[MS-OXCMAIL]: RFC2822 and MIME to E-mail Object Conversion Protocol Specification

Intellectual Property Rights Notice for Protocol Documentation

- **Copyrights**. This protocol 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 protocols, and may distribute portions of it in your implementations of the protocols or your documentation as necessary to properly document the implementation. This permission also applies to any documents that are referenced in the protocol documentation.

- **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 protocols. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, the protocols may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp/default.mspx). If you would prefer a written license, or if the protocols are not covered by the OSP, patent licenses are available by contacting protocol@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.

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.

Preliminary Documentation. This documentation is preliminary documentation for these protocols. Since the documentation may change between this preliminary version and the final version, 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.

Tools. This protocol documentation is intended for use in conjunction with publicly available standard specifications and networking programming art, and assumes that the reader is either familiar with the aforementioned material or has immediate access to it. A protocol specification does not require the use of Microsoft programming tools or programming environments in order for a Licensee to develop an implementation. Licensees who have access to Microsoft programming tools and environments are free to take advantage of them.

Revision Summary			
Author	Date	Version	Comments
Microsoft Corporation	April 4, 2008	0.1	Initial Availability.
Microsoft Corporation	April 25, 2008	0.2	Revised and updated property names and other technical content.

Table of Contents

1 Introduction

The RFC2822 and MIME to E-mail Object Conversion Protocol specifies what clients and servers do when they have data in one of these formats, but need it in the other. The process of converting message object data to MIME format is referred to as "MIME generation," while the reverse process is referred to as "MIME analysis."

1.1 Glossary

The following terms are defined in [MS-OXGLOS]:

body part
code page
character set
charset
Coordinated Universal Time (UTC)
header field
Internet Message Access Protocol – Version 4 (IMAP4)
JPG
message body
message class
MIME
MIME entity
Personal Information Manager (PIM)
Post Office Protocol - Version 3 (POP3)
property
Transport Neutral Encapsulation Format (TNEF)

The following terms are specific to this document:

addressee property group: A group of four related **properties** – display name, entry Id, e-mail address type, and e-mail address – that together specify one addressee on a message object.

header: A series of fields (name-value pairs) that supply structured data in an Internet e-mail message, as specified in [RFC2822], or a MIME entity. See also: **header field**, **MIME entity**.

Internet Mail Connector Encapsulated Address (IMCEA): A means of encapsulating an e-mail address that is not compliant with [RFC2821] within an e-mail address that is compliant with [RFC2821].

MIME analysis: The process of converting data from an Internet wire protocol to a format suitable for storage by Exchange or Outlook.

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

MIME generation: The process of converting data held by Exchange or Outlook to a format suitable for Internet-standard wire protocols.

MIME reader: An agent performing **MIME analysis**; it might be either a client or server.

MIME writer: An agent performing **MIME generation**; it might be either client or server.

Pure MIME message: A **MIME** representation of an e-mail message with no **Transport Neutral Encapsulation Format (TNEF) body part**.

TNEF message: A **MIME** representation of an e-mail message in which attachments and some message properties are carried in a **Transport Neutral Encapsulation Format (TNEF) body part**.

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

1.2 References

1.2.1 Normative References

[MS-DTYP] Microsoft Corporation, "Windows Data Types", March 2007, http://go.microsoft.com/fwlink/?LinkId=111558.

[MS-LCID] Microsoft Corporation, "Windows Language Code Identifier (LCID) Reference", March 2007, http://go.microsoft.com/fwlink/?LinkId=112265.

[MS-OXBBODY] Microsoft Corporation, "Best Body Retrieval Protocol Specification", April 2008.

[MS-OXCDATA] Microsoft Corporation, "Data Structures Protocol Specification", April 2008.

[MS-OXCICAL] Microsoft Corporation, "iCalendar to Appointment Object Conversion Protocol Specification", April 2008.

[MS-OXCMSG] Microsoft Corporation, "Message and Attachment Object Protocol Specification", April 2008.

[MS-OXCSPAM] Microsoft Corporation, "Spam Confidence Level, Allow and Block Lists Protocol Specification", April 2008.

[MS-OXGLOS] Microsoft Corporation, "Office Exchange Protocols Master Glossary", April 2008.

[MS-OXOCAL] Microsoft Corporation, "Appointment and Meeting Object Protocol Specification", April 2008.

[MS-OXOMSG] Microsoft Corporation, "E-mail Object Protocol Specification", April 2008.

[MS-OXOSMIME] Microsoft Corporation, "S/MIME E-mail Object Protocol Specification", April 2008.

[MS-OXPROPS] Microsoft Corporation, "Office Exchange Protocols Master Property List Specification", April 2008.

[MS-OXPROTO] Microsoft Corporation, "Office Exchange Protocols Overview", April 2008.

[MS-OXRTFEX] Microsoft Corporation, "Rich Text Format (RTF) Extensions Specification", April 2008.

[MS-OXTNEF] Microsoft Corporation, "Transport Neutral Encapsulation Format (TNEF) Protocol Specification", April 2008.

[MS-RTF] Microsoft Corporation, "Word 2007: Rich Text Format (RTF) Specification, Version 1.9", February 2007, http://go.microsoft.com/fwlink/?LinkId=112393.

[MS-WMF] Microsoft Corporation, "Windows Metafile Format Specification", June 2007, http://go.microsoft.com/fwlink/?LinkId=112205.

[RFC1740] Faltstrom, Patrick; Crocker, Dave; and Fair, Erik, "MIME Encapsulation of Macintosh files – MACMIME", RFC 1740, December 1994, http://www.ietf.org/rfc/rfc1740.txt.

[RFC1741] Faltstrom, Patrick; Crocker, Dave; and Fair, Erik, "MIME Content Type for BinHex Encoded Files", RFC 1741, December 1994, http://www.ietf.org/rfc/rfc1741.txt.

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

[RFC2046] Freed, N. and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996,http://www.ietf.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://www.ietf.org/rfc/rfc2047.txt.

[RFC2076] Palme, J., "Common Internet Message Headers", RFC 2076, February 1997, http://www.ietf.org/rfc/rfc2076.txt.

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

[RFC2231] Freed, N., Moore, K., "MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations", RFC 2231, November 1997, http://www.ietf.org/rfc/rfc2231.txt.

[RFC2387] Levinson, E., "The MIME Multipart/Related Content-type", RFC 2387, August 1998, http://www.ietf.org/rfc/rfc2387.txt.

[RFC2557] Palme, J., Hopmann, A., Shelness, N., "MIME Encapsulation of Aggregate Documents, such as HTML (MHTML)", RFC 2557, March 1999, http://www.ietf.org/rfc/rfc2557.txt.

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

[RFC3282] Alvestrand, H., "Content Language Headers", RFC 3282, May 2002, http://www.ietf.org/rfc/rfc3282.txt.

[RFC3464] Moore, K. and Vaudreuil, G., "An Extensible Message Format for Delivery Status Notifications", RFC 3464, January 2003, http://www.ietf.org/rfc/rfc3464.txt.

[RFC3798] Hansen, T. and Vaudreil, G., "Message Disposition Notification", RFC 3798, May 2004, http://www.ietf.org/rfc/rfc3798.txt.

[RFC3851] Ramsdell, B., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, July 2004, http://www.ietf.org/rfc/rfc3851.txt.

[RFC4646] Phillips, A. and Davis, M., "Tags for the Identification of Languages", RFC 4646, September 2006, http://www.ietf.org/rfc/rfc4646.txt.

1.2.2 Informative References

[MacBin] Apple Corporation, "Macintosh Binary Transfer Format ("MacBinary III") Standard Proposal", http://www.lazerware.com/macbinary/macbinary_iii.html.

[MS-OXPROTO] Microsoft Corporation, "Office Exchange Protocols Overview", April 2008.

[RFC1939] Myers, J., Rose, M., "Post Office Protocol – Version 3", RFC 1939, May 1996, http://www.ietf.org/rfc/rfc1939.txt.

[RFC3501] Crispin, M., "Internet Message Access Protocol – Version 4rev1", RFC 3501, March 2003, http://www.ietf.org/rfc/rfc3501.txt.

1.3 Structure Overview (Synopsis)

The representation of electronic mail, calendar items, and other Personal Information Manager (PIM) objects by message objects and their properties is outlined in the Outlook-Exchange Protocols System Overview [MS-OXPROTO] and detailed in the E-mail Object Protocol [MS-OXOMSG], the Appointment and Meeting Object Protocol [MS-OXOCAL] and related specifications.

In contrast, electronic mail, calendar items, and other PIM objects are represented as textual streams when sent over Internet protocols. The textual representation of these streams is commonly referred to as 2822 and/or MIME format as specified by "Internet Message Format" (see [RFC2822]), and "Multipurpose Internet Message Extensions (MIME)" (see [RFC2045] through [RFC2049]).

The RFC2822 and MIME to E-mail Object Conversion Protocol specifies how to convert between message objects and MIME formatted textual streams. The process of converting message object data to MIME formatted textual streams is referred to as "MIME generation," while the reverse process is referred to as "MIME analysis." Similarly, the agent performing MIME generation (which might be either a client or server) is referred to as a "MIME writer," and the agent performing MIME analysis is referred to as a "MIME reader."

1.3.1 Data Models

Message objects model e-mail messages and other PIM objects after a business memo: there is a single message body, with zero or more attachments and zero or more recipients. Each message object has a message class property indicating its type, and an arbitrary collection of properties. Attached messages allow for nesting of content.

MIME, in contrast, models e-mail messages as a nested set of MIME entities, each of which has header fields and a (possibly empty) body. No entity is distinguished as "the" message body. The content-type header field indicates the type of each body part; other header fields indicate whether a body part is intended as a message body or attachment. Recipients are modeled by e-mail addresses in certain header fields on the top-level body part. Multipart body parts allow for grouping and nesting of content, including attached messages.

At a high level, the parts of each data model correspond as in the following table.

Table 1 Data model components

MIME	Message object

E-mail address	Recipient
Header field	Property
Body part	Message body
Body part	Attachment

At the next level of detail, some problems become visible. Because the data models do not match exactly, each format becomes a more difficult to convert lower level items originating in the other format.

One of the challenges in mapping message object content to MIME arises from the need to generate human readable text. Many message object properties have data types, including binary blob, that do not lend themselves to representation as text. Two solutions are available for these problems:

1. Generate a "pure MIME" message, in which data that does not lend itself to representation in MIME is simply omitted from the MIME representation.
2. Generate a TNEF message, in which data that does not lend itself to representation in MIME is placed in a TNEF body part with content-type of "`application/ms-tnef`". Recipients, plain body text and top-level header fields are visible to all MIME clients.

Challenges in mapping MIME content to message objects include distinguishing message body from attachments; analyzing multipart structures that do not fit the message object data model; and mapping header fields or header field parameters that have no corresponding property.

Finally, each message object has a single charset (although nested message objects can have different charsets). MIME, on the other hand, permits the charset of each header field and body to be specified separately.

1.4 Relationship to Protocols and Other Structures

Data on the MIME side of the conversion is specified by [RFC2822], [RFC2045] through [RFC2049], and related specifications as listed in section 1.2.1 or as referenced from the specifications themselves. Data on the message object side of the conversion is specified by [MS-OXCMSG], [MS-OXOMSG], [MS-OXOCAL], and related specifications as listed in [MS-OXPROTO].

1.5 Applicability Statement

Conversion between MIME and message object format is performed in the context of several different protocols, for example:

- Clients and servers perform MIME generation for mail outbound to SMTP.
- Clients and servers perform MIME analysis for mail inbound from SMTP.
- Servers perform MIME generation for message objects being downloaded via POP3 or IMAP4. Clients perform MIME generation for messages being uploaded via IMAP4.
- Servers perform MIME analysis for message objects being uploaded via IMAP4.Clients perform MIME analysis for message objects being downloaded via POP3 or IMAP4.

1.6 Versioning and Localization

This document covers localization issues in the following areas:

- Localization: Localization dependent content is specified in Sections 2.1.3 and 2.2.3.

Localization is supported by marking messages with locale and character set information.

1.7 Vendor-Extensible Fields

[RFC2045] and related RFCs define extensibility mechanisms for MIME header fields and content types.

[MS-OXPROPS] defines extensibility mechanisms for message object properties and message classes.

2 Structures

The bulk of this specification is divided into two symmetrical parts. section 2.1 specifies how clients SHOULD set message object properties in order to produce desired MIME data, and section 2.2 specifies how clients SHOULD create MIME in order to produce a desired message object property or structure.

