Open Document Format for Office Applications (OpenDocument) Version 1.2

Part 3: Packages – ENCRYPTION ONLY

Committee Draft 05 - ENCRYPTION ONLY CHANGES

Table of Contents

1 Introduction

1.1 Introduction [no changes]

1.2 Terminology [no changes]

1.3 Normative References

[**Blowfish**] Bruce Schneier, Applied Cryptography (Second Edition), John Wiley & Sons, ISBN: 0-471-11709-9, 1996

[**ISO/IEC Directives**] ISO/IEC Directives, Part 2 (Fifth Edition) *Rules for the structure and drafting of International Standards*, International Organization for Standardization and International Electrotechnical Commission, 2004

[**OWL**] Deborah L. McGuinness, Frank van Harmelen, *OWL Web Ontology Language Overview*, http://www.w3.org/TR/2004/REC-owl-features-20040210/, W3C, 2004.

[**PNG**] David Duce, *Portable Network Graphics (PNG) Specification (Second Edition)*, http://www.w3.org/TR/2003/REC-PNG-20031110, W3C, 2003.

[**RDF-CONCEPTS**] Graham Klyne, Jeremy J. Carroll, Brian McBride, *Resource Description Framework (RDF): Concepts and Abstract Syntax*, http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/, W3C, 2004.

[**RDF-XML**] Dave Beckett, Brian McBride, *RDF/XML Syntax Specification (Revised)*, http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/, W3C, 2004.

[**RFC2898**] B. Kaliski, *PKCS #5: Password-Based Cryptography Specification Version 2.0*, http://www.ietf.org/rfc/rfc2898.txt, IETF, 2000.

[**RFC3174**] D. Eastlake, 3rd, P. Jones, *US Secure Hash Algorithm 1 (SHA1)*, http://www.ietf.org/rfc/rfc3174.txt, IETF, 2001.

[**RFC3986**] T. Berners-Lee, R. Fielding, L. Masinter, *Uniform Resource Identifier (URI): Generic Syntax*, http://www.ietf.org/rfc/rfc3986.txt, IETF, 2005.

[**RFC4288**] N. Freed, J. Klensin, *Media Type Specifications and Registration Procedures*, http://www.ietf.org/rfc/rfc4288.txt, IETF, 2005.

[**RNG**] ISO/IEC 19757-2 *Document Schema Definition Language (DSDL) -- Part 2: Regular-grammar-based validation -- RELAX NG*, International Organization for Standardization and International Electrotechnical Commission, 2003

[**XAdES**] XML Advanced Electronic Signatures (XAdES) (ETSI TS 101 903 v1.3.2 March 2006), ETSI, 650 Route des Lucioles, F-06921 Sophia Antipolis Cedex, FRANCE, http://webapp.etsi.org/workprogram/Report_WorkItem.asp?WKI_ID=21353, 2006

[**XML-ID**] Jonathan Marsh, Daniel Veillard, Norman Walsh, *xml:id Version 1.0*, http://www.w3.org/TR/2005/REC-xml-id-20050909/, W3C, 2005.

[**xml-names**] Tim Bray et al., *Namespaces in XML 1.0 (Second Edition)*, http://www.w3.org/TR/2006/REC-xml-names-20060816, W3C, 2006.

[**XML1.0**] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, François Yergeau , *Extensible Markup Language (XML) 1.0 (Fourth Edition)*, http://www.w3.org/TR/2006/REC-xml-20060816/, W3C, 2004.

[**xmldsig-core**] Donald Eastlake, Joseph Reagle, David Solo, Frederick Hirsch, Thomas Roessler, *XML Signature Syntax and Processing (Second Edition)*, http://www.w3.org/TR/2008/REC-xmldsig-core-20080610/, W3C, 2008.

[**xmlenc-core**] Donald Eastlake, Joseph Reagle, *XML Encryption Syntax and Processing*, http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/, W3C, 2002.

[**xmlschema-2**] Paul V. Biron, Ashok Malhotra, *XML Schema Part 2: Datatypes Second Edition*, http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/, W3C, 2004.

