[MS-OXOSMIME]:

S/MIME Email Object Algorithm

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks**. The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Preliminary Documentation. This Open Specification provides documentation for past and current releases and/or for the pre-release version of this technology. This Open Specification is final documentation for past or current releases as specifically noted in the document, as applicable; it is preliminary documentation for the pre-release versions. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. As the documentation may change between this preliminary version and the final version of this technology, there are risks in relying on preliminary documentation. To the extent that you incur additional

development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1		Initial Availability.
4/25/2008	0.2		Revised and updated property names and other technical content.
6/27/2008	1.0		Initial Release.
8/6/2008	1.01		Revised and edited technical content.
9/3/2008	1.02		Updated references.
12/3/2008	1.03		Minor editorial fixes.
3/4/2009	1.04		Revised and edited technical content.
4/10/2009	2.0		Updated technical content and applicable product releases.
7/15/2009	3.0	Major	Revised and edited for technical content.
11/4/2009	3.0.1	Editorial	Revised and edited the technical content.
2/10/2010	3.0.1	None	Version 3.0.1 release
5/5/2010	3.1	Minor	Updated the technical content.
8/4/2010	3.2	Minor	Clarified the meaning of the technical content.
11/3/2010	3.2	No change	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	3.2	No change	No changes to the meaning, language, or formatting of the technical content.
8/5/2011	4.0	Major	Significantly changed the technical content.
10/7/2011	4.1	Minor	Clarified the meaning of the technical content.
1/20/2012	5.0	Major	Significantly changed the technical content.
4/27/2012	6.0	Major	Significantly changed the technical content.
7/16/2012	6.0	No Change	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	7.0	Major	Significantly changed the technical content.
2/11/2013	7.1	Minor	Clarified the meaning of the technical content.
7/26/2013	7.1	No Change	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	7.1	No Change	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	7.1	No Change	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	8.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
7/31/2014	8.0	No Change	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	8.0	No Change	No changes to the meaning, language, or formatting of the technical content.
3/16/2015	9.0	Major	Significantly changed the technical content.
5/26/2015	9.0	No Change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

The S/MIME Email Object Algorithm provides the details of the internal format of a message and describes the mapping between the internal format and the Internet e-mail message format for two specific classes of Internet e-mail messages: opaque-signed or **encrypted messages**, and **clear-signed messages**, as described in [RFC5751].

When the server receives an Internet e-mail message, it maps the message to an internal format known as the **Message object** format. Similarly, when the client submits an e-mail message by way of the server, the server maps the message from its internal format to the Internet e-mail message format for sending. Also, in cases where protocols supported by the server allow saving or reading e-mail messages in the Internet e-mail message format, a similar mapping is required to and/or from the internal format. For more information about the mapping between the internal format and the Internet e-mail message format for other classes of messages, see [MS-OXCMAIL].

Section 2 of this specification is normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [RFC2119]. Section 1.6 is also normative but does not contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are specific to this document:

ASCII: The American Standard Code for Information Interchange (ASCII) is an 8-bit character-encoding scheme based on the English alphabet. ASCII codes represent text in computers, communications equipment, and other devices that work with text. ASCII refers to a single 8-bit ASCII character or an array of 8-bit ASCII characters with the high bit of each character set to zero.

Attachment object: A set of properties that represents a file, **Message object**, or structured storage that is attached to a Message object and is visible through the attachments table for a Message object.

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable **ASCII** characters, as described in [RFC4648].

body part: A part of an Internet message, as described in [RFC2045].

clear-signed message: An Internet email message that is in the format described by [RFC1847] and is identified with the media type "multipart/signed", or the **Message object** representing such a message. An important class of clear-signed message, based on a "multipart/signed" format, is the S/MIME clear-signed message, as described in [RFC5751] and [RFC3852].

encrypted message: An Internet email message that is in the format described by [RFC5751] and uses the EnvelopedData CMS content type described in [RFC3852], or the **Message object** that represents such a message.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [RFC4122] or [C706] must be used for generating the **GUID**. See also universally unique identifier (UUID).

header field: A component of a Session Initiation Protocol (SIP) message header, as described in [RFC3261].

media type: A value that is specified in a Content-Type Header field, as described in [RFC2045].

message body: The main message text of an email message. A few properties of a **Message object** represent its message body, with one property containing the text itself and others defining its code page and its relationship to alternative body formats.

message class: A property that loosely defines the type of a message, contact, or other Personal Information Manager (PIM) object in a mailbox.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME body: The content of a **MIME** entity, which follows the header of the MIME entity to which they both belong.