There is a wide variety of possible structures for MIME messages. One particular structure carries a Transport Neutral Encapsulation Format (TNEF) MIME element, which provides a high level of fidelity to original message object content. All TNEF messages have the same structure:

1. At the top level, a MIME entity with a Content-Type of "`multipart/mixed`" specifying all address elements, plus two child entities:

    1.1. A MIME entity with a Content-Type of "`text/plain`", containing a plain text rendering of the message body.

    1.2. A MIME entity with a Content-Type of "`application/ms-tnef`" and containing all attachment content, the HTML or RTF message body, and any message object properties for which no mapping to MIME header fields is defined, encoded as specified in [MS-OXTNEF].

Because a TNEF message is itself a MIME structure, MIME messages without a TNEF element are sometimes referred to as "pure MIME" to distinguish them from TNEF messages.

2.1 MIME Generation

This section specifies both conversion to pure MIME and conversion to TNEF from message objects.

When generating a MIME rendering of a message object, whether pure MIME or TNEF, MIME writers SHOULD retrieve all properties of the message object by issuing one of the following ROP sequences (see [MS-OXCROPS]):

- **RopGetPropertiesList** followed by **RopGetPropertiesSpecific**
- **RopGetPropertiesAll**

Clients can explicitly request conversion to pure MIME or TNEF by one of the following means. A MIME writer SHOULD honor such a client request for message format.

- A client can request conversion to pure MIME for all recipients by setting the value of the PidTagSendRichInfo property to FALSE on the message object, and request conversion to TNEF for all recipients by setting the same property value to TRUE.
- A client can request conversion to pure MIME for an individual recipient by setting the value of the PidTagSendRichInfo property to FALSE on that recipient, and request conversion to TNEF for an individual recipient by setting the same property value to TRUE.
- Alternatively, a client can request conversion to pure MIME for an individual one-off recipient by setting the NoRichInfo bit in the one-off entry ID, as specified in [MS-OXCDATA], and request conversion to TNEF by resetting the same bit.

Similarly, when conversion to pure MIME is requested, clients can explicitly request plain text or HTML message body generation by one of the following means. Again, a MIME writer SHOULD honor such a client request for message format.

- A client can request a specific MIME body format for all recipients by setting the value of the PidTagSendInternetEncoding property on the message object as specified in the table below.
- A client can request a specific MIME body format for an individual recipient by setting the value of the PidTagSendInternetEncoding property on the recipient as specified in the table below.

PidTagSendInternetEncoding property value (hex, ABNF)	Desired format
0x00060000 %x00.00.06.00	Plain text only
0x000E0000, %x00.00.0E.00	HTML only
0x00160000, %x00.00.16.00	Both plain text and HTML

2.1.1 Address Elements

In general, address elements are generated in MIME only and not in TNEF. However, when a TNEF message is generated, all address elements of messages attached to the top-level message are generated in TNEF only, as specified in [MS-OXTNEF]. This is because there is no MIME entity corresponding to the attached messages; they are wholly contained in the TNEF.

MIME writers MUST generate e-mail addresses for MIME recipients in compliance with the address specification dictated by [RFC2822]. For example, in cases where a display name is generated in MIME address header field, servers MUST use the encoding specified by [RFC2047] to encode any display name value that has characters that are not allowed in a MIME header field per [RFC2822]. These addresses are always SMTP addresses. When a client supports other types of e-mail addresses through the PidTagAddressType property, servers SHOULD use Internet Mail Connector Encapsulated Address (IMCEA) encapsulation of the e-mail address to form an SMTP address, as specified in section 2.1.1.8.

Address elements other than recipients, such as From and Sender, are represented in a message object by a group of four properties: display name, address type, e-mail address, and entry ID. In subsequent sections such properties might be referred to as a group. For example, "The PidTagSentRepresenting property group" includes the following properties:

PidTagSentRepresentingName, PidTagSentRepresentingAddressType, PidTagSentRepresentingEmailAddress, and PidTagSentRepresentingEntryId.

2.1.1.1 Recipients

To create a recipient in a MIME recipient header field, clients MUST create a message object recipient with either PidTagEntryId or both PidTagAddressType and PidTagEmailAddress properties that suffice to fully represent the recipient's e-mail address type and e-mail address. Clients SHOULD in addition set PidTagSmtpAddress, particularly to save the SMTP address when the value of PidTagAddressType is not SMTP.

Clients MUST set the PidTagRecipientType property value for each recipient as specified by the following table in order to specify whether a recipient is a To, Cc, or Bcc recipient:

PidTagRecipientType value	Recipient header field
0x00000001	To
0x00000002	Cc
0x00000003	Bcc

When generating MIME or TNEF, MIME writers SHOULD ignore recipient types other than To, Cc, and Bcc. MIME writers MUST generate one MIME recipient for a message object recipient that has a value of To, Cc, or Bcc. Each MIME recipient MUST be generated in the header field that corresponds to the PidTagRecipientType property value, as specified by the PidTagRecipientType value table.

Clients SHOULD set the PidTagDisplayName property for recipients, where that information is available. MIME writers SHOULD copy the PidTagDisplayName property value, when it exists, when generating display name of [RFC2822] address specification. The display name MUST be encoded as specified in [RFC2047] when necessary.

MIME writers MUST generate the angle address portion (angle-addr) of [RFC2822 section 3.4] address specification from addressee properties, used in the following order of preference: PidTagEntryId, PidTagAddressType/PidTagEmailAddress, PidTagSmtpAddress. More specifically:

1. If PidTagEntryId is present and bytes 4-19 are equal to the MUIDEMSAB uuid value {%xdc.a7.40.c8.c0.42.10.1a.b4.b9.08.00.2b.2f.e1.82}, then it is an address book entry ID. In this case MIME writers SHOULD look up the address book entry corresponding to the DN contained in the entry ID, and use the primary SMTP proxy address found on the address book entry. Otherwise, continue to step 2. (Entry ID

format is specified in [MS-OXCDATA], and procedure for looking up address book entries is specified in [MS-OXOABK].)

2. If PidTagEntryId is present and bytes 4-19 are equal to the MUIDOOP uuid value {%x81.2b.1f.a4.be.a3.10.19.9d.6e.00.dd.01.0f.54.02}, then it is a one-off entry ID. The e-mail type and address are encoded in the entry ID as specified in [MS-OXCDATA]. If the e-mail type is SMTP then use this e-mail address; otherwise, continue to step 6.

3. If PidTagEntryId is present and bytes 4-19 are some value other than is shown in items 1 and 2 above, the MIME writer MUST reject the recipient. If MIME generation is being done for SMTP then a failure DSN MUST be generated for that recipient. The format of a failure DSN is specified in [RFC3464]. The corresponding message object structure is referred to as a non-delivery receipt; its format is specified in [MS-OXOMSG]. Otherwise, continue to step 4.

4. If both PidTagAddressType and PidTagEmailAddress are present and PidTagAddressType matches "SMTP" then continue at step 6 using the value of PidTagEmailAddress. Otherwise, continue to step 5.

5. If the PidTagSmtpAddress property is present, use its value. Otherwise, continue to step 6.

6. If an e-mail address and address type are present, whether obtained from PidTagAddressType and PidTagEmailAddress or from an entry ID, but the address type does not match "SMTP", then the MIME writer SHOULD attempt IMCEA encapsulation of the e-mail address as specified in section 2.1.1.8.

7. Finally, if all of the above conditions fail then the MIME writer MUST reject the recipient. If MIME generation is being done for outbound SMTP then a failure delivery status notification (DSN) MUST be generated for that recipient. The format of a failure DSN is specified in [RFC3464]. The corresponding message object structure is referred to as a non-delivery receipt; its format is specified in [MS-OXOMSG].

2.1.1.1.1 To and Cc Recipients

To generate a To or Cc MIME header field, clients MUST add a recipient to the message object and set its PidTagRecipientType property value corresponding to desired recipient type, as specified by the PidTagRecipientType value table in section 2.1.1.1.

MIME writers MUST map recipients to the To or Cc MIME header fields as requested by clients. An exceptional situation occurs when generating MIME for an attached DSN message. A DSN message is one that has a PidTagMessageClass value of:

```
; The most common values are "REPORT.IPM.Note.NDR" and
"REPORT.IPM.Note.DR"
ReportMsgClass = "REPORT" 1*("." MsgClassToken) (".NDR" / ".DR")
```

```
MsgClassToken = ALPHA *(ALPHA / DIGIT)
```

In that case MIME writers MUST ignore the recipients of the attached message and MUST instead populate the To header field of the attached message using the PidTagReceivedRepresenting property group of the attached message (if they exist), or using the PidTagReceivedBy property group of the attached message otherwise.

When generating TNEF, MIME writers MUST not generate attRecipTable for the top-level message. For attached messages, MIME writers MUST copy all recipients, along with all of their properties, into the attRecipTable TNEF attribute in the TNEF body part as specified in [MS-OXTNEF]. This applies to attached DSN messages as well.

2.1.1.1.2 Bcc recipients

To generate a Bcc MIME header field, clients MUST add a recipient to the message object and set its PidTagRecipientType property value for that recipient to 0x00000003.

When generating a message for outbound submission to SMTP, MIME writers MUST NOT copy Bcc recipients to the MIME Bcc header field. This applies to MIME Bcc header field of attached messages as well. MIME writers MUST NOT copy Bcc recipients to the TNEF attRecipTable for attached messages.

When generating a message for POP3, IMAP4, or similar protocols, MIME writers SHOULD copy Bcc recipients to the MIME Bcc header field. This applies to MIME Bcc header field of attached messages as well. MIME writers SHOULD copy Bcc recipients to the TNEF recipient table for attached messages.

2.1.1.2 Reply-to

To generate a Reply-To MIME header field, clients MUST set the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties to the desired values, as specified in [MS-OXOMSG].

When generating MIME, MIME writers MUST generate a Reply-To MIME header field using the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties. MIME writers SHOULD ignore the PidTagReplyRecipientNames value if the count of names does not match the count of entries in the PidTagReplyRecipientEntries property. Assuming the counts do match, each entry in the value of the PidTagReplyRecipientNames property maps to one display name, and each entry ID in the value of the PidTagReplyRecipientEntries property maps to one address as follows:

1. If bytes 4-19 are equal to the MUIDEMSAB uuid value {%xdc.a7.40.c8.c0.42.10.1a.b4.b9.08.00.2b.2f.e1.82}, then it is an address book entry ID. In this case the MIME writer SHOULD look up the address book entry corresponding to the DN contained in the entry ID, and use its primary SMTP proxy address.

2. If bytes 4-19 are equal to the MUIDOOP uuid value {%x81.2b.1f.a4.be.a3.10.19.9d.6e.00.dd.01.0f.54.02}, then it is a one-off entry ID. The e-mail type and address are encoded in the entry ID and SHOULD be extracted. If the e-mail type is SMTP then the e-mail address SHOULD be used as is; otherwise, the address MUST be IMCEA-encapsulated as specified in section 2.1.1.8.

When generating TNEF, in addition to the above, MIME writers MUST copy the values of the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties to the attMsgProps in the TNEF body part as specified in [MS-OXTNEF].

2.1.1.3 From

To generate a From MIME header field, clients MUST set the PidTagSentRepresenting property group.

When generating MIME, MIME writers MUST generate a From header field using values of the PidTagSentRepresenting property group. The order of preference in that property group MUST be as specified in section 2.1.1.1.

When generating TNEF, in addition to the above, MIME writers MUST copy the values of the PidTagSentRepresenting property group to attSentFor and attMsgProps in the TNEF body part as specified in [MS-OXTNEF].

2.1.1.4 Sender

To generate a Sender MIME header field, clients MUST set the value of the PidTagSender property group.

MIME writers MUST generate a Sender header field using values of the PidTagSender property group. The order of preference in that property group MUST be as specified in section 2.1.1.1. MIME writers SHOULD NOT generate the Sender header field if the PidTagSender property group and the PidTagSentRepresenting property group represent the same recipient.

When generating TNEF, in addition to the above, MIME writers MUST copy the values of the PidTagSender property group to attFrom and attMsgProps in the TNEF body part as specified in [MS-OXTNEF].

2.1.1.5 Return Receipt

To generate a (non-standard) Return-Receipt-To header field, protocol clients MUST set the PidTagOriginatorDeliveryReportRequested property to TRUE and also MUST set the PidTagSentRepresenting property group to the desired values.

MIME writers MUST check the PidTagOriginatorDeliveryReportRequested property value first. If the property is not set or the value is FALSE, MIME writers MUST NOT generate the Return-Receipt-To header field.

If the PidTagOriginatorDeliveryReportRequested property is set and its value is TRUE and the PidTagSentRepresenting property group is set, MIME writers MUST copy the PidTagSentRepresenting property group to the Return-Receipt-To header field.