[**ZIP**] *PKWARE Inc. Zip APPNOTE Version 6.2.0*, available at http://www.pkware.com/support/application-note-archives, 2004

1.4 Non Normative References [no change]

[**XAdES**] XML Advanced Electronic Signatures (XAdES) (ETSI TS 101 903 v1.3.2 March 2006), ETSI, 650 Route des Lucioles, F-06921 Sophia Antipolis Cedex, FRANCE, http://webapp.etsi.org/workprogram/Report_WorkItem.asp?WKI_ID=21353, 2006

1.5 Namespaces [no change]

2 Packages, Package Consumers and Package Producers [no changes]

3 Packages

3.1 General [no change]

3.2 Manifest [no change]

3.3 MIME Media Type [no change]

3.4 Encryption

3.4.1 General

OpenDocument packages may be encrypted by encrypting some or all files within the package. The encryption process takes place in the following stages:

- TheA single start key is generated and used for all of the keys that will be derived.
- The derived key is generated based on the start key.
- The files are encrypted based on the derived key and the encryption algorithm.

Package consumers and producers that support encryption shall support the digest and encryption algorithms defined in 3.4.2. They may support additional algorithms. The information regarding the algorithms that were used to encrypt a file and required parameters are contained in the manifest. The manifest shall not be encrypted.

Each file entry that is encrypted shall be compressed with the “deflate” algorithm before being encrypted. Encrypted file entries shall be flagged as 'STORED' rather than 'DEFLATED' in the Zip file's central directory. The size of the encrypted file should replace the real size value in the file entry's central directory records, its local file header and the data descriptor, if any. The original uncompressed, unencrypted size shall be contained in the `manifest:size` Error: Reference source not found attribute of the `<manifest:file-entry>` 4.3 element for the file entry.

The encrypted form can be of greater size than the DEFLATED file used as the plaintext (e.g., because of padding of plaintext, inclusion of additional information, and other characteristics of the encryption technique. The encryption method shall be such that the exact size and value of the plaintext DEFLATED file is exactly recovered by the corresponding decryption process.

3.4.2 Encryption Process using default algorithms

The three stages of the encryption process proceed as follows, using the legacy algorithms to illustrate each stage.The encryption process for file entries using the default digest and encryption algorithms has three steps:

1.The start key is generated: The byte sequence representing the password in UTF-8 is used to generate a 20-byte SHA1 digest (see [RFC3174]).

2. For each file to be encrypted, a separate derived key is generated from the start keyThe derived key is generated from the start key. The PBKDF2 algorithm based on the HMAC-SHA-1 function (see [RFC2898]) is used for the key derivation. For each file, a 16-byte salt is generated

by a random generator. The salt is used together with the start key to derive a unique 128-bit key for each file. The default iteration count for the algorithm is 1024.

3.The files are encrypted: The random number generator is used to generate the 8-byte initialization vector for the algorithm. The derived key is used together with the initialization vector to encrypt the file using the Blowfish algorithm in cipher feedback (8-bit CFB) mode (see [Blowfish]).

4 Manifest File

4.1 Introduction

The format of the manifest file is defined by the OpenDocument manifest Relax-NG [RNG] schema. See appendix A. This chapter describes the semantics of the elements and attributes defined by this schema.

4.2 <manifest:manifest> [no change]

4.3 <manifest:file-entry> [no change]

4.4 <manifest:encryption-data>

The `<manifest:encryption-data>` element contains information required to decrypt a file entry.

element-manifest:encryption-data

The `<manifest:encryption-data>` element is usable with the following element: `<manifest:file-entry>` 4.3.

The `<manifest:encryption-data>` element has the following attributes: `manifest:checksum` 4.8.2 and `manifest:checksum-type` 4.8.3.

The `<manifest:encryption-data>` element has the following child elements: `<manifest:algorithm>` 4.5, `<manifest:key-derivation>` 4.7 and `<manifest:start-key-generation>` 4.6.

4.5 <manifest:algorithm>

The `<manifest:algorithm>` element specifies the algorithm used to encrypt data.

Depending on the algorithm specified by the `manifest:algorithm-name` attribute 4.8.1 , the `<manifest:algorithm>` element may have further child elements.

WhenIf the `manifest:algorithm-name` attribute value matches one of those defined in section §3.2 of [xmlenc-core], does not have the value `Blowfish CFB` or `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#blowfish`, the `<manifest:algorithm>` element shall not havecontain only child elements except thosethat are permitted as child elements of anthe [xmlenc-core] `<EncryptionMethod>` element, as defined in §3.2 of [xmlenc-core], whose `Algorithm` attribute value is the same as the <manifest:algorithm>has the value of the `manifest:algorithm-name` attribute value.