MIME entity: An entity that is as described in [RFC2045], [RFC2046], and [RFC2047].

MIME entity header: A type of header that is as described by [RFC2045].

MIME message: A message that is as described in [RFC2045], [RFC2046], and [RFC2047].

Multipurpose Internet Mail Extensions (MIME): A set of extensions that redefines and expands support for various types of content in email messages, as described in [RFC2045], [RFC2046], and [RFC2047].

named property: A property that is identified by both a GUID and either a string name or a 32-bit identifier.

opaque-signed message: An Internet email message that is in the format described by [RFC5751] and uses the SignedData CMS content type described in [RFC3852], or the **Message object** that represents such a message.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [RFC5751].

top-level message: A message that is not included in another message as an Embedded Message object. Top-level messages are messaging objects.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[MS-OXCMAIL] Microsoft Corporation, "RFC 2822 and MIME to Email Object Conversion Algorithm".

[MS-OXCMSG] Microsoft Corporation, "Message and Attachment Object Protocol".

[MS-OXOMSG] Microsoft Corporation, "Email Object Protocol".

[MS-OXPROPS] Microsoft Corporation, "Exchange Server Protocols Master Property List".

[RFC1847] Galvin, J., Murphy, S., Crocker, S., and Freed, N., "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, October 1995, http://www.rfc-editor.org/rfc/rfc1847.txt

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, http://www.rfc-editor.org/rfc/rfc2045.txt

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2183] Troost, R., Dorner, S., and Moore, K., Ed., "Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field", RFC 2183, August 1997, http://www.rfc-editor.org/rfc/rfc2183.txt

[RFC3852] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 3852, July 2004, http://www.ietf.org/rfc/rfc3852.txt

[RFC5751] Ramsdell, B., and Turner, S., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification", RFC 5751, January 2010, http://www.rfc-editor.org/rfc/rfc5751.txt

1.2.2 Informative References

[MS-OXPROTO] Microsoft Corporation, "Exchange Server Protocols System Overview".

[RFC2046] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996, http://www.rfc-editor.org/rfc/rfc2046.txt

[RFC2047] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996, http://ietf.org/rfc/rfc2047.txt

[RFC2048] Freed, N., Klensin, J., and Postel, J., "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", BCP 13, RFC 2048, November 1996, http://www.rfc-editor.org/rfc/rfc2048.txt

[RFC2049] Freed, N., and Borenstein N., "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples", RFC 2049, November 1996, http://www.rfc-editor.org/rfc/rfc2049.txt

[RFC2311] Dusse, S., Hoffman, P., Ramsdell, B., et al., "S/MIME Version 2 Message Specification", RFC 2311, March 1998, http://www.rfc-editor.org/rfc/rfc2311.txt

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, http://www.ietf.org/rfc/rfc2822.txt

1.3 Overview

This algorithm specifies the conversion between the Internet e-mail message format and the Message object format for clear-signed, opaque-signed, and encrypted messages. This algorithm extends the [RFC2822] and MIME to Email Object Conversion Algorithm as described in [MS-OXCMAIL], which

describes the general conversion between the Internet e-mail message format and the Message object format. The Internet e-mail message format is described in [RFC2822], [RFC2045], [RFC2046], [RFC2047], [RFC2048], [RFC2049], [RFC1847], and [RFC5751], and the Message object format is described in [MS-OXCMSG].

A conversion from the Internet e-mail message format to the Message object format is performed by the protocol role when it receives a message in the Internet e-mail message format. Likewise, a conversion from the Message object format to the Internet e-mail message format is performed by the protocol role when it sends a message. In cases where the protocol role supports protocols that allow for saving or reading messages in the Internet e-mail message format, a similar mapping is required to and/or from the internal format.

The conversion process maps **MIME entities** to **Attachment objects** or the **message body**, and it maps message **header fields** and **MIME entity header** fields to properties of the Message object or Attachment object. For details about the entire conversion process, see [MS-OXCMAIL] section 2.