When generating TNEF, in addition to the above, MIME writers MUST copy the values of the PidTagOriginatorDeliveryReportRequested and the PidTagSentRepresenting property group to attMsgProps in the TNEF body part as specified in [MS-OXTNEF].

2.1.1.6 Read Receipt

To generate a Disposition-Notification-To MIME header field, protocol clients MUST set the PidTagReadReceiptRequested property to TRUE and also MUST set either the PidTagReadReceipt property group or the PidTagSentRepresenting property group to the desired values.

MIME writers MUST check the PidTagReadReceiptRequested property value first. If the property is not set or the value is FALSE, MIME writers MUST NOT generate the Disposition-Notification-To header field.

If the PidTagReadReceiptRequested property is set and its value is TRUE, MIME writers MUST generate the Disposition-Notification-To header field from the PidTagReadReceipt property group, if that property group is set. The order of preference in that property group MUST be as specified in section 2.1.1.1. If the PidTagReadReceipt property group is not set, protocol servers SHOULD generate the Disposition-Notification-To header field from the PidTagSentRepresenting property group.

MIME writers MUST generate the Disposition-Notification-To MIME header field as specified in [RFC3798].

When generating TNEF, in addition to the above, MIME writers MUST copy the values of the PidTagReadReceiptRequested, the PidTagReadReceipt and PidTagSentRepresenting groups of properties to attMsgProps in the TNEF body part as specified in [MS-OXTNEF].

2.1.1.7 Directory Lookups

Clients SHOULD specify the primary SMTP proxy address or the address book (EX) proxy address in all address elements of a message. But clients MAY use any proxy address. MIME writers SHOULD look up each proxy address in the address book, as specified in [MS-OXOABK]. If a matching address book entry is found, MIME writers SHOULD substitute its primary SMTP proxy address for the address specified by the client.

2.1.1.8 IMCEA Encapsulation

When no SMTP proxy address is available for an address element, protocol servers MUST encapsulate any other address type to produce the required SMTP address, utilizing the Internet Mail Connector Encapsulated Address (IMCEA) encapsulation mechanism as specified below. The domain part of the encapsulated SMPT address SHOULD be the MIME

writer's local domain, or another domain that "knows" how to de-encapsulate and deliver to the encapsulated address.

The IMCEA encapsulation mechanism is defined for the following address types:

Address type	Value of PidTagAddressType or related property
Address book	"EX"
Facsimile	"FAX"
X.400	"X400"

Encapsulated-address = "IMCEA" address-type "-" encoded-address "@" domain
address-type = *VCHAR
domain = dot-atom-text ; see [RFC2822] section 3.2.4 for definition

encoded-address = * (Escaped-chars/ Normal-chars)

Escaped-chars = (ESCSLASH / ESCCHARS)

; Encoded form for "/" (%x2F) is "_"
ESCSLASH = %x5F

; All OCTETS not ALPHA, DIGIT, or in "-=/"
; These are a "+" and the two hex digits of the OCTET's value.
ESCCHARS = "+" 2(HEXDIG)

; All other characters
Normal-chars= (ALPHA / DIGIT / HYPHEN / EQUALSIGN)
HYPHEN = %x2D
EQUALSIGN = %x3D

Encapsulated addresses MUST NOT include line breaks, and therefore can require longer line lengths than those recommended by [RFC2822].

2.1.2 Envelope Elements

Many message object properties that map to MIME header fields have string values. Where not otherwise specified below, the string values are simply copied from the property to the header field. When generating MIME header field values, the encoding specified in [RFC2047] MUST be used where necessary to encode Unicode characters.

Likewise, unless otherwise specified, when a MIME message with a TNEF body part is being generated, all message object properties SHOULD be copied to the attMsgProps attribute of the TNEF body part, even if there is also a corresponding MIME header field.

2.1.2.1 Message Class

When generating TNEF, MIME writers MUST copy the value of the PidTagMessageClass property to attMsgProps in the TNEF body part as specified in [MS-OXTNEF]. In addition, MIME writers SHOULD map the value of the PidTagMessageClass property to the attMessageClass attribute as specified in [MS-OXTNEF].

When generating pure MIME, the value of PidTagMessageClass SHOULD NOT be copied to MIME messages. Instead, its value is reflected in the structure of the MIME message, as specified in the following table. The MIME structure is indicated by listing the value of the Content-Type header field, indented according to how the MIME entities are nested.

PidTagMessageClass value	MIME structure
"IPM.Note.SMIME.MultipartSigned", or begins with "IPM.InfoPathForm." and ends with ".SMIME.MultipartSigned"	multipart/signed, as specified in [RFC3851] and [MS-OXOSMIME]
"IPM.Note.SMIME", or begins with "IPM.InfoPathForm." and ends with ".SMIME"	application/pkcs7-mime, as specified in [RFC3851] and [MS-OXOSMIME]
"REPORT.IPM.Note.DR" or "REPORT.IPM.Note.NDR" (other values MAY be substituted for "IPM.Note")	As specified in [RFC3464]: multipart/report text/html message/delivery-status <original message structure>
"REPORT.IPM.Note.RN" or "REPORT.IPM.Note.NRN" (other values MAY be substituted for "IPM.Note")	As specified in [RFC3798]: multipart/report text/html message/disposition-notification
begins with "IPM.Appointment."	text/calendar, as specified in [RFC2445] and [MS-OXCICAL]

begins with "IPM.Schedule.Meeting."	content mapped to text/calendar, as specified in [RFC2445] and [MS-OXCICAL] Top-level message structure is multipart/alternative or multipart/mixed, depending on the presence and type of message body and attachments; please refer to section 2.1.3
"IPM.Note" or any other value	Text/plain, text/html, multipart/alternative, multipart/related, or multipart/mixed, depending on the presence and type of message body and attachments; please refer to section 2.1.3

2.1.2.2 Content Class

MIME writers SHOULD generate the following values for Content-Class header field, based on the value of **PidTagMessageClass** property:

PidTagMessageClass value	Content-Class header field value
"IPM.Note.Microsoft.Fax"	"fax"
"IPM.Note.Microsoft.Fax.CA"	"fax-ca"
"IPM.Note.Microsoft.Missed.Voice"	"missedcall"
"IPM.Note.Microsoft.Conversation.Voice"	"voice-uc"
"IPM.Note.Microsoft.Voicemail.UM.CA"	"voice-ca"
"IPM.Note.Microsoft.Voicemail.UM"	"voice"

PidTagMessageClass value begins with	**Content-Class header field value**
"IPM.Note.Custom."	"urn:content-class:custom.", followed by the value of **PidTagMessageClass** property with

	"IPM.Note.Custom." prefix removed
"IPM.InfoPathForm."	If **PidLidInfoPathFormName** property has some value, Content-Class header field SHOULD be generated with the value of "InfoPathForm.", followed by a string, that is generated as follows: 1) MIME writer SHOULD take the value of **PidTagMessageClass** property, and remove "IPM.InfoPathForm." prefix. 2) If the remaining string contains a '.' symbol, the value SHOULD be truncated before the period '.'. 3) The value of **PidLidInfoPathFormName** property SHOULD be appended to this string, preceded by '.' character.

If MIME writer was unable to generate a value for Content-Class MIME header based on the value of **PidTagMessageClass** property, it SHOULD look up a value of **PidNameContentClass** property. If this property has a value, it SHOULD be used as a value of Content-Class header field, otherwise no header SHOULD be generated.

2.1.2.3 Unified Messaging Properties

To generate an X-CallingTelephoneNumber header field, clients SHOULD set the value of the PidTagSenderTelephoneNumber property to the desired value. They MAY instead use PidTagXSenderTelephoneNumber. MIME writers MUST copy either property to the X-CallingTelephoneNumber header field, preferring PidTagSenderTelephoneNumber.

To generate an X-VoiceMessageDuration header field, clients SHOULD set the value of the PidTagVoiceMessageDuration property to the desired value. They MAY instead use PidTagXVoiceMessageDuration. MIME writers MUST map either property to the X-VoiceMessageDuration header field, preferring PidTagVoiceMessageDuration. The value of the PidTagVoiceMessageDuration property is a PtypInteger32 and MUST be formatted as a decimal string in the header field.

To generate an X-VoiceMessageSenderName header field, clients SHOULD set the value of the PidTagVoiceMessageSenderName property to the desired value. They MAY instead use PidTagXVoiceMessageSenderName. MIME writers MUST copy either property to the X-VoiceMessageSenderName header field, preferring PidTagVoiceMessageSenderName.

To generate an X-FaxNumberOfPages header field, clients SHOULD set the value of the PidTagFaxNumberOfPages property to the desired value. They MAY instead use PidTagXFaxNumberOfPages. MIME writers MUST map either property to the X-FaxNumberOfPages header field, preferring PidTagFaxNumberOfPages. The value of the PidTagFaxNumberOfPages property is a PtypInteger32 and MUST be formatted as a decimal string in the header field.

To generate an X-AttachmentOrder header field, clients SHOULD set the value of the PidTagVoiceMessageAttachmentOrder property to the desired value. They MAY instead use PidTagXVoiceMessageAttachmentOrder. MIME writers MUST copy either property to the X-AttachmentOrder header field, preferring PidTagVoiceMessageAttachmentOrder.

To generate an X-CallID header field, clients SHOULD set the value of the PidTagCallId property to the desired value. They MAY instead use PidTagXCallId. MIME writers MUST copy either property to the X-CallID header field, preferring PidTagCallId.

2.1.2.4 Arbitrary MIME Header Fields

To generate an arbitrary header field on a MIME message, a client MUST create a named property in the PS_INTERNET_HEADERS property set, with the property name equal to the header field name and the data type equal to string. The value of this property MUST be set to the desired MIME header field value.

MIME writers MUST use the name and value of such a property to create a header field on the generated MIME message with the corresponding name and value. If necessary, MIME writers MUST encode the header field value as specified in [RFC2047]. But MIME writers MUST NOT create such a header field if a different message object property is already mapped to the same header field, or if the header name begins with one of the reserved name prefixes “X-Microsoft-Exchange-Organization-" or “X-Microsoft-Exchange-Forest-".

2.1.2.5 Importance

To generate an Importance header field, a client MUST set the value of the PidTagImportance property as specified in the following table.

PidTagImportance value	Importance header field value
0x00000000	Low
0x00000001	Normal
0x00000002	High

MIME writers MUST map the value of the PidTagImportance property to the Importance header field, as specified in the table. MIME writers MAY generate no Importance header field for a PidTagImportance value of 1 (normal) or for values other than 0, 1, or 2.

When generating TNEF, in addition to the above, MIME writers MUST copy the value of the PidTagImportance property to the attPriority and attMsgProps attributes as specified in [MS-OXTNEF].

2.1.2.6 Sensitivity

To generate a Sensitivity header field, a client MUST set the value of the PidTagSensitivity property as specified in the following table.

PidTagSensitivity value	Sensitivity header field value
0x00000000	Normal
0x00000001	Personal
0x00000002	Private
0x00000003	Company Confidential

MIME writers MUST map the value of the PidTagSensitivity property to the Sensitivity header field, as specified in the table. MIME writers MAY generate no Sensitivity header field value for a PidTagSensitivity value of 0 (normal) or for values other than 0,1,2, or 3.

When generating TNEF, in addition to the above, MIME writers MUST copy the value of the PidTagSensitivity property to the attMsgProps attribute as specified in [MS-OXTNEF].

2.1.2.7 Sent Time

To generate a Date header field, clients MUST set the value of PidTagClientSubmitTime to the desired value. The property value MUST be expressed in UTC time.

MIME writers MUST copy the value of the PidTagClientSubmitTime property to the Date header field, formatting it as specified by [RFC2822]. MIME writers SHOULD include hours, minutes, and seconds in the generated Date header field value. MIME writers MAY convert the date and time value from UTC to another time zone of their choice.

If no value is specified for PidTagClientSubmitTime when a message is submitted to SMTP, MIME writers SHOULD generate a Date header field with a value of the current time.

When generating TNEF, in addition to the above, MIME writers MUST copy the value of the PidTagClientSubmitTime property to the attDateSent and attMsgProps attributes as specified in [MS-OXTNEF].

2.1.2.8 Subject

To generate a Subject header field, clients SHOULD set the PidTagSubjectPrefix and PidTagNormalizedSubject properties on the message object. Clients MAY set the PidTagSubject property instead, but in that case, the separation of subject from subject prefix is vulnerable to limitations of the server's parsing procedure, which is specified in section 2.2.2.6.1. Subject property values SHOULD NOT contain line breaks.