WhenIf the value of the `manifest:algorithm-name` attribute is `Blowfish CFB` or `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#blowfish`identifies the legacy Blowfish algorithm, the `<manifest:algorithm>` element shall not have child elements.shall be an empty element.

element-manifest:algorithm

The <manifest:algorithm> element is usable with the following element: <manifest:encryption-data> 4.4.

The <manifest:algorithm> element has the following attributes: manifest:algorithm-name 4.8.1 and manifest:initialisation-vector 4.8.5.

The <manifest:algorithm> element has no child elements.

4.6 <manifest:start-key-generation>

The optional <manifest:start-key-generation> element specifies how the encryption start key iswas calculated from the user specified password. The password shall be provided as a sequence of bytes in UTF-8 encoding to the start key generation algorithm.

When a <manifest:start-key-generation> element is absent as a child of a <manifest:encryption-data> element, interpretation is the same as if the element is present with default attribute values.

element-manifest:start-key-generation

The <manifest:start-key-generation> element is usable with the following element: <manifest:encryption-data> 4.4.

The <manifest:start-key-generation> element has the following attributes: manifest:key-size 4.8.7 and manifest:start-key-generation-name 4.8.6.

The <manifest:start-key-generation> element has no child elements.

4.7 <manifest:key-derivation>

The <manifest:key-derivation> element specifies how the encryption key was calculated from the encryption start key.

element-manifest:key-derivation

The <manifest:key-derivation> element is usable with the following element: <manifest:encryption-data> 4.4.

The <manifest:key-derivation> element has the following attributes: manifest:iteration-count 4.8.8, manifest:key-derivation-name 4.8.9, manifest:key-size 4.8.7 and manifest:salt 4.8.12.

The <manifest:key-derivation> element has no child elements.

4.8 Manifest Attributes

4.8.1 manifest:algorithm-name

The manifest:algorithm-name attribute identifiesspecifies the name of the algorithm and mode used to encrypt a file entry, and also specifies in which mode this algorithm was used.

The defined values for the manifest:algorithm-name attribute are:

- An IRI listed in §5.2 of [xmlenc-core]: The algorithm specified in §5.2 of [xmlenc-core] for this IRI
- `Blowfish CFB`: The Blowfish algorithm in 8-bit CFB mode. See [Blowfish].
- `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#blowfish`: The same algorithm as identified by `Blowfish CFB`.
- , or of [xmlenc-core] for this IRI or §5.3 of [xmlenc-core]: The algorithm specified in §5.2or §5.3An IRI listed in §5.2
- The IRI of an alternative algorithm as specified in §5.1 of [xmlenc-core]. Alternative algorithms may be specified by extended conforming packages only. They shall not be specified by conforming packages.

Package producers and consumers shall support the AES-128 CBC algorithm and mode identified by value `http://www.w3.org/2001/04/xmlenc#aes128-cbc`. The legacy Blowfish algorithm need not be supported. [Note: Support of the Blowfish algorithm by package consumers provides compatibility with existing applications and documents conforming to earlier versions of this specification.] that support encryption shall support the value `Blowfish CFB`. Package consumers that support encryption shall support the values `Blowfish` and `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#blowfish`.

attribute-manifest:algorithm-name

The `manifest:algorithm-name` attribute is usable with the following element: `<manifest:algorithm>` 4.5.

The values of the `manifest:algorithm-name` attribute are `Blowfish CFB` or a value of type `anyURI` 7.2.

4.8.2 manifest:checksum

The `manifest:checksum` attribute specifies a digest in BASE64 encoding that can be used to detect password correctness as specified by a `manifest:checksum-type` attribute 4.8.3 .

attribute-manifest:checksum

The `manifest:checksum` attribute is usable with the following element: `<manifest:encryption-data>` 4.4.

The `manifest:checksum` attribute has the data type `base64Binary` 7.2.

4.8.3 manifest:checksum-type

The `manifest:checksum-type` attribute specifies the name of a digest algorithm that can be used to check password correctness. The digest is build from the compressed unencrypted file.