Ordinarily, when an Internet e-mail message or **MIME message** is mapped to a Message object, it is completely deconstructed into a form suitable for direct consumption by a wire protocol, and it can be mapped to a typical client's message presentation. This manner of message deconstruction is not feasible for **S/MIME** messages for the following two reasons:

1. Encrypted message content and the entire message structure are not accessible without a proper decryption key, which is typically not available at the time of delivery.

2. Signed message content has to be preserved in its entirety, in the exact form in which it was signed, in order for the message signature (as described in [RFC5751]) to be verifiable at a later date.

The preceding points impose restrictions on how the server and the client map an S/MIME message to a Message object; the algorithm described in [MS-OXCMAIL] cannot be used without modifications.

A set of mapping conventions exists to resolve this problem and to enable the handling of S/MIME messages as Message objects. The mapping conventions are as follows:

- Unprotected **top-level message** header fields and MIME entity header fields are mapped to properties of a Message object or Attachment object in accordance with the algorithm described in [MS-OXCMAIL].

- The Message object is identified as an S/MIME message by having its message class (**PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3)) set to one of the reserved values described in sections 2.1.3.1 and 2.1.3.2.

- The entire protected content of the S/MIME message is mapped to a single Attachment object of a corresponding Message object.

This algorithm does not distinguish **opaque-signed messages** from encrypted messages; they are both converted in the same manner.

1.4 Relationship to Protocols and Other Algorithms

This algorithm defines a special case of mapping between the Message object format and e-mail messages in the following Internet formats: [RFC2822], [RFC2045], [RFC2046], [RFC2047], [RFC2048], [RFC2049], [RFC1847], [RFC5751], and [RFC3852]. General mapping is described in [MS-OXCMAIL].

For conceptual background information and overviews of the relationships and interactions between this and other protocols, see [MS-OXPROTO].

1.5 Applicability Statement

This algorithm can be used by any protocol role that is required to represent S/MIME messages by using a Message object format. It can also be used by any protocol role that is required to send or receive S/MIME messages by using a server that implements the Message object format.

The algorithm is limited to top-level clear-signed or S/MIME wrapping only; a message classified as a clear-signed message, an opaque-signed message, or an encrypted message can contain other nested S/MIME wrapping layers.

This algorithm specifies the interpretation and rendering of clear-signed messages, opaque-signed messages, and encrypted messages based on the assumption that the client or server that requires the interpretation or rendering of such messages can parse and interpret the corresponding Internet e-mail message format as defined in the following protocols: [RFC2822], [RFC2045], [RFC2046], [RFC2047], [RFC2048], [RFC2049], [RFC1847], [RFC5751], and [RFC3852].

1.6 Standards Assignments

None.

2 Algorithm Details

2.1 S/MIME E-Mail Object Conversion Algorithm Details

This section specifies the algorithm to convert clear-signed messages that use the "multipart/signed" **MIME** format, and messages signed or encrypted according to the S/MIME standard, to an internal Message object format.

2.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this algorithm. The described organization is provided to facilitate the explanation of how the algorithm behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

This protocol includes the following abstract data model (ADM) types and elements:

Mailbox, as specified in [MS-OXCMSG] section 3.1.1.2.

MessageObject, as specified in [MS-OXCMSG] section 3.1.1.3.

2.1.2 Initialization

None.

2.1.3 Processing Rules

The processing rules for converting clear-signed messages, opaque-signed messages, or encrypted messages to an internal Message object format are specified in sections 2.1.3.1 through 2.1.3.2.1.3.

2.1.3.1 Clear-Signed Message

A clear-signed message in the Internet e-mail message format is a message in which the message's MIME entity has the **media type** "multipart/signed", as specified in [RFC1847]. Such a MIME entity has two **body parts**: the first part represents signed message content; the second part contains a message signature, as specified in [RFC5751].

A clear-signed message in Internet e-mail message format is mapped to a Message object with the following structure:

- The **message class**, as specified by the **PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3) SHOULD<1> <2> be set to "IPM.Note.SMIME.MultipartSigned"; otherwise, it MAY<3> be set to "IPM.Note.SMIME".
- The message body SHOULD be set by promoting a primary message body MIME entity to appropriate properties of a Message object, as specified in [MS-OXCMSG] section 2.2, by adhering to the following requirements:
    - Consider the first body part of a multipart/signed message MIME entity as a complete Internet e-mail message.
    - Apply the heuristics specified in [MS-OXCMAIL] to identify a nested MIME entity as a message body and promote its content according to [MS-OXCMAIL] section 2.1.