MIME writers MUST generate the Subject header field by combining the values of the PidTagSubjectPrefix and PidTagNormalizedSubject properties.

If those two properties are not available, MIME writers MUST copy the value of the PidTagSubject property to the Subject header field. MIME writers MAY truncate the subject value; a typical size limit is the first 255 characters. The property value SHOULD NOT be truncated in the middle of a multibyte character.

When generating TNEF, in addition to the above, MIME writers MUST copy the message subject (however obtained) to the attSubject and attMsgProps attributes as specified in [MS-OXTNEF].MIME writers SHOULD also copy the PidTagSubjectPrefix and PidTagNormalizedSubject properties, with their values, to the attMsgProps attribute.

2.1.2.9 Conversation Topic

To generate a Thread-Topic header field, clients MUST set the value of the PidTagConversationTopic property to the desired value. Clients SHOULD set this property to the same value as PidTagNormalizedSubject, with any subject prefix removed as specified in section 2.2.2.6.1.

MIME writers MUST copy the value of the PidTagConversationTopic property to the Thread-Topic header field.

2.1.2.10 Conversation Index

To generate a Thread-Index header field, clients MUST set the value of the PidTagConversationIndex property to the desired value, as specified in [MS-OXOMSG]

MIME writers MUST copy the value of the PidTagConversationIndex property to the Thread-Index header field. The property type is binary; the value MUST be encoded using base64 encoding as specified in [RFC2045].

2.1.2.11 Message ID

MIME writers MUST copy the value of the PidTagInternetMessageId property to the Message-ID header field. If no value is specified for PidTagInternetMessageId when a message is submitted to SMTP, MIME writers SHOULD generate a value as specified in [RFC2822].

Clients SHOULD NOT set the PidTagInternetMessageId property when submitting a message via RPC. Per [RFC2822] the value of Message-ID MUST be unique, and for this reason it is normally assigned by servers. Servers MAY overwrite PidTagInternetMessageId from a client before submitting the message to SMTP.

Once set, the value of the Message-ID header field and the corresponding property value, PidTagInternetMessageId, SHOULD remain constant. MIME writers SHOULD NOT overwrite the value of PidTagInternetMessageId when generating MIME for protocols such as POP/IMAP.

2.1.2.12 References

To generate a References header field, clients MUST set the value of the PidTagInternetReferences property to the desired value.

MIME writers MUST copy the value of the PidTagInternetReferences property to the References header field.

2.1.2.13 Categories

To generate a Keywords header field, clients MUST set the value of the PidLidCategories property to the desired values. The type of PidLidCategories is multiple strings; each category SHOULD be mapped to a single keyword.

MIME writers SHOULD copy each sub-value of the PidLidCategories property to a separate keyword in the Keywords header field, with a comma (U+002C) and space (U+0020) separating each keyword. MIME writers MAY drop the PidLidCategories instead of copying it to the Keywords header field, in order to avoid conflict among different sets of Categories in different organizations.

2.1.2.14 In-Reply-To Message ID

To generate an In-Reply-To header field, clients MUST set the value of the PidTagInReplyToId property to the desired value.

MIME writers MUST copy the value of the PidTagInReplyToId property to the In-Reply-To header field.

2.1.2.15 List Server Properties

To generate a List-Help header field, clients MUST set the value of the PidTagListHelp property to the desired value.

MIME writers MUST copy the value of the PidTagListHelp property to the List-Help header field.

To generate a List-Subscribe header field, clients MUST set the value of the PidTagListSubscribe property to the desired value.

MIME writers MUST copy the value of the PidTagListSubscribe property to the List-Subscribe header field.

To generate a List-Unsubscribe header field, clients MUST set the value of the PidTagListUnsubscribe property to the desired value.

MIME writers MUST copy the value of the PidTagListUnsubscribe property to the List-Unsubscribe header field.

2.1.2.16 Language Properties

To generate an [RFC3282] Accept-Language header field, clients MUST set the value of the PidNameAcceptLanguage property to the desired value.

MIME writers MUST copy the value of the PidNameAcceptLanguage property to the Accept-Language header field. If the PidNameAcceptLanguage property is missing, MIME writers SHOULD identify the acceptable locales of the sender's mailbox and write the corresponding language tag, as specified by [RFC4646], as the value of the Accept-Language header field.

To generate an [RFC3282] Content-Language header field, clients MUST set the value of the PidTagMessageLocaleId property to the desired locale ID.

MIME writers MUST use the value of the PidTagMessageLocaleId property to write the Content-Language header. The value of PidTagMessageLocaleId is a Windows LCID (a 32-bit integer value), but the header field value is a language tag as specified by [RFC4646]. Mapping between LCID and language tag MUST be done as specified in [MS-LCID].

2.1.2.17 Classification Properties

To generate header fields related to message classification, clients MUST set the value of the PidLidClassified property to true and the following properties to their desired values: PidLidClassification, PidLidClassificationDescription, PidLidClassificationGuid, PidLidClassificationKeep.

When the value of the PidLidClassified property is TRUE, MIME writers MUST copy all classification property values to their corresponding header fields as specified below:

Classification property	Classification header field	Property value mapping
PidLidClassified	X-Microsoft-Classified	True => “true" False => no header
PidLidClassificationKeep	X-Microsoft-ClassKeep	copy string value
PidLidClassification	X-Microsoft-Classification	copy string value
PidLidClassificationDescription	X-Microsoft-ClassDesc	copy string value
PidLidClassificationGuid	X-Microsoft-ClassID	True => “true" False => no header

2.1.2.18 Payload Properties

To generate an X-Payload-Provider-Guid header field, clients MUST set the value of the PidTagAttachPayloadProviderGuidString property to the desired value.

MIME writers MUST copy the value of the PidTagAttachPayloadProviderGuidString property to the X-Payload-Provider-Guid header field.

To generate an X-Payload-Class header field, clients MUST set the value of the PidTagAttachPayloadClass property to the desired value.

MIME writers MUST copy the value of the PidTagAttachPayloadClass property to the X-Payload-Class header field.

2.1.2.19 Has Attach

To generate an X-MS-HasAttach header field, clients MUST add at least one attachment to the message object’s attachment table.

When the message object’s attachment table contains at least one attachment, MIME writers MUST generate an X-MS-HasAttach header field with a value of “Yes". When the message object’s attachment table is empty, MIME writers MUST NOT generate a X-MS-HasAttach header field.

2.1.2.20 Auto Response Suppress

To generate an X-Auto-Response-Suppress header field, clients MUST set the value of the PidTagAutoResponseSuppress property to its desired value.

When the PidTagAutoResponseSuppress property has a value of 0 or -1, MIME writers MUST map its value to the X-Auto-Response-Suppress header field as follows.

PidTagAutoResponseSuppress property value	X-Auto-Response-Suppress header field value
0	"None"
-1	"All"

When the PidTagAutoResponseSuppress property has a value other than 0 or -1, MIME writers MUST construct the value of the X-Auto-Response-Suppress header field as follows. For each bit of the value of PidTagAutoResponseSuppress that is set (left-hand column), append the string in the right-hand column to the header field value. If the header field value was nonempty, append a comma (U+0032) and space (U+0020) before the new value.

PidTagAutoResponseSuppress property value	X-Auto-Response-Suppress header field value	Description
0x00000001	"DR"	suppress delivery reports from transport
0x00000002	"NDR"	suppress non-delivery reports from transport
0x00000004	"RN"	suppress read notifications from receiving client
0x00000008	"NRN"	suppress non-read notifications from receiving client
0x00000010	"OOF"	suppress out-of-office notifications
0x000000020	"AutoReply"	suppress auto-reply messages other than out-of-office notifications

For example, if the value of PidTagAutoResponseSuppress is 0x000C, the header field MUST be written as:

```
X-Auto-Response-Suppress: RN, NRN
```

2.1.2.21 Is Auto Forwarded

To generate an X-MS-Exchange-Organization-AutoForwarded header field, clients MUST set the value of the PidTagAutoForwarded property to TRUE.

If the value of the PidTagAutoForwarded property is TRUE, MIME writers MUST generate the following header field:

X-MS-Exchange-Organization-AutoForwarded: true

If the property is absent or the property value is false, no header field SHOULD be generated.

2.1.2.22 Sender Id Status

To generate an X-MS-Exchange-Organization-SenderIdResult header field, clients MUST set the value of the PidTagSenderIdStatus property to its desired value.

MIME writers MUST copy the value of the PidTagSenderIdStatus property, which is a PtypInteger32, to the X-MS-Exchange-Organization-SenderIdResult header field, formatting it as a string.

2.1.2.23 Purported Sender Domain

To generate an X-MS-Exchange-Organization-PRD header field, clients MUST set the value of the PidTagPurportedSenderDomain property to its desired value.

MIME writers MUST copy the value of the PidTagPurportedSenderDomain property to the X-MS-Exchange-Organization-PRD header field.

2.1.2.24 Spam Confidence Level

To generate an X-MS-Exchange-Organization-SCL header field, clients MUST set the value of the PidTagContentFilterSpamConfidenceLeve property to its desired value in the range -1 to 10.

MIME writers MUST copy the value of the PidTagContentFilterSpamConfidenceLeve property, which is an Int32, to the X-MS-Exchange-Organization-SCL header field, formatting it as a decimal numeric string.

2.1.2.25 Phishing Confidence Level

To generate an X-MS-Exchange-Organization-PCL header field, clients MUST set the value of the PidTagContentFilterSpamConfidenceLeve property to its desired value in the range -1 to 10.

MIME writers MUST copy the value of the PidTagContentFilterSpamConfidenceLeve property, which is an Int32, to the X-MS-Exchange-Organization-PCL header field, formatting it as a decimal numeric string.

2.1.2.26 Flag Request

To generate an X-Message-Flag header field, clients MUST set the value of the PidLidFlagRequest property to its desired value.

MIME writers MUST copy the value of the PidLidFlagRequest property to the X-Message-Flag header field.

2.1.2.27 TNEF Correlation Key

When creating a new TNEF message, MIME writers MUST choose a unique key relating the TNEF body part to its parent message. (MIME writers SHOULD use the value of PidTagInternetMessageId for this purpose.) The chosen value MUST be written in two places:

1. As the value of the X-MS-TNEF-Correlator header field on the MIME message;
2. And as the value of PidTagTnefCorrelationKey in the attMsgProps attribute of the TNEF body part itself.

This pair of values SHOULD be used by MIME writers to validate that the top-level message and its TNEF body part do in fact belong to each other, and are not (for instance) the result of a non-TNEF-aware MUA forwarding a message with attached TNEF body part and retaining the attachment.

2.1.2.28 Received Header Fields

MIME writers SHOULD, under certain circumstances, copy all Received header fields from PidTagTransportMessageHeaders to the generated MIME header. MIME writers MUST NOT copy Received header fields to a MIME message that is bound for SMTP, but SHOULD copy the Received header fields to a MIME message that is bound for POP3 or IMAP4. With the exception of headers specifically mentioned elsewhere in section 2.1.2, headers beginning with the reserved name prefixes “X-MS-Exchange-Organization" and “X-MS-Exchange-Forest" SHOULD NOT be copied to PidTagTransportMessageHeaders.

Clients SHOULD NOT set the value of PidTagTransportMessageHeaders. This property value SHOULD be set only upon delivery of a message from SMTP, in which case it SHOULD be set to the MIME message header.

2.1.3 Body Text

When generating pure MIME, MIME writers MUST generate a single MIME entity for the message body, and it MUST be the first entity generated. (For message objects without attachments, it SHOULD be the only MIME entity generated.) The MIME entity generated for the message body can have several different structures, some of them fairly complex.

In the discussion that follows, please refer to section 2.1.4 for diagrams of message structure with attachments, and for how to determine whether an attachment object represents an inline attached file.

2.1.3.1 Client Actions

To create a plain text message body in MIME, clients SHOULD set the value of the PidTagBody property. Additionally, clients SHOULD set the value of the PidTagInternetCodepage property to a codepage corresponding to the charset that the client wants to appear in MIME. Clients SHOULD NOT create inline attachment objects when the message object's best body format is plain text.

To create an HTML message body in MIME, clients SHOULD set the value of the PidTagHtml property to the desired HTML text. When this property is set, clients MUST set the value of the PidTagInternetCodepage to the codepage of the HTML text (note that PidTagHtml is a Binary property, not a String property). Clients MAY, instead, set the value of the PidTagRtfCompressed property to the desired body text in compressed RTF format, depending on the MIME writer to convert this text to HTML format. Clients MUST NOT create HTML message text in Unicode (UTF-16LE), and the value of PidTagInternetCodepage MUST NOT be set to 1200. UTF-32 and UTF-16GE are also not acceptable for this purpose; UTF-7 (codepage 65000) and UTF-8 (codepage 65001) are acceptable.