The defined values for the `manifest:checksum-type` attribute are:

- `SHA1/1K`: SHA1 algorithm (see [RFC3174]) applied to first 1024 bytes of the compressed unencrypted file.
- `SHA1`: The same as `http://www.w3.org/2000/09/xmldsig#sha1`.
- `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#sha1-1k`: The same as `SHA1/1K`.

- `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#sha2561-1256k`: SHA256 algorithm (see [RFC3174]) applied to first 1024 bytes of the compressed unencrypted file.
- An IRI listed in §5.7 of [xmlenc-core]: The algorithm specified in §5.7 of [xmlenc-core] for this IRI.
- The IRI of an alternative algorithm as specified in §5.1 of [xmlenc-core]. Alternative algorithms may be specified by extended conforming packages only. They shall not be specified by conforming packages.

Package producers that support encryption should use the `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#sha2561-1256k` algorithm, Package consumers that support encryption shall support the values `SHA1/1K`, `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#sha1`-1k and `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#sha2561-1256k`.

attribute-manifest:checksum-type

The `manifest:checksum-type` attribute is usable with the following element: `<manifest:encryption-data>` 4.4.

The values of the `manifest:checksum-type` attribute are `SHA1/1K` or a value of type `anyURI` 7.2.

4.8.4 manifest:full-path [no change]

4.8.5 manifest:initialisation-vector

The optional `manifest:initialisation-vector` attribute value providesspecifies the byte-sequence for theused as an initialization vector to aused by the encryption algorithm when dey of a required initialization vector is not specified as part of the encryption algorithm definition. The initialization vector is a BASE64 encoded binary sequence.

The format and length of the initialization vector, in bytes, shall be as required by the encryption algorithm specification.

attribute-manifest:initialisation-vector

The `manifest:initialisation-vector` attribute is usable with the following element: `<manifest:algorithm>` 4.5.

The `manifest:initialisation-vector` attribute has the data type `base64Binary` 7.2.

4.8.6 manifest:start-key-generation-name

The `manifest:start-key-generation-name` attribute specifies the algorithm used to generate a start key from the user password.

The defined values for the `manifest:start-key-generation-name` attribute are:

- `SHA1`: The SHA1 algorithm (see [RFC3174]).
- An IRI listed in §5.7 of [xmlenc-core]: The algorithm specified in §5.7 of [xmlenc-core] for this IRI.

- The IRI of an alternative algorithm as specified in §5.1 of [xmlenc-core] Alternative algorithms may be specified by extended conforming packages only. They shall not be specified by conforming packages.

The default value for this attribute is `SHA1`.

Package producers that support encryption should use the `http://www.w3.org/2000/09/xmldsig#sha256` algorithm.. Package consumers that support encryption shall support the values `SHA1`, and `http://www.w3.org/2000/09/xmldsig#sha1` and `http://www.w3.org/2000/09/xmldsig#sha256`.

attribute-manifest:start-key-generation-name

The `manifest:start-key-generation-name` attribute is usable with the following element: `<manifest:start-key-generation>` 4.6.

The values of the `manifest:start-key-generation-name` attribute are `SHA1` or a value of type `anyURI` 7.2.

The manifest:start-key-generation-name attribute has the value `SHA1` or a value of data type `anyURI`.

4.8.7 manifest:key-size

The `manifest:key-size` attribute specifies the length of a key.

For a `<manifest:start-key-generation>` element, the default value for this attribute is 20 [Note: the value used will need to be compatible with the result obtained from the start-key-generation algorithm and the input requirements of the key derivation algorithm.]

For a `<manifest:key-derivation>` element, the default value for this attribute is 16. [Note: the value used will need to be one obtainable from the key-derivation algorithm and acceptable for the encryption algorithm being used.]

attribute-manifest:key-size

The `manifest:key-size` attribute is usable with the following elements: `<manifest:key-derivation>` 4.7 and `<manifest:start-key-generation>` 4.6.

The `manifest:key-size` attribute has the data type `nonNegativeInteger` 7.2.

4.8.8 manifest:iteration-count

The `manifest:iteration-count` attribute specifies the number of iterations used by the key derivation algorithm to derive a key.

attribute-manifest:iteration-count

The `manifest:iteration-count` attribute is usable with the following element: `<manifest:key-derivation>` 4.7.

The `manifest:iteration-count` attribute has the data type `nonNegativeInteger` 7.2.