- Message object properties other than the message class (**PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3)) or the message body SHOULD be set as specified in [MS-OXCMAIL] section 2.1 and [MS-OXOMSG] section 2.2.
- The Message object MUST contain exactly one Attachment object.
    - Attachment object content, stored in the **PidTagAttachDataBinary** property ([MS-OXPROPS] section 2.580), MUST be set as the entire outer content of a multipart/signed message MIME entity, including a **Content-Type** header field, as specified in [RFC2045], with the value "multipart/signed" and any original parameters. All other message entity header fields SHOULD be excluded. It is important to preserve the entire original outer content of the first body part within a multipart/signed MIME entity without modification, as it is protected by a message signature in its original form, and any modification will invalidate the message signature. Note that all message header fields that are excluded are normally processed to populate Message object properties, as specified in [MS-OXCMAIL] section 2.1.
    - Other Attachment object properties are to be set as follows:
        - The **PidTagAttachMethod** property ([MS-OXPROPS] section 2.592) MUST be set to a value of 0x00000001 (binary data attachment, as specified in [MS-OXCMSG] section 2.2.2.9).
        - The **PidTagAttachMimeTag** property ([MS-OXPROPS] section 2.593) MUST be set to a value of "multipart/signed".
        - The **PidTagAttachFilename** property ([MS-OXPROPS] section 2.584) either is set to a value of "SMIME.txt" or "SMIME.p7m" or is not set.<4>
        - The **PidTagAttachLongFilename** property ([MS-OXPROPS] section 2.586) SHOULD<5> be set to a value of "SMIME.p7m".
        - The **PidTagDisplayName** property ([MS-OXPROPS] section 2.667) SHOULD<6> be set to a value of "SMIME.p7m" by the server. The client does not set this property during conversion.

2.1.3.1.1 Converting from an Internet Format Message to a Message Object

2.1.3.1.1.1 Recognizing a Clear-Signed Message in Internet Format

The media type of the message MIME entity is the value of the last **Content-Type** header field, as specified in [RFC2045] section 5. If the media type of the message MIME entity is "multipart/signed", the message SHOULD be treated as a clear-signed message. Additional verification steps can be performed at this point. For example, a client or server could choose to verify that the multipart/signed MIME entity contains exactly two **MIME body** parts, as specified in [RFC1847].

2.1.3.1.1.2 Converting a Clear-Signed Message in Internet Format into a Message Object

To convert a clear-signed message from the Internet e-mail message format to the Message object format, perform the following steps:

1. From the message MIME entity, promote message header fields to Message object properties, as specified in [MS-OXCMAIL].
2. Create an Attachment object.
3. Set Attachment object properties, as specified in section 2.1.3.1.

4. Remove all header fields except the **Content-Type** header field, as specified in [RFC2045] section 5, from the message MIME entity.

5. Save the resulting MIME entity as content of the Attachment object created in step 2. For example, set the value of the **PidTagAttachDataBinary** property ([MS-OXPROPS] section 2.580) on the Attachment object.

2.1.3.1.1.3 Composing a New Message Object that Represents a Clear-Signed Message

To compose a new Message object that represents a clear-signed message, first compose a clear-signed message by using the Internet e-mail message format specified in [RFC1847], and then convert that message to a Message object, as specified in section 2.1.3.1.1.2. A client can use a different process as long as it results in the same Message object content.

2.1.3.1.2 Converting from a Message Object to an Internet Format Message

2.1.3.1.2.1 Reading and Interpreting a Message Object that Represents a Clear-Signed Message

For details about how to recognize a Message object that represents a clear-signed message, see section 2.1.3.1.2.2.

To read and interpret a clear-signed message, the Internet e-mail message format SHOULD be reconstructed from a Message object, as specified in section 2.1.3.1.2.3. The resulting clear-signed message in its Internet e-mail message format SHOULD be rendered or interpreted following the guidelines specified in [RFC1847], and possibly [RFC5751]. A client can use a different process, as long as it leads to the same rendering or interpretation.