To create a multipart/related message body in MIME with HTML body text and inline images, clients SHOULD set the value of the PidTagHtml property to the desired HTML text. When this property is set, the value of the PidTagInternetCodepage property MUST be set to the codepage of the HTML text (note that PidTagHtml is a binary property, not a string property). Clients MUST supply a value for either the PidTagAttachContentId or the PidTagAttachContentLocation property on related file attachments such as images; PidTagAttachContentId SHOULD be chosen for this purpose. Depending on the choice of attachment property, inline image links in the HTML body MUST use either:

1. The "cid:" URI scheme and a unique content identifier that matches the value of the PidTagAttachContentId property on the corresponding attachment object
2. A copy of the value of the PidTagAttachContentLocation property [see MS-OXCMSG] on the corresponding attachment object.

Instead of setting the value of the PidTagHtml property, clients MAY set the value of the PidTagRtfCompressed property and include OLE attachments, depending on the protocol server to convert the RTF text to HTML and the static renderings of the OLE attachments to image attachments. Please refer to section 2.1.3.7 for further details.

For plain text messages, clients SHOULD write the value of the PidTagBody property in Unicode and SHOULD set the value of the PidTagInternetCodepage property to the codepage matching the sender's preferred charset. When generating a MIME element for the plain text body, MIME writers MUST map this codepage to a charset name, MUST convert the Unicode text into that charset, and MUST write that charset name to the value of the Content-Type header field's charset parameter. The plain text MIME element generated for a TNEF message SHOULD be treated in the same way.

For HTML messages, clients SHOULD write the value of the PidTagHtml property using text in the sender's preferred charset. Clients MUST set the value of the PidTagInternetCodepage property to the codepage corresponding to the preferred charset. Clients MUST NOT use UTF-16 (codepage 1200) as the preferred charset. If the HTML document contains a content-type meta tag, its charset parameter value SHOULD match the preferred charset.

When generating a MIME element or elements for an HTML message body, MIME writers MUST map the value of the PidTagInternetCodepage property to a charset name, MUST write the MIME element body in that charset, and MUST write that charset name as the value of the Content-Type header field's charset parameter. If the HTML document contains a content-type meta tag, its charset parameter value SHOULD match the Content-Type header field's charset parameter value.

For RTF messages, clients SHOULD write the value of the PidTagRtfCompressed property using text in the sender's preferred charset. Clients SHOULD set the value of the PidTagInternetCodepage property to the codepage corresponding to the preferred charset. The preferred charset MUST NOT be UTF-16 (codepage 1200). MIME writers MUST NOT rely on the value of the PidTagInternetCodepage property, but treat it as a preference; MIME writers SHOULD instead rely on the value of one or more \ansicpg elements in the RTF stream, as specified in [MS-RTF], to determine the actual body codepage.

When generating a MIME element or elements for an RTF message body, MIME writers MUST convert the RTF text to plain text or HTML, MUST map the body codepage to a charset name, MUST write the MIME element body in that charset, and MUST write that charset name as the value of the Content-Type header field's charset parameter

Even if a message object has no body, clients SHOULD set the value of the PidTagInternetCodepage property to indicate a preferred charset for header field text, to be used in [RFC2047] encoding.

When generating header fields for a MIME entity, it might be necessary to encode the characters as specified in [RFC2047]. MIME writers SHOULD use the same charset for all

header fields and the message body. Attachments which are themselves messages are independent and can have a different charset.

2.1.3.2 Message Body in TNEF

When generating TNEF, MIME writers SHOULD identify the "best body" property of the message object as specified in [MS-OXBBODY] and copy its value to the attMsgProps attribute of the TNEF body part. MIME writers MUST also place a plain text version of the message body in the first child body part of the TNEF message, as specified at the beginning of section 2, generating plain text from the value of the "best body" property if necessary. Finally, when the best body is plain text, MIME writers SHOULD also write a matching value for the PidTagRtfCompressed property (generating it if necessary) to the attMsgProps attribute of the TNEF body part.<1>

2.1.3.3 Simple Plain Text Message Body

When the best body format type is plain text, MIME writers SHOULD generate a single MIME entity with the value of its content-type header field set to text/plain.

The charset parameter value of this MIME entity's Content-Type header field SHOULD be set to a charset corresponding to the value of the PidTagInternetCodepage property (a codepage number). If there is no PidTagInternetCodepage property, the value of the PidTagMessageCodepage property MAY be used instead, but in that case it SHOULD first be mapped from a Windows codepage to the corresponding Internet codepage. MIME writers SHOULD verify that the plain text, which is stored as UTF-16, can actually be encoded in this charset and SHOULD, if necessary, choose a different charset that can in fact encode the entire message body; the codepage properties express a preference rather than a requirement.

The value of the PidTagBody property MUST be written to the content of the text/plain MIME element, after being converted to the chosen charset.

2.1.3.4 HTML Text Message Body without Inline Attachments

When the best body format type is HTML and no inline attachment objects exist, MIME writers SHOULD generate a MIME entity with multipart/alternative for the value of its Content-Type header field, and with two child entities:

1. The first child entity MUST have "`text/plain`" for the value of its Content-Type header field. Its body SHOULD be plain text generated from the value of the PidTagHtml property, but MAY instead be copied from the value of the PidTagBody property, assuming that it contains substantially similar text.

2. The second child entity MUST have "`text/html`" for the value of its Content-Type header field. Its body MUST be the value of the PidTagHtml property.

The plain text and charset parameters SHOULD be processed as specified in section 2.1.3.3. HTML text MAY be processed in exactly the same way, or characters that do not fit the preferred charset MAY instead be encoded within the HTML.

2.1.3.5 HTML Text Message Body from RTF without Inline Attachments

When the best body format type is RTF and no inline (OLE) attachment objects exist, MIME writers SHOULD generate a multipart/alternative MIME entity with two child entities:

1. The first child entity MUST have "`text/plain`" for the value of its Content-Type header field. Its body SHOULD be plain text generated from the value of the PidTagRtfCompressed property, but MAY instead be copied from the value of the PidTagBody property, assuming that it contains substantially similar text.

2. The second child entity MUST have "`text/html`" for the value of its Content-Type header field. Its body MUST be HTML text. The HTML text SHOULD be generated from the value of the PidTagRtfCompressed property, but MAY instead be copied form the value of the PidTagHtml property, assuming that it contains substantially similar text.

The text and charset parameters SHOULD be processed as specified in section 2.1.3.4.

2.1.3.6 HTML Text Message Body with Inline Attachments

When the best body format type is HTML and inline attachment objects exist, MIME writers SHOULD generate a MIME entity with "multipart/related" for the value of its Content-Type header field and two or more child elements.

1. The first child entity MUST be a "`multipart/alternative`" structure, exactly as specified in section 2.1.3.4.

2. Subsequent child entities MUST be generated from the message object's inline attachments. Each entity MUST be generated as specified in section 2.1.4.1.

MIME writers SHOULD verify that the HTML text actually contains a reference to each inline attachment object, either by its PidTagAttachContentId or PidTagAttachContentLocation property, as specified in section 2.1.3.1. If the HTML text contains no such reference then the MIME writer SHOULD consider this attachment object as not inline and generate its MIME entity as a peer of the multipart/related MIME entity, instead of as its child.<2>

2.1.3.7 HTML Text Message Body from RTF with Inline (OLE) Attachments

When the best body format type is RTF and inline (OLE) attachment objects exist, MIME writers SHOULD generate a MIME entity with multipart/related for the value of its Content-Type header field and three or more child entities.

1. The first child entity MUST be a multipart/alternative structure, exactly as specified in section 2.1.3.5.

2. Subsequent child entities MUST be generated from the message object's inline attachments. Each entity MUST be generated as specified in section 2.1.4.4, since inline attachment objects in RTF messages are always OLE attachments.

2.1.3.8 Calendar Items and Meeting Messages

A message object is a calendar item when the value of PidTagMessageClass starts with "`IPM.Appointment.`" or equals "`IPM.Appointment`". A message object is a meeting message when the value of PidTagMessageClass starts with "`IPM.Schedule.Meeting.`" or equals "`IPM.Schedule.Meeting`". Clients SHOULD create items of these types with a best body format type of RTF. Clients MAY use a plain text body instead, but SHOULD NOT create calendar items or meeting messages with a best body format type of HTML.

Each of the leaf MIME entities specified in this section SHOULD use UTF-8 as its charset, as preferred by [RFC2445].

2.1.3.8.1 Plain Text Calendar Message

When the best body format type of a calendar item or meeting message is plain text, MIME writers SHOULD generate a MIME entity with multipart/alternative for the value of its Content-Type header field and two child entities.

1. The first child entity MUST have "`text/plain`" for the value of its Content-Type header field, and its content MUST be copied from PidTagBody.

2. The second child entity MUST have text/calendar for the value of its Content-Type header field, and its content MUST be generated as specified in [MS-OXCICAL].

2.1.3.8.2 Calendar Message Without Inline Attachments

When the best body format type of a calendar item or meeting message is RTF and there are no inline attachments, MIME writers SHOULD generate a MIME entity with "`multipart/alternative`" for the value of its Content-Type header field and three child entities.

1. The first child entity MUST have "`text/plain`" for the value of its Content-Type header field. Its content SHOULD be plain text generated from the value of the PidTagRtfCompressed property, but MAY instead be copied from the value of the PidTagBody property, assuming that it contains substantially similar text.

2. The second child entity MUST have "`text/html`" for the value of its Content-Type header field. Its content SHOULD be HTML text generated from the value of the PidTagRtfCompressed property, but MAY instead be copied from the value of the PidTagHtml property, assuming that it contains substantially similar text.

3. The third child entity MUST have "`text/calendar`" for the value of its Content-Type header field, and its content MUST be generated as specified in [MS-OXCICAL].

2.1.3.8.3 Calendar Message with Inline Attachments

When the best body format type of a calendar item or meeting message is RTF and there are inline attachments, MIME writers SHOULD generate a MIME entity with "`multipart/related`" for the value of its Content-Type header field and two or more child entities.

1. The first child entity MUST be a multipart/alternative structure generated as specified in section 2.1.3.8.2.

2. Subsequent child entities MUST be generated from the message object's inline attachments. Each entity MUST be generated as specified in section 2.1.4.1.

2.1.4 Attachments

Each attachment object in a message object represents one attachment. MIME writers SHOULD classify attachment objects based on the value of PidTagAttachMethod property:

- 5 - embedded message attachments
- 6 - OLE attachments
- All other values - ordinary file attachments

Note that ordinary file attachments might contain additional Macintosh-specific data. These attachments require special handling and are discussed in section 2.1.4.3. Additionally, MIME writers SHOULD classify attachment objects as inline or not inline as specified in section 2.1.4.1.

MIME writers SHOULD generate different MIME structures for the message depending on the presence of inline and non-inline attachments, as specified in the three examples that follow.