4.8.9 manifest:key-derivation-name

The `manifest:key-derivation-name` attribute specifies the name of the algorithm used to derive a name.password-based key-derivation algorithm used to derive a cryptographic key for use in encryption and decryption of the file.

The defined values for the `manifest:key-derivation-name` attribute are:

- `PBKDF2`: The PBKDF2 key derivation method with HMAC-SHA-1 Pseudo-Random Function (PRF). See [RFC2898] section 5.2 and B.1.1.
- `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#pbkdf2`: The same algorithms as identified by `PBKDF2`.
- An IRI listed in §5.7 of [xmlenc-core]: The algorithm specified in §5.7 of [xmlenc-core] for this IRI.
- The IRI of an implementation-definedalternative algorithm specified in §5.1 of [xmlenc-core] as. Alternative algorithms may be specified by extended conforming packages only. They shall not be specified by conforming packages.

Package producers that support encryption shall support the value `PBKDF2`. Package consumers that support encryption shall support the values `PBKDF2` and `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#pbkdf2`.

If the value of this attribute is `PBKDF2` or `urn:oasis:names:tc:opendocument:xmlns:manifest:1.0#pbkdf2` the `<manifest:encryption-data>` 4.4 shall contain a `<manifest:start-key-generation>` 4.6 child element that specifies the start key for the PBKDF2 algorithm.

attribute-manifest:key-derivation-name

The `manifest:key-derivation-name` attribute is usable with the following element: `<manifest:key-derivation>` 4.7.

The values of the `manifest:key-derivation-name` attribute are `PBKDF2` or a value of type `anyURI` 7.2.

4.8.10 manifest:media-type [no change]

4.8.11 manifest:preferred-view-mode [no change]

4.8.12 manifest:salt

The `manifest:salt` attribute carries the value of a cryptographically-random binary value designed to mitigate certain cryptographic attacks on the password and the encrypted file. There is no maximum length to the salt. See [RFC2898] for further considerations in the use of salts with key-derivation and other cryptographic functions. The salt is encoded in the attribute value as base64Binary. Specifies the sequence used as the 'salt' by a key derivation algorithm. The salt is a BASE64 encoded binary sequence.

attribute-manifest:salt

The `manifest:salt` attribute is usable with the following element: `<manifest:key-derivation>` 4.7.

The `manifest:salt` attribute has the data type `base64Binary` 7.2

5 Digital Signatures File [no change]

6 Metadata Manifest Files [no change]

7 Datatypes

7.1 Introduction

The values of attributes and elements are often described as having datatypes. These datatypes either are datatypes defined within [xmlschema-2], or are defined by this specification. Datatypes for which no [xmlschema-2] datatype exists are expressed in the schema by [xmlschema-2] datatypes. Some of these datatypes have additional constraints.

7.2 W3C Schema Datatypes

The following [xmlschema-2] datatypes are used in this specification:

- anyURI
- base64Binary
- nonNegativeInteger
- string

datatype-anyURI

datatype-base64Binary

datatype-nonNegativeInteger

datatype-string

7.3 Other Datatypes

7.3.1 namespacedToken

A namespaced token is an [xmlschema-2] QName that matches the definition of PrefixedName in §4 of [xml-names].

datatype-namespacedToken

Appendix A.Schemas [no changes]

Appendix B. [no changes]

Appendix C. [no changes]

Appendix D.Changes From “Open Document Format for Office Applications (OpenDocument) v1.1” (Non Normative)

The OpenDocument specification has been divided into three parts and has been restructured.

This appendix describes changes that are related to part 3 of this specification.

The following is a list of major features that have been added. For minor features please see the lists of new and changed elements and attributes.

- Digital Signatures Error: Reference source not found
- RDF based metadata Error: Reference source not found
- Support for additional encryption algorithms 3.4

The following element is new for manifest files:

- `<manifest:start-key-generation>` 4.6

The following attributes are new for manifest files:

- `manifest:key-size` 4.8.7
- `manifest:preferred-view-mode` 4.8.11
- `manifest:start-key-generation-name` 4.8.6
- `manifest:version` Error: Reference source not found

The value types of the following attributes changed:

- `manifest:algorithm-name` 4.8.1 of `<manifest:algorithm>` 4.5
- `manifest:checksum-type` 4.8.3 of `<manifest:encryption-data>` 4.4
- `manifest:key-derivation-name` 4.8.9 of `<manifest:key-derivation>` 4.7