2.1.3.1.2.2 Recognizing a Message Object that Represents a Clear-Signed Message

If a Message object has a message class (**PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3)) value of "IPM.Note.SMIME.MultipartSigned" and contains exactly one Attachment object, it SHOULD be treated as a clear-signed message. Additional verification steps can be performed to verify that the Attachment object is marked with the appropriate media type (for example, the **PidTagAttachMimeTag** property ([MS-OXPROPS] section 2.593) has a value of "multipart/signed") and represents a valid multipart/signed MIME entity as specified in [RFC1847]. If the message class value is not "IPM.Note.SMIME.MultipartSigned" but it ends with the suffix ".SMIME.MultipartSigned", the Message object MAY<7> <8> be treated as a clear-signed message.

If a Message object with a message class value of "IPM.Note.SMIME.MultipartSigned" does not have the structure specified in section 2.1.3.1, the behavior is undefined.

2.1.3.1.2.3 Reconstructing an Internet Format Message from a Clear-Signed Message Object

To reconstruct a message from the clear-signed Message object format to the Internet e-mail message format, perform the following steps:

1. Verify that the Message object contains exactly one Attachment object.

2. Read the value of the Attachment object's **PidTagAttachDataBinary** property ([MS-OXPROPS] section 2.580), and treat it as a MIME entity.

3. Remove all header fields except the last **Content-Type** header field, as specified in [RFC2045] section 5, from the MIME entity.

4. Add any message header fields resulting from promotion of Message object properties (as specified in [MS-OXCMAIL] section 2.1) to the MIME entity.

5. The resulting MIME entity is a clear-signed message in its Internet e-mail message format. A client or server can use a different approach, as long as it leads to an equivalent result.

2.1.3.2 Opaque-Signed and Encrypted S/MIME Message

An opaque-signed message or encrypted message in the Internet e-mail message format is identified as a MIME message that consists of exactly one MIME entity. The MIME entity usually has the media type "application/pkcs7-mime" or "application/x-pkcs7-mime". Note, however, that it can alternatively have the media type "application/octet-stream" if a file name, identified by the **Content-Type** header field, as specified in [RFC2045] section 5, or the **Content-Disposition** header field, as specified in [RFC2183], has the file extension ".p7m". The content of the MIME body is a Cryptographic Message Syntax (CMS) encapsulation of protected message content, together with all necessary cryptographic metadata. For more details about CMS, see [RFC3852]. For the purposes of this algorithm, the content is treated as opaque binary data. Message types specified in [RFC5751] other than opaque-signed messages or encrypted messages are not supported.

An opaque-signed message or an encrypted message in the Internet e-mail message format is mapped to a Message object with the following structure:

- The message class (**PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3)) SHOULD<9> <10> have a value of "IPM.Note.SMIME".
- The message body SHOULD NOT be set. Even for an opaque-signed message, for which a decryption key is not required to access message content, the message body SHOULD NOT be promoted to a Message object.
- Message object properties other than message class or message body SHOULD be set as specified in [MS-OXCMAIL] section 2.2.
- The Message object SHOULD have a **named property** with **GUID** = PS_INTERNET_HEADERS ({00020386-0000-0000-C000-000000000046}) and a string name "Content-Type" that contains the raw **ASCII** string value of a message MIME entity's **Content-Type** MIME header field, including any parameters of the header field.
- The message MUST contain exactly one Attachment object.
  - Attachment content, stored in the **PidTagAttachDataBinary** property ([MS-OXPROPS] section 2.580), MUST be set as the inner content of a message MIME entity. Any Content-Transfer-Encoding applied to a MIME entity body MUST be removed before storing MIME body content in an Attachment object.
  - Attachment object properties other than content SHOULD be set according to [MS-OXCMAIL] section 2.2, just as they would be if the MIME entity was a normal message attachment. In particular, the **PidTagAttachMimeTag** property ([MS-OXPROPS] section 2.593) MUST be set to match the media type of a message MIME entity.

2.1.3.2.1 Converting from an Internet Format Message to a Message Object

2.1.3.2.1.1 Recognizing an Opaque-Signed or Encrypted S/MIME Message in Internet Format

The media type of the message MIME entity is the value of the last **Content-Type** header field, as specified in [RFC2045] section 5. If the message MIME entity's media type is "application/pkcs7-mime" or "application/x-pkcs7-mime", the message SHOULD be treated as an opaque-signed message

or an encrypted message. Also, if the message MIME entity's media type is "application/octet-stream", and a file extension specified by the name parameter of a **Content-Type** header field or the filename parameter of a **Content-Disposition** header field, as specified in [RFC2183], ends with ".p7m" (case-insensitive), the message SHOULD be treated as an opaque-signed message or an encrypted message. Additional verification steps can be performed. For example, a client or server could choose to verify that MIME body content has valid syntax, as specified in [RFC5751].

