">10</a>
<a href="#">2.6</a> EncryptionMethod Algorithms.....	<a href="#">11</a>
<a href="#">2.6.1</a> ARCFOUR Encryption Algorithm.....	<a href="#">11</a>
<a href="#">2.6.2</a> Camellia Block Encryption.....	<a href="#">12</a>
<a href="#">2.6.3</a> Camellia Key Wrap.....	<a href="#">12</a>
<a href="#">2.6.4</a> PSEC-KEM.....	<a href="#">13</a>
<a href="#">3</a> . KeyInfo.....	<a href="#">13</a>
<a href="#">3.1</a> PKCS #7 Bag of Certificates and CRLs.....	<a href="#">13</a>
<a href="#">3.2</a> Additional RetrievalMethod Type Values.....	<a href="#">14</a>

<a href="#">4. IANA Considerations.....</a>	<a href="#">15</a>
<a href="#">5. Security Considerations.....</a>	<a href="#">15</a>
<a href="#">6. Copyright and Disclaimer.....</a>	<a href="#">15</a>
Normative References.....	<a href="#">16</a>
Informative References.....	<a href="#">17</a>
Author's Address.....	<a href="#">19</a>
Expiration and File Name.....	<a href="#">19</a>

## [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
-----		-----	-----
<a href="#">[RFC 3275]</a>	Draft Std	<a href="#">[XMLDSIG]</a>	XML Digital Signatures
<a href="#">[RFC 3076]</a>	Info	<a href="#">[CANON]</a>	Canonical XML
- - - - -		<a href="#">[XMLENC]</a>	XML Encryption
<a href="#">[RFC 3741]</a>	Info	<a href="#">[EXCANON]</a>	Exclusive XML Canonicalization

All of these standards and recommendations use URIs [\[RFC 2396\]](#) 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 is included here.

## [2.](#) Algorithms

The URI [[RFC 2396](#)] 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 are given URIs that start with

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

An "xmlsig-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 this URI prefix 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 [[RFC 1321](#)] 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 [[RFC 2045](#)] 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 [FIPS 180-2change, [RFC 3874](#)] 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 [RFC 3275] for the convenience of the reader. 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 [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

kpRyejY4uxwT9I74FYv8nQ==

Schema Definition:

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

DTD:

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

The Schema Definition and DTD immediately above are copied from [RFC 3275].

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-sha224>  
<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 [FIPS 180-2, FIPS 180-2change, [RFC 3874](#)] 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 [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

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



## INTERNET-DRAFT

## Additional XML Security URIs

the data. "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 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 [[RFC 2437](#)] 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 [[RFC 2437](#)] 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 [[RFC 2437](#)] 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 [[RFC 2437](#)] 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\*

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>

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 [ECDSA].

### [2.3.7](#) ESIGN-SHA1

Identifier

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

D. Eastlake 3rd

[Page 9]

---

INTERNET-DRAFT

Additional XML Security URIs

<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 ESIGN 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 [RFC 3075] to Draft Standard [RFC 3275], Minimal Canonicalization was 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, its identifier remains:  
<http://www.w3.org/2000/09/xmlsig#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/xmlsig-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/xmlsig-more/xptr">
```

```
<XPointer
  xmlns="http://www.w3.org/2001/04/xmlsig-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 [[RFC 3275](#)] 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/xmlsig-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/xmlsig-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/xmlsig-more#camellia128-cbc>  
<http://www.w3.org/2001/04/xmlsig-more#camellia192-cbc>  
<http://www.w3.org/2001/04/xmlsig-more#camellia256-cbc>

Camellia is an efficient and secure block cipher with the same interface as the AES [[Camellia](#), [RFC 3713](#)], 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, [RFC 3713](#)] key wrap is identical to the AES key wrap algorithm [[RFC 3394](#)] 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 [ISO/IEC 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 [ISO/IEC 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 [[RFC 2315](#)] "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 [[RFC 2405](#)] encoded. Any signer information present is ignored. The following is an example, 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>



---

INTERNET-DRAFT

Additional XML Security URIs

#### [4.](#) IANA Considerations

None.

As it is easy for people to construct their own unique URIs [RFC 2396] and, possible, if appropriate, to obtain a URI from the W3C, it is not intended that any additional "http://www.w3.org/2001/04/xmldsig-more#" URIs be created beyond those enumerated in this document. (W3C Namespace stability rules prohibit the creation of new URIs under "http://www.w3.org/2000/09/xmldsig#".)

#### [5.](#) 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.

#### [6.](#) Copyright and Disclaimer

Copyright (C) 2004 The Internet Society. 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.

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 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.

---

INTERNET-DRAFT

Additional XML Security URIs

#### Normative References

[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.

[ECDSA] - "ECDSA with XML-Signature Syntax", S. Blake-Wilson, G. Karlinger, T. Kobayashi, Y. Want, January 2004. [draft-blake-wilson-xmldsig-ecdsa-\\*.txt](#)

[FIPS 180-1] - "Secure Hash Standard", (SHA-1) US Federal Information Processing Standard, 17 April 1995.

[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.

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

2002.

[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 2119] - "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997.

[RFC 2396] - "Uniform Resource Identifiers (URI): Generic Syntax", T. Berners-Lee, R. Fielding, L. Masinter, August 1998.

[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.

D. Eastlake 3rd

[Page 16]

---

INTERNET-DRAFT

Additional XML Security URIs

[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. ([RFC 3075](#) was obsoleted by [RFC 3275](#) but is referenced in this document for its description of Minimal Canonicalization which was dropped in [RFC 3275](#).)

[RFC 3275] - "XML-Signature Syntax and Processing", D. Eastlake, J. Reagle, D. Solo, March 2002.

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

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

[RFC 3874] - "A 224-bit One-way Hash Function: SHA-224", R. Housley, September 2004.

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

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

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

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

[RFC 3741] - "Exclusive XML Canonicalization Version 1.0", J. Boyer, D. Eastlake 3rd, J. Reagle, March 2004.



Telephone: +1-508-786-7554 (w)  
+1-508-634-2066 (h)  
EMail: Donald.Eastlake@motorola.com

#### Expiration and File Name

This draft expires in March 2005.

Its file name is [draft-eastlake-xmlsig-uri-09.txt](#)