1. If both inline and non-inline attachments are present, MIME writers SHOULD generate the following structure. Figure 1 presents a graphical representation of the actual message structure which follows the figure:

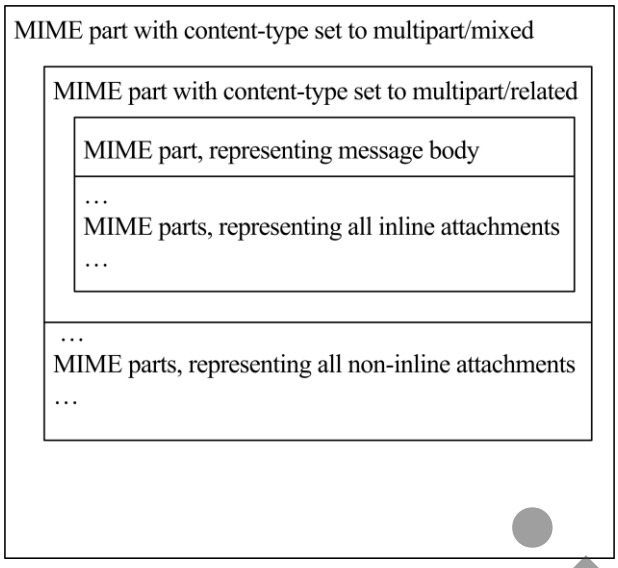

Figure 1 Inline and non-inline attachments present

Example 1:

```
From: <user1@example.com>
To: <user2@example.com>
Subject: Example with inline and non-inline attachments.
Date: Mon, 10 Mar 2008 14:36:46 -0700
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="simple boundary 1"

--simple boundary 1
Content-Type: multipart/related; boundary="simple boundary 2"

--simple boundary 2
Content-Type: multipart/alternative; boundary="simple boundary 3"

--simple boundary 3
Content-Type: text/plain

...Text without inline reference...
--simple boundary 3
Content-Type: text/html
```

...Text with inline reference...
--simple boundary 3--
--simple boundary 2
Content-Type: image/png; name="inline.PNG"
Content-Transfer-Encoding: base64
Content-ID: <6583CF49B56F42FEA6A4A118F46F96FB@example.com>
Content-Disposition: inline; filename="Inline.png"

...Attachment data encoded with base64...
--simple boundary 2--

--simple boundary 1
Content-Type: image/png; name=" Attachment "
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="Attachment.png"

...Attachment data encoded with base64...
--simple boundary 1--

2. If only inline attachments are present, MIME writers SHOULD generate the following structure. Figure 2 presents a graphical representation of the actual message structure which follows the figure:

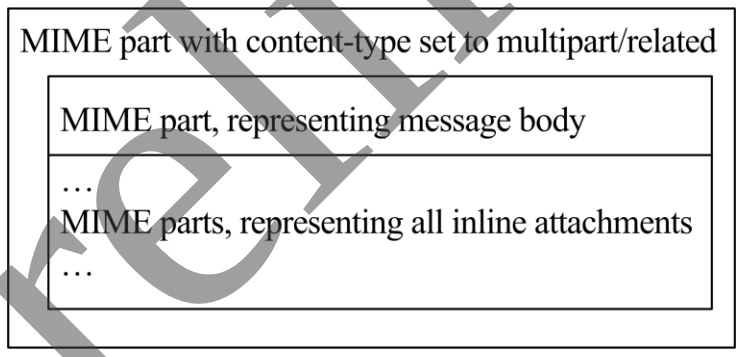

Figure 2 Only inline attachments present

Example 2:
From: <user1@example.com>
To: <user2@example.com>
Subject: Example with inline attachment.

Date: Mon, 10 Mar 2008 14:36:46 -0700
MIME-Version: 1.0
Content-Type: multipart/related; boundary="simple boundary"

--simple boundary
Content-Type: text/html;

...Text with reference...

--simple boundary
Content-Type: image/png; name="inline.PNG"
Content-Transfer-Encoding: base64
Content-ID: <6583CF49B56F42FEA6A4A118F46F96FB@example.com>
Content-Disposition: inline; filename=" inline.png"

...Attachment data encoded with base64...
--simple boundary--

3. If only non-inline attachments are present, MIME writers SHOULD generate the following structure. Figure 3 presents a graphical representation of the actual message structure which follows the figure:

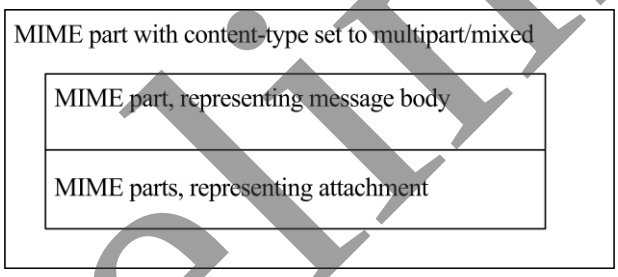

Figure 3 Only non-inline attachments present

Example 3:

From: <user1@example.com>
To: <user2@example.com>
Subject: Example with non-inline attachment.
Date: Mon, 10 Mar 2008 14:36:46 -0700
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="simple boundary"

```
--simple boundary
Content-Type: text/plain;

...Text without reference...

--simple boundary
Content-Type: image/png; name=" Attachment"
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="Attachment.png"

...Attachment data encoded with base64...
--simple boundary--
```

2.1.4.1 Inline Attachments

Clients SHOULD NOT create inline attachments if the best body text format is plain text. MIME writers SHOULD ignore PidTagAttachFlags and other indications that an attachment is inline for plain text messages. Likewise, clients SHOULD NOT designate attached message objects as inline, and MIME writers SHOULD NOT treat attached message objects as inline.<3>

2.1.4.1.1 Inline Attachments in RTF Messages

If the best body text format is RTF, MIME writers SHOULD treat all OLE attachments, and only OLE attachments, as inline attachments <4>. OLE attachments have 0x0000006 for the value of PidTagAttachMethod.

RTF text does not contain explicit references to inline attachments as HTML text does. Instead, the position of an inline attachment in the RTF text is indicated by an "`\objattph`" tag; clients MUST insert such a tag into the RTF text for each inline attachment, as specified in [MS-OXRTFEX]). Clients MUST also set the value of the PidTagRenderingPosition property to indicate the order of inline attachments: the attachment with the lowest value of this property matches the first "`\objattph`" tag; the next lowest matches the second "`\objattph`" tag, and so forth. Finally, clients SHOULD set the 0x00000002 bit in the value of the PidTagAttachmentFlags property to indicate that it is inline. MIME writers MUST sort inline attachments by the value of PidTagRenderingPosition when converting RTF text with inline attachments to HTML, and map the RTF "`\objattph`" tag to an HTML IMG tag at the corresponding position in the generated HTML.

2.1.4.1.2 Inline Attachments in HTML Messages

To mark an attachment object in a message whose best body text format is HTML as inline, clients MUST

1. Set bit 2 (0x00000004) in the value of the attachment object's PidTagAttachFlags property to true.
2. Set the value of either PidTagAttachContentId (preferred) or PidTagAttachContentLocation on the attachment object.
3. PidTagAttachContentBase MAY be set to fully qualify a relative URI in PidTagAttachContentLocation (see [RFC2557] for more information).
4. Include a tag that refers to the URI specified in (2) in the HTML message text. If PidTagAttachContentId is used, the URI MUST use the "cid:" scheme.

MIME writers SHOULD NOT rely entirely on bit 0x00000002 of the PidTagAttachFlags property value to be set correctly for all attachments. Instead, MIME writers SHOULD verify all three conditions specified above when deciding whether to treat an attachment as inline.5

2.1.4.2 Attached Files

This section describes MIME generation for ordinary attachments without Macintosh-specific data.

The remainder of this section concerns generating attachments for pure MIME messages. When generating a TNEF message, all attachment data MUST be written to the TNEF body part as specified in [MS-OXTNEF].

2.1.4.2.1 File Name

For the file name in a MIME representation of an attached file, MIME writers SHOULD use the value of the PidTagAttachLongFilename property. If this value is not available, a MIME writers SHOULD use the value of the PidTagAttachFilename property, and MAY use empty string if this value is also not available. The attached file name SHOULD be written to several different MIME headers as specified in the next section.

If a file extension is needed for mapping the attachment content type, it SHOULD be obtained by copy all characters after the last "." (U+002E) character in the file name.

2.1.4.2.2 Content-type, content-description, content-disposition

MIME writers SHOULD determine the primary value of the content-type header field for an attached file using the following steps:

1. Acquire the value of the PidTagAttachMimeTag property. If this value is not available, MIME writers SHOULD determine content-type by mapping it from the file extension (which SHOULD be determined from the attachment file name as

specified in section 2.1.4.2.1), or by examining the file content itself. As a last resort the MIME writer MUST use "`application/octet-stream`".

2. If the value acquired in the previous step doesn't match requirements for MIME content-type as specified in [RFC2045], or if the value represents any multipart content-type, or if the value matches one of the values listed below, then MIME writers SHOULD replace it with "`application/octet-stream`":
    - application/applefile
    - application/mac-binhex40
    - message/rfc822

The value acquired as a result is then used as the value of the content-type MIME header field. MIME writers SHOULD also generate the "`name`" parameter for this header field, using the attachment file name (determined as specified in section 2.1.4.2.1) as a value.

MIME writers SHOULD generate a Content-Description header field using the value of the PidTagDisplayName property. If the property has no value, an empty header field MAY be generated. The value of the Content-Description header field SHOULD be encoded as specified in [RFC2047] when applicable.

The value for Content-Disposition header field SHOULD be generated based on whether the attachment is inline or not, as specified in section 2.1.4.1. For inline attachments it MUST be "`inline`", and for non-inline attachments it MUST be "`attachment`". MIME writers SHOULD generate the following parameters for this header field:

- filename: the attachment file name determined as specified in section 2.1.4.2.1 MUST be used as a value.
- size: PidTagAttachSize property value SHOULD be used as a parameter value. The size parameter SHOULD be generated only if this property value is available and greater than 0.
- creation-date: PidTagCreationTime property value SHOULD be used as parameter value; if property value is not available, current time SHOULD be used. In either case the creation time SHOULD be converted from UTC to a local time zone of the MIME writer's choice and formatted as specified in [RFC2822].
- modification-date: PidTagLastModificationTime property value SHOULD be used as parameter value; if property value is not available, current time SHOULD be used. In either case the creation time SHOULD be converted from UTC to a local time zone of the MIME writer's choice and formatted as specified in [RFC2822].

2.1.4.2.3 Content-ID, Content-Location, Content-Base

MIME writers SHOULD generate a Content-ID MIME header field if the value of the PidTagAttachContentId contains non-whitespace characters. All trailing and leading

whitespace characters SHOULD be removed from this value. If the resulting value doesn't start with '<' (U+003C), or doesn't end with '>' (U+003E), it SHOULD be enclosed in angle brackets. The resulting string becomes the value of the Content-ID header field.

MIME writers SHOULD generate a Content-Location MIME header field if the PidTagAttachContentLocation property contains a value that is a valid URI. This value SHOULD be copied to the value of the Content-Location header field.

MIME writers SHOULD generate a Content-Base MIME header field if the PidTagAttachContentBase property contains a value that is a valid absolute URI. This value SHOULD be copied to the value of the Content-Base header field.

2.1.4.2.4 Content-Transfer-Encoding, MIME Part Body

Server SHOULD use base64 (see [RFC4648]) as encoding for all ordinary file attachment MIME part bodies. According to [RFC2045], this also means that server SHOULD correspondingly generate Content-Transfer-Encoding MIME header field, and set its value to "`base64`".

MIME writers MUST use the value of the PidTagAttachDataBinary property to generate the MIME entity body for this attachment. If the property does not exist or has 0 length, an empty MIME entity body SHOULD be generated.

2.1.4.3 MacBinary Attached Files

For interoperability with Macintosh-based mail clients, sometimes it is useful to encode message attachments in MIME using one of the following content-types that are recommended for use in Macintosh environment:

- application/applefile, as specified in [RFC1740]
- application/mac-binhex40, as specified in [RFC1741]
- multipart/appledouble, as specified in [RFC1740]

MIME writers SHOULD generate multipart/appledouble, as this MIME type is recommended by [RFC1740] for use in most cases.

As described in [RFC1740], multipart/appledouble MIME part contains two sub-parts: a header part, with content-type of "`application/applefile`", and data part, that contains actual file data (with content-type set to the value that is corresponding to actual MIME type of the file encoded).

To trigger encoding of an attachment object as multipart/appledouble, clients MUST set property values on the attachment object as follows:

1. The value of the PidTagAttachMethod property MUST be0x00000001 (file attachment).

2. The value of the PidTagAttachEncoding property MUST be the following byte string (expressed in hexadecimal): %x2A.86.48.86.F7.14.03.0B.01.
3. The attachment content, which is the value of the PidTagAttachDataBinary property, MUST be encoded in MacBinary format as specified in [MacBin].

MacBinary is a way of serializing all attributes of a Macintosh file, including both data and resource forks, into a single stream. MacBinary format is described in [MacBin] and the elements relied upon in this specification are summarized very briefly by the following two tables. What follows is intended to specify Exchange behavior with respect to MacBinary data; it is not normative with respect to the MacBinary format itself.

MacBinary data field	Length	Description
MacBinary header	128 bytes	See more detail below
Secondary header data	Length is specified in bytes 120:121 of MacBinary header.	SHOULD be ignored by MIME writers <6>
Data fork	Length is specified in bytes 83:86 of MacBinary header; begins on an even multiple of 128 bytes.	Contents of the file
Resource fork	Length is specified in bytes 87:90 of MacBinary header; begins on an even multiple of 128 bytes.	Resources associated with the file
Get Info comment	Length is specified in byte 99 of MacBinary header.	SHOULD be ignored by MIME writers <7>

Byte offset and length	Value
Byte 0	Old version number, MUST be zero

Byte 1	Length of file name, MUST be less than 64
Bytes 2 : 64	File name, in us-ascii charset; characters beyond the length specified in byte 1 MUST be ignored
Byte 65 : 68	File type, signed integer
Byte 69 : 72	File creator, signed integer
Byte 74	Pad, MUST be 0
Byte 82	Pad, MUST be 0
Bytes 83 : 86	Data fork length, signed 32-bit integer in big-endian format
Bytes 87 : 90	Resource fork length, signed 32-bit integer in big-endian format

MIME writers MUST create a MIME entity with a Content-Type value of "`multipart/appledouble`" according to [RFC1740]. MIME writers SHOULD NOT write a "`name`" parameter for the Content-Type header in this MIME part (according to [RFC1740], this parameter is optional).According to [RFC1740], all additional information (other than the file contents) for a file that is to be transmitted using multipart/appledouble MIME content-type, SHOULD be put into a sub-part with content-type application/applefile.