2.1.3.2.1.2 Converting an Opaque-Signed or Encrypted S/MIME Message in Internet Format into a Message Object

To convert an opaque-signed message or an encrypted message in the Internet e-mail message format into a Message object, perform the following steps:

1. From the message MIME entity, promote message header fields to Message object properties, as specified in [MS-OXCMAIL] section 2.2.

2. Save the raw ASCII string value of the last **Content-Type** header field, as specified in [RFC2045] section 5, including any parameters of the header, as a Message object named property with GUID = PS_INTERNET_HEADERS ({00020386-0000-0000-C000-000000000046}) and the name "Content-Type".

3. Promote the message MIME entity as a new Attachment object, as specified in [MS-OXCMAIL] section 2.2.3.4, for a general conversion case.

2.1.3.2.1.3 Composing a New Message Object that Represents an Opaque-Signed or Encrypted S/MIME Message

To compose a new Message object that represents an S/MIME message, first compose an opaque-signed message or an encrypted message by using the Internet e-mail message format specified in [RFC5751], and then convert that message to a Message object, as specified in section 2.1.3.2.1.2. The client SHOULD generate media types using an "x-" prefix; for example, application/x-pkcs7-mime. For more information about the "x-" prefix, see [RFC2311] section C.1. A client can use a different process as long as it results in the same Message object structure and content.

2.1.3.2.2 Converting from a Message Object to an Internet Format Message

2.1.3.2.2.1 Reading and Interpreting a Message Object that Represents an Opaque-Signed or Encrypted S/MIME Message

For details about how to recognize a Message object that represents an opaque-signed message or an encrypted message, see section 2.1.3.2.2.2.

To read and interpret an **S/MIME** message, the Internet e-mail message format SHOULD be reconstructed from a Message object, as specified in section 2.1.3.2.2.3. The resulting S/MIME message in its Internet e-mail message format SHOULD be rendered or interpreted by following the guidelines specified in [RFC5751]. A client can use a different process as long as it leads to the same rendering or interpretation.

2.1.3.2.2.2 Recognizing a Message Object that Represents an Opaque-Signed or Encrypted S/MIME Message

If a Message object has the message class (**PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3)) value of "IPM.Note.SMIME" and contains exactly one Attachment object, it SHOULD be treated as an opaque-signed message or an encrypted message. Additional verification steps can be performed to verify that the Attachment object is marked with the appropriate media type (for example, the **PidTagAttachMimeTag** property ([MS-OXPROPS] section 2.593) is either

"application/pkcs7-mime" or "application/x-pkcs7-mime", or it is "application/octet-stream" and filename, as specified by the **PidTagAttachFilename** property ([MS-OXPROPS] section 2.584), and has a file extension ".p7m") and represents a valid encrypted or opaque-signed message, as specified in [RFC3852]. If the value of the message class is not "IPM.Note.SMIME", but ends with the suffix ".SMIME", then the Message object MAY<11> be treated as an opaque-signed message or an encrypted message.

The message class value "IPM.Note.SMIME" can be ambiguous.<12>

If a Message object has a message class value of "IPM.Note.SMIME" does not have the appropriate structure or content as specified in section 2.1.3.2, then the behavior is undefined.

2.1.3.2.2.3 Reconstructing an Internet Format Message from an Opaque-Signed or Encrypted S/MIME Message Object

To reconstruct a message in the Internet e-mail message format from an opaque-signed message object or an encrypted message object, perform the following steps:

1. Verify that the Message object contains exactly one Attachment object.
2. Create an empty MIME entity.
3. Add any message header fields that result from promotion of the Message object properties, as specified in [MS-OXCMAIL] section 2.1, to the MIME entity.
4. Add the **Content-Type** header field, as specified in [RFC2045] section 5, to the MIME entity:
   - If the Message object has a named property "Content-Type" with GUID = PS_INTERNET_HEADERS ({00020386-0000-0000-C000-000000000046}), construct the **Content-Type** header field by using the value of the named property, assuming that the value can contain unparsed MIME parameters.
   - Otherwise, construct the **Content-Type** header field by using a media type string obtained from the value of the Attachment object's **PidTagAttachMimeTag** property ([MS-OXPROPS] section 2.593). When doing this, add a *name* parameter with a value obtained from the **PidTagAttachFilename** property ([MS-OXPROPS] section 2.584) of the Attachment object.
5. Add a MIME body with a disposition value "attachment" to the MIME entity; add a single *file name* parameter with a value obtained from the **PidTagAttachFilename** property of the Attachment object, encoded if necessary.
6. Add the **Content-Transfer-Encoding** header field, as specified in [RFC2045] and with a value of "base64", to the MIME entity.
7. Read the value of the Attachment object's **PidTagAttachDataBinary** property ([MS-OXPROPS] section 2.580), and encode the result using **base64 encoding**. Add the result of the encoding as a body of the MIME entity.