If the attachment object's PidNameAttachmentMacInfo property has a value, MIME writers MUST use it as the body of the application/applefile body part. The value of this property SHOULD be application/applefile data, as specified in [RFC1740] and further described in section 2.2.4.2.2, but containing only the header and resource fork sections.

If the attachment object's PidNameAttachmentMacInfo has no value, MIME writers SHOULD generate the body of the application/applefile body part from the resource fork and header data present in MacBinary structure from PidTagAttachDataBinary property, using the mappings specified in section 2.2.4.2.2.

According to [RFC1740], the actual contents (or data fork) of the file that is to be transmitted using multipart/appledouble MIME content-type MUST be put into a second MIME sub-part under multipart/appledouble.

This MIME part is written out in the same way as in case of ordinary file attachment, with the following exceptions:

1. MIME writers MUST generate this part's MIME body by extracting only the file's data fork from the MacBinary structure in PidTagAttachDataBinary property on the attachment, instead of just using raw data from this property.
2. MIME writers SHOULD copy the value of the PidNameAttachmentMacContentType to the attachment body part's Content-Type header field.

If PidNameAttachmentMacContentType has no value, MIME writers SHOULD write Content-Type: "`application/octet-stream`". An application/octet-stream type SHOULD also be written if PidNameAttachmentMacContentType has one of the following values:

- message/rfc822
- application/applefile
- application/mac-binhex40
- any multipart content-type

2.1.4.4 OLE Attachments

This section describes generation of MIME entities corresponding to OLE attachments. An attachment object is an OLE attachment if its PidTagAttachMethod property is set to 0x00000006.

MIME writers SHOULD generate a MIME part with "`image/jpeg`" for the value of its Content-Type MIME header field to represent an OLE attachment in MIME. MIME writers SHOULD generate a description string for an OLE attachment, using the value of the PidTagDisplayName property, but ensuring that this value ends with "`.jpg`". The description string SHOULD be used as the name parameter of the Content-Type MIME header field, and as the value of the Content-Description MIME header field SHOULD be generated with the same value.

A Content-Disposition header field SHOULD be generated in the same way as for ordinary file attachments, with the following exceptions:

1. Size parameter SHOULD NOT be generated
2. Filename parameter value SHOULD be set to description string (see above)

The rest of MIME part headers SHOULD be generated in the same way as for ordinary file attachments, as specified in section 2.1.4.1.

OLE attachments SHOULD NOT have PidTagAttachDataBinary property set, so MIME part body cannot be generated in the same way as for ordinary file attachments. Instead,

PidTagAttachDataObject property SHOULD be used. This property SHOULD contain a static rendition of OLE object in Windows metafile format, as specified in [MS-WMF]. MIME writers SHOULD use this data to generate a JPEG image representing this OLE object, and generate MIME part body using this image data. If image generation fails, server SHOULD use some default image.

2.1.4.5 Embedded Message Attachments

This section describes generation of MIME entities corresponding to embedded message attachments. An attachment MUST be considered by MIME writers to be an embedded message attachment if the value of its PidTagAttachMethod property is 0x00000005. MIME writers SHOULD generate a MIME entity with content-type header field set to "`message/rfc822`" (no parameters should be generated). No other MIME headers SHOULD be generated. Instead, MIME writers SHOULD use properties of the embedded message to generate a pure MIME representation of this message, exactly as specified for ordinary messages, and use this data as the content of the message/rfc822 MIME entity. This MIME representation SHOULD be generated exactly as specified for ordinary messages, with the following exception: when writing MIME message headers using PS_INTERNET_HEADERS properties as specified in section 2.1.2.4, properties whose names begin with "X-MS-Exchange-Organization-" or "X-MS-Exchange-Forest-", SHOULD NOT be excluded from MIME generation (as they are for ordinary messages).

2.2 MIME Analysis

This section specifies both conversion from pure MIME to message objects, and from TNEF to message objects. The agent performing the conversion is referred to as a "MIME reader" for clarity, since both clients and servers perform this conversion for different protocols.

As a general rule, when data occurs both in MIME and in a TNEF body part, the version found in MIME is to be preferred. The message body is an exception to this rule: the plain text rendering found in MIME SHOULD NOT be used in preference to a richer (HTML or RTF) rendering found in TNEF. As an implementation guideline, MIME readers MAY process the TNEF body part before processing remaining MIME data so that data from MIME simply overwrites conflicting data from TNEF.

2.2.1 Address Elements

Most MIME address elements correspond to a group of four properties in the message object. The MIME address element itself has three parts, display name, comment, and e-mail address, as specified in [RFC2822]. The four properties are DisplayName, EmailAddress, AddressType and EntryId. For a recipient in a message object, the four properties are referred to as the Recipient property group, whose members are

- PidTagDisplayName
- PidTagEmailAddress

- PidTagAddressType
- PidTagEntryId

For other address elements in a message object the four properties are grouped by name. For example, the four properties corresponding to the From header field are

- PidTagSentRepresentingName
- PidTagSentRepresentingEmailAddress
- PidTagSentRepresentingAddressType
- PidTagSentRepresentingEntryId

Collectively these four properties are referred to as "the PidTagSentRepresenting property group."

2.2.1.1 Mapping Internet Address Elements to a Property Group

In general, MIME readers map the three elements of an Internet e-mail address to the four properties as follows. The comment part of the Internet address SHOULD be ignored. Property names are written as "*DisplayName" below to indicate that this algorithm applies to that member of any property group.

- *DisplayName: If the Internet address has a display name part, convert it to a Unicode string, performing decoding as specified in [RFC2047] if required, and write it to this property value. If there is no display name part, use the e-mail address part.
- *AddressType: First check whether the e-mail address was encoded using IMCEA encapsulation (see section 2.1.1.8). If it was, perform de-encapsulation (section 2.2.1.2) to obtain the e-mail address and type, and write the type to this property. Otherwise, write "SMTP" to this property value. If there is no e-mail address part, do not set this property value.
- *EmailAddress: If the Internet address was IMCEA-encapsulated, use the e-mail address obtained by de-encapsulation. Otherwise, convert the entire e-mail address part to Unicode and write it to this property value. If there is no e-mail address part, do not set this property value.
- *EntryId: If there is an e-mail address part, after the de-encapsulation step, perform a lookup against the address book for an entry any of whose proxy addresses matches this address. If an entry is found, construct an address book entry ID from that entry's DN as specified in [MS-OXCDATA]. If no entry is found, construct a one-off entry ID from the display name, address type, and e-mail address property values computed above, according to the one-off entry ID specification in [MS-OXCDATA].

2.2.1.2 Recognizing and De-Encapsulating IMCEA-Encapsulated Addresses

Please refer to section 2.1.1.8 for the specification of IMCEA encapsulation. De-encapsulation SHOULD be attempted only if the domain part of the encapsulated address is recognized as local, or otherwise able to deliver mail to the de-encapsulated address.

An IMCEA-encapsulated SMTP address consists of the following six elements.

1. The literal string "`IMCEA`" in any combination of upper or lowercase letters.
2. The original address type, one or more ASCII characters.
3. A literal hyphen character, U+002D.
4. The encoded original address. Legal characters are upper and lower case ASCII letters, digits, hyphen (U+002D), equal sign (U+003D), underscore (U+005F), and plus sign (U+002B). Any other characters MUST be encoded as plus sign (U+002B) followed by two hex digits.
5. A literal "@" sign, U+0040.
6. The encapsulation domain, such as "example.com".

To identify an e-mail address as IMCEA-encapsulated, it is sufficient to match items 1-3.

To obtain the original e-mail address and type from an encapsulated address,

1. Copy item 2 to the e-mail address type.
2. Extract item 4, the encoded e-mail address.
3. Decode item 4 by replacing any underscore (U+005F) with a forward slash (U+002F), and replacing any sequence of plus sign (U+002B) followed by two hex digits with the single character whose hex value is those two digits.

2.2.1.3 From

To set the value of the PidTagSentRepresenting property group, MIME clients MUST set the From header field value as specified in [RFC2822].

MIME readers MUST set the value of the PidTagSentRepresenting property group to the value of the first e-mail address component of the From header field (which MAY contain multiple e-mail addresses). If the From header field contains multiple addresses, the first address MUST be used; the others MUST be ignored.

When reading TNEF, MIME readers SHOULD use a From header field value specified in MIME in preference to attSentFor attribute or PidTagSentRepresenting values of attMsgProps

attribute specified in TNEF, except for messages attached to a TNEF message, where a MIME header field does not exist.

2.2.1.4 Sender

To set the value of the PidTagSender property group, MIME clients MUST set either the Sender or the From header field value as specified in [RFC2822].

MIME readers MUST set the value of the PidTagSender property group to the value of the Sender header field, if the Sender header field is present in the MIME header. Otherwise, protocol servers SHOULD set the PidTagSender property group to the value of the first [RFC2822] mailbox of the From header field.

When processing TNEF, MIME readers SHOULD use values specified in MIME in preference to attFrom attribute or PidTagSender property group values of attMsgProps attribute specified in TNEF, except for messages attached to a TNEF message, where a MIME header field does not exist.

2.2.1.5 To, Cc, Bcc

To set the value of a Recipient property group, MIME clients MUST set one of the To, Cc, or Bcc header field values as specified in [RFC2822], corresponding to the desired recipient type, as specified by the table below:

PidTagRecipientType value	Recipient type
0x00000001	To
0x00000002	Cc
0x00000003	Bcc

MIME readers MUST add one recipient to the message object for each address in the To, Cc and Bcc header fields. MIME readers MUST map the value of the Recipient property group from address elements as specified in section 2.2.1.1. Clients MAY specify multiple To, Cc, or Bcc header fields, and MIME readers SHOULD process all of them.

MIME readers MUST set the value of PidTagRecipientType property for each recipient row to the value specified by the table above.

When processing TNEF, MIME readers SHOULD use values specified in MIME in preference to the value of attRecipTable attribute specified in TNEF, except for TNEF DSN messages and any messages attached to a TNEF message.

2.2.1.6 Reply Recipients

To set the values of the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties, MIME clients MUST set the Reply-To header field value as specified in [RFC2822]. Note that because Reply-to is an address list and not a single address, the property mapping is not a normal four-property group.

MIME readers MUST set the values of the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties (as specified in MS-OXOMSG) by mapping addresses from the Reply-To header field.

When processing TNEF, MIME readers SHOULD use a Reply-To header field value specified in MIME in preference to PidTagReplyRecipientEntries and the PidTagReplyRecipientNames values of attMsgProps attribute specified in TNEF (except for messages attached to a TNEF message, where the MIME counterpart is not available).

2.2.1.7 Disposition Notification Recipients

To set the value of the PidTagReadReceiptRequested and the PidTagReadReceipt property group, MIME clients MUST set the Disposition-Notification-To header field value as specified in [RFC3798].

MIME readers MUST set the value of the PidTagReadReceiptRequested property to TRUE if the MIME header contains the Disposition-Notification-To header field.

MIME readers MUST map the value of the PidTagReadReceipt property group from the value of the Disposition-Notification-To header field, if the field exists.

When processing TNEF, MIME readers SHOULD use a Disposition-Notification-To header field value specified in MIME in preference to PidTagReadReceiptRequested and PidTagReadReceipt property group values of attMsgProps attribute specified in TNEF (except for messages attached to a TNEF message, where the MIME counterpart is not available).

2.2.1.8 Return-Receipt-To

To set the value of the PidTagOriginatorDeliveryReportRequested property, MIME clients MUST set the [non-standard] Return-Receipt-To header field value.

MIME readers MUST set the value of the PidTagOriginatorDeliveryReportRequested property to TRUE if the message contains the Return-Receipt-To header field. The actual value of the header field is ignored, and receipts will be returned to the sender.

When processing TNEF, MIME readers SHOULD use a Return-Receipt-To header field value specified in MIME in preference to PidTagOriginatorDeliveryReportRequested property value of attMsgProps attribute specified in TNEF (except for messages attached to a TNEF message, where MIME counterpart is not available).