The resulting MIME entity is an opaque-signed message or an encrypted message in its Internet e-mail message format. A client or server can use a different approach as long as it leads to an equivalent result.

3 Algorithm Examples

None.

4 Security

4.1 Security Considerations for Implementers

This algorithm does not have any security implications beyond those described in [RFC5751]. Furthermore, this algorithm treats S/MIME content as opaque binary data and does not deal with any sensitive material or data such as encryption keys. Although it is best for clients or servers that render, interpret, or compose S/MIME data to do so in a secure manner, that subject area is beyond the scope of this specification.

4.2 Index of Security Parameters

None.

5 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs.

- Microsoft Exchange Server 2003
- Microsoft Exchange Server 2007
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016 Preview
- Microsoft Office Outlook 2003
- Microsoft Office Outlook 2007
- Microsoft Outlook 2010
- Microsoft Outlook 2013
- Microsoft Outlook 2016 Preview

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 2.1.3.1: Exchange 2003, Office Outlook 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview identify any message that has a message class suffix of "SMIME.MultipartSigned" as a clear-signed message; however, this is not recommended for other clients or servers.

<2> Section 2.1.3.1: Exchange 2007, Exchange 2010, Exchange 2013, Exchange 2016 Preview Office Outlook 2007, Microsoft Outlook 2010, Outlook 2013, and Outlook 2016 Preview recognize messages that were signed and encrypted by Microsoft Office InfoPath 2007, Microsoft InfoPath 2010, or Microsoft InfoPath 2013, and for such messages, they use a dynamically determined message class that starts with the prefix "IPM.InfoPathForm" and ends with the suffix "SMIME" or "SMIME.MultipartSigned", as specified in [MS-OXCMAIL] section 2.1.3.2.1. Exchange 2007, Exchange 2010, Exchange 2013, and Exchange 2016 Preview recognize such message classes as identifying opaque-signed, encrypted, or clear-signed messages, despite the fact that, in general, they do not recognize other message classes ending with the suffix "SMIME" or "SMIME.MultipartSigned".

<3> Section 2.1.3.1: A value of "IPM.Note.SMIME" in Exchange 2003 is ambiguous in respect to defining message format. It is recommended that a client or server that is required to interoperate with Exchange 2003 disambiguate the "IPM.Note.SMIME" Message object either by analyzing the content of an attachment (for example, the value of the Attachment object **PidTagAttachDataBinary** property ([MS-OXPROPS] section 2.580)) or by inspecting the value of the Attachment object **PidTagAttachMimeTag** property ([MS-OXPROPS] section 2.593). If the value represents a valid multipart/signed MIME entity, it is recommended that the client or server identify the message as a clear-signed message and interpret the message according to section 2.1.3.1.

<4> Section 2.1.3.1: Exchange 2003 sets the **PidTagAttachFilename** property ([MS-OXPROPS] section 2.584) to a value of "SMIME.txt". Exchange 2007, Exchange 2010, Exchange 2013, and Exchange 2016 Preview set the **PidTagAttachFilename** property to a value of "SMIME.p7m". Office Outlook 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview do not set the **PidTagAttachFilename** property when converting a message from the Internet e-mail message format to the Message object format. Office Outlook 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview set the **PidTagAttachFilename** property to a value of "SMIME.p7m" when creating a new Message object.

<5> Section 2.1.3.1: Exchange 2003 sets the **PidTagAttachLongFilename** property ([MS-OXPROPS] section 2.586) to a value of "SMIME.txt". Office Outlook 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview do not set the **PidTagAttachLongFilename** property.

<6> Section 2.1.3.1: Exchange 2003 sets the **PidTagDisplayName** property ([MS-OXPROPS] section 2.667) to a value of "SMIME.txt". Office Outlook 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview do not set the **PidTagDisplayName** property.

<7> Section 2.1.3.1.2.2: Office Outlook 2003, Exchange 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview identify any message that has a message class suffix of "SMIME.MultipartSigned" as a clear-signed message. In general, however, this is not recommended for other clients or servers.

<8> Section 2.1.3.1.2.2: Exchange 2007, Exchange 2010, Exchange 2013, Exchange 2016 Preview, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview recognize messages that were signed and encrypted by Office InfoPath 2007, InfoPath 2010, or InfoPath 2013, and for such messages, they use a dynamically determined message class that starts with the prefix "IPM.InfoPathForm" and ends with the suffix "SMIME" or "SMIME.MultipartSigned", as specified in [MS-OXCMAIL] section 2.1.3.2.1. Exchange 2007, Exchange 2010, Exchange 2013, and Exchange 2016 Preview recognize such message classes as identifying opaque-signed, encrypted, or clear-signed messages, despite the fact that, in general, they do not recognize other message classes ending with the suffix "SMIME" or "SMIME.MultipartSigned".

<9> Section 2.1.3.2: Exchange 2003, Office Outlook 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview set the message class to "IPM.Note.Receipt.SMIME" when they identify an S/MIME message that contains a secure receipt, as indicated by the *smime-type* parameter with a value of "signed-receipt" on the **Content-Type** header field, as specified in [RFC2045] section 5. Exchange 2003, Office Outlook 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview identify any message that has the message class suffix "SMIME" as an opaque-signed or encrypted message; however, this is not recommended for other clients or servers.

<10> Section 2.1.3.2: Exchange 2007, Exchange 2010, Exchange 2013, Exchange 2016 Preview, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview recognize messages that were signed and encrypted by Office InfoPath 2007, InfoPath 2010, or InfoPath 2013, and for such messages, they use a dynamically determined message class that starts with the prefix "IPM.InfoPathForm" and ends with the suffix "SMIME" or "SMIME.MultipartSigned", as specified in [MS-OXCMAIL] section 2.1.3.2.1. Exchange 2007, Exchange 2010, Exchange 2013, and Exchange 2016 Preview recognize such message classes as identifying opaque-signed, encrypted, or clear-signed messages, despite the fact that, in general, they do not recognize other message classes having suffixes "SMIME" or "SMIME.MultipartSigned".

<11> Section 2.1.3.2.2.2: Exchange 2007, Exchange 2010, Exchange 2013, Exchange 2016 Preview, Office Outlook 2007, Outlook 2010, Outlook 2013, and Outlook 2016 Preview recognize messages that were signed and encrypted by Office InfoPath 2007, InfoPath 2010, or InfoPath 2013, and for such messages, they use a dynamically determined message class that starts with the prefix "IPM.InfoPathForm" and ends with the suffix "SMIME" or "SMIME.MultipartSigned", as specified in [MS-OXCMAIL] section 2.1.3.2.1. Exchange 2007, Exchange 2010, Exchange 2013, and Exchange

2016 Preview recognize such message classes as identifying opaque-signed, encrypted, or clear-signed messages, despite the fact that, in general, they do not recognize other message classes ending in the suffix "SMIME" or "SMIME.MultipartSigned".

<12> Section 2.1.3.2.2.2: In Exchange 2003, if a Message object has a message class value of "IPM.Note.SMIME", it is possible that the message represents a mislabeled clear-signed message with inner opaque-signed or encrypted content. This means that, in Exchange 2003, the message class "IPM.Note.SMIME" is ambiguous with respect to defining message format. It is recommended that a client or server that is required to interoperate with Exchange 2003 disambiguate the "IPM.Note.SMIME" Message object either by analyzing the content of an attachment (for example, the value of the Attachment object **PidTagAttachDataBinary** property ([MS-OXPROPS] section 2.580)) or by inspecting the value of the Attachment object **PidTagAttachMimeTag** property ([MS-OXPROPS] section 2.593). If the value represents a valid multipart/signed MIME entity, it is recommended that the client or server identify the message as a clear-signed message and interpret the message according to section 2.1.3.1.

6 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated**.

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
5 Appendix A: Product Behavior	Added Exchange 2016 and Outlook 2016 to the list of applicable products.	Y	Content update.

7 Index