2.2.2 Envelope Elements

Many MIME header fields that map directly to message object properties have string values. Unless otherwise specified, such string values are copied directly. All string values SHOULD be converted to Unicode (UTF-16) before they are copied to property values, and where applicable, the decoding specified in [RFC2047] MUST be applied before generating the Unicode characters.

If there are multiple instances of a header field, MIME readers SHOULD use the first instance to set the value of the corresponding property.

2.2.2.1 MessageID

To set the value of the PidTagInternetMessageId property, MIME clients MUST set the Message-ID header field value as specified in [RFC2822]. MIME readers MUST copy the value of the Message-ID header field to the PidTagInternetMessageId property.

2.2.2.2 Sent time

To set the value of the PidTagClientSubmitTime property, MIME clients MUST set the Date header field value as specified in [RFC2822].

MIME readers MUST set the value of the PidTagClientSubmitTime property to the value of the Date header field, converted to UTC time. Full precision of the Date header field, including seconds, MUST be preserved. If the Date header field is missing or contains an invalid value, MIME readers MUST set the value of PidTagClientSubmitTime property to the current UTC time.

When processing TNEF, MIME readers MUST use a Date header field value specified in MIME in preference to an attDateSent or PidTagClientSubmitTime value specified in TNEF.

2.2.2.3 References

To set the value of the PidTagInternetReferences property, MIME clients MUST write the desired value to a References header field.

MIME readers MUST copy the value of the References header field to the value of the PidTagInternetReferences property. MIME readers MAY truncate the value of the PidTagInternetReferences property if it exceeds 64KB in length.

2.2.2.4 Sensitivity

To set the value of the PidTagSensitivity property to a value other than normal, MIME clients MUST write the desired value to a Sensitivity header field.

MIME readers MUST map Sensitivity header field values to PidTagSensitivity values as follows:

PidTagSensitivity value	Sensitivity header field value
0x00000000	Normal
0x00000001	Personal
0x00000002	Private
0x00000003	Company Confidential

2.2.2.5 Importance

To set the value of the PidTagImportance property, MIME clients SHOULD write the desired value to an Importance header field.

MIME readers MUST map Importance header field values to PidTagImportance values as follows:

Importance header fieldvalue	PidTagImportance value
Low	0x00000000
Normal	0x00000001
High	0x00000002

MIME clients MAY use a Priority, X-Priority, or X-MSMail-Priority header field instead of an Importance header field to set the value of the PidTagImportance property. In that case, MIME readers MUST map the header field values to PidTagImportance values as specified below. However, if an Importance header field is present, MIME readers SHOULD use its value in preference to any of the others.

Priorityheader field value	PidTagImportance value
NonUrgent	0x00000000
Normal	0x00000001
Urgent	0x00000002

X-Priorityheader field value	PidTagImportance value
5	0x00000000
4	0x00000000
3	0x00000001
2	0x00000002
1	0x00000002

X-MSMail-Priorityheader field value	PidTagImportance value

Low	0x00000000
Normal	0x00000001
High	0x00000002

2.2.2.6 Subject

To set the value of the PidTagSubjectPrefix and PidTagNormalizedSubject properties, MIME clients MUST set the Subject header field value as specified in [RFC2822].

MIME readers SHOULD analyze the Subject header field value into a prefix and a normalized subject value, as specified in section 2.2.2.6.1, and then set the values of the PidTagSubjectPrefix and PidTagNormalizedSubject properties, rather than simply setting the value of PidTagSubject. MIME readers MAY truncate the Subject value; the first 255 characters is a typical length restriction.

MIME readers MUST use a Subject header field value specified in MIME in preference to an attSubject or PidTagSubject value specified in TNEF. They SHOULD, however, use PidTagSubjectPrefix and PidTagNormalizedSubject values from TNEF when they match the MIME subject, because of limitations in the subject normalization algorithm of section 2.2.2.6.1.

2.2.2.6.1 Normalizing the Subject

If no values are available for PidTagNormalizedSubject and PidTagSubjectPrefix in the MIME message, protocol servers SHOULD parse the Subject property value and set those values as follows. If the Subject header field value consists of one, two, or three characters (exclusive of colon (U+003A), blank (U+0020), or digits (U+0030 through U+0039)), followed by a colon (U+003A) and any number of blanks (U+0020), then the protocol server SHOULD set the value of PidTagSubjectPrefix to the aforementioned one, two, or three characters appended with a colon and a space (": "), and SHOULD set the value of PidTagNormalizedSubject to the remainder of the Subject header field value beginning immediately after the aforementioned blanks.

2.2.2.7 Conversation Topic

To set the value of the PidTagConversationTopic property, MIME clients MUST write the desired value to a Thread-Topic header field. This value SHOULD be the same as the value of the Subject header field, normalized as specified in section 2.2.2.6.1 to remove any prefix.

MIME readers MUST copy the value of a Thread-Topic header field to the value of the PidTagConversationTopic property. MIME readers SHOULD also use this header field value as a hint to normalize the subject, as specified in section 2.2.2.6.1, if this value matches the tail of the Subject header field value.

2.2.2.8 Conversation Index

To set the value of the PidTagConversationIndex property, MIME clients MUST write the desired value to a Thread-Index header field. The property data type is binary, and protocol clients MUST encode the header field value using base64 encoding as specified in [RFC2045]. The format of the desired value is specified in [MS-OXOMSG].

MIME readers MUST copy the value of a Thread-Index header field to the value of the PidTagConversationTopic property, assuming the base64-encoded text can be successfully decoded to binary data. MIME readers SHOULD ignore a Thread-Index header that does not contain base64-encoded binary data.

2.2.2.9 In-Reply-To Message ID

To set the value of the PidTagInReplyToId property, MIME clients MUST write the desired value to an In-Reply-To header field as specified in [RFC2822].

MIME readers MUST copy the value of an In-Reply-To header field to the value of the PidTagInReplyToId property.

2.2.2.10 ReplyBy Time

To set the value of the PidTagReplyTime property, MIME clients MUST set the Reply-By header field value as specified in [RFC2822].

MIME readers MUST set the value of the PidTagReplyTime property to the value of the Date header field, converted to UTC time.

When processing TNEF, MIME readers MUST use a Reply-By header field value specified in MIME in preference to a PidTagReplyTime value specified in TNEF.

2.2.2.11 Language Properties

To set the value of the PidTagMessageLocaleId property, MIME clients MUST set the Content-Language header as specified in [RFC3282].

MIME readers MUST set the value of the PidTagMessageLocaleId property by extracting the first language tag from the value of the Content-Language header and mapping it to an LCID as specified in [MS-LCID]. MIME readers SHOULD use the value of a Content-Language header field in preference to the value of PidTagMessageLocaleId found in the attMsgProps attribute of a TNEF message.

To set the value of the PidNameAcceptLanguage property, MIME clients SHOULD write an Accept-Language header field with the desired value. MIME clients MAY write an X-Accept-Language header field instead.

MIME readers MUST copy the value of either header field to the value of the PidNameAcceptLanguage property. If both header fields are present, MIME readers MUST use the Accept-Language header field.

2.2.2.12 Categories

To set the value of the PidLidCategories property, MIME clients MUST set the Keywords header field as specified in [RFC2076].

MIME readers SHOULD map the value of a Keywords header field to the value of the PidLidCategories property by splitting the Keywords header field value at each comma (U+0032), trimming whitespace, and storing each keyword as an individual value of the multiple string property.

To prevent conflicts among category schemes in different organizations, MIME readers MAY omit mapping the Keywords header field to the PidLidCategories property.

2.2.2.13 Message Expiry Time

To set the value of the PidTagExpiryTime property, MIME clients MUST set the Expires header field to the desired value.

MIME readers MUST copy the value of the Expires header field to the value of the PidTagExpiryTime property, after converting it to UTC time.

MIME clients MAY use an Expiry-Date header field instead of an Expires header field. Protocol servers MUST use the value of the Expires header field in preference to Expiry-Date, if both header fields are present.

2.2.2.14 Suppression of Automatic Replies

To set the value of the PidTagAutoResponseSuppress property to -1, indicating that all automatic replies to the message are to be suppressed, MIME clients SHOULD write an X-AUTO-Response-Suppress header field with the value "All". MIME clients MAY, instead, write a Precedence header field with any value.

To set the value of the PidTagAutoResponseSuppress property to a more specific value, MIME clients MUST write an X-AUTO-Response-Suppress header field with one or more values selected from the table of section 2.1.2.20

MIME readers MUST map individual elements of an X-Auto-Response-Suppress header field to bits in the value of the PidTagAutoResponseSuppress property according to the table. If both X-Auto-ResponseSuppress and Precedence header fields are present, the

PidTagAutoResponseSuppress property value MUST be 0xFFFFFFFF. If the value of the X-Auto-Response-Suppress header field is other than as specified in the table of section 2.1.2.20, MIME readers SHOULD ignore the entire header field.

2.2.2.15 Content Class

To set the value of the PidNameContentClass property, MIME clients MUST write a Content-Class header field with the desired value.

MIME readers MUST copy the value of a Content-Class header field to the value of the PidNameContentClass property.

MIME readers SHOULD also set the value of the **PidTagMessageClass** property for certain Content-Class header field values as follows, but only if the value of **PidTagMessageClass** would be otherwise set to "`IPM.Note`":

Content-Class header field value	**PidTagMessageClass property value**
"fax"	"IPM.Note.Microsoft.Fax"
"fax-ca"	"IPM.Note.Microsoft.Fax.CA"
"missedcall"	"IPM.Note.Microsoft.Missed.Voice"
"voice-uc"	"IPM.Note.Microsoft.Conversation.Voice"
"voice-ca"	"IPM.Note.Microsoft.Voicemail.UM.CA"
"voice"	"IPM.Note.Microsoft.Voicemail.UM"
Starts with "urn:content-class:custom."	"IPM.Note.Custom.", followed by the value of Content-Class header field, with "urn:content-class:custom." prefix removed

Additionally, if the Content-Class header field value begins with "`InfoPath.`", then MIME readers SHOULD extract a substring from the header field value beginning immediately after that prefix. If this string contains a period character (U+002E), and the first occurrence of this character is not the last one in the string, this string SHOULD be further separated into two substrings. The delimiting period is not included into either one of the substrings.

The first substring SHOULD be additionally checked to match the string format of a GUID string (see [MS-DTYP]). If this check succeeds, the second substring SHOULD be saved as a value of **PidLidInfoPathFormName** property. In addition, the first substring SHOULD be appended to "`IPM.InfoPathForm.`" and written to the value of the **PidTagMessageClass** property.

If a message that is being processed by MIME reader is clear signed or opaque signed, as specified in [MS-OXOSMIME], then the appropriate suffix ("`.SMIME.MultipartSigned`" or "`.SMIME`") SHOULD be appended to the value of PidTagMessageClass.

2.2.2.16 Message Flagging

To set the value of the PidLidFlagRequest property, MIME clients MUST write an X-Message-Flag header with the desired value.

MIME readers MUST copy the value of an X-Message-Flag header to the value of the PidLidFlagRequest property. In addition, when an X-Message-Flag header is present, MIME readers SHOULD do all of the following:

1. Set the value of the PidTagFlagStatus property to 2 (denoting that the message is flagged).
2. Copy the value of the PidTagSubject property to the value of the PidLidToDoTitle property.
3. Set the value of the PidLidTaskStatus property to 0 (denoting that a task is not started).
4. Delete or disregard any existing property values for the following properties:
   PidLidTaskDueDate
   PidLidTaskStartDate
   PidTagFlagCompleteTime
   PidLidTaskDateCompleted
5. Set the value of the PidLidTaskComplete property to false.
6. Set the value of the PidLidPercentComplete property to 0.0.
7. Set the value of the PidTagToDoItemFlags property to 8.

2.2.2.17 List Server Properties

To set the values of list server related properties, MIME clients MUST write header fields as specified in the following table.

Property	Preferred header field name	Alternate header field name

PidTagListHelp	List-Help	X-List-Help
PidTagListSubscribe	List-Subscribe	X-List-Subscribe
PidTagListUnsubscribe	List-Unsubscribe	X-List-Unsubscribe

MIME readers MUST copy header field values to property values as specified in the table.

2.2.2.18 Payload Properties

To set the value of the PidTagAttachPayloadClass or PidTagAttachPayloadProviderGuidString properties, MIME clients SHOULD write an X-Payload-Class and an X-Payload-Provider-Guid header field, respectively. Such header fields SHOULD be written to a MIME entity that will be analyzed as an attachment, as specified in section 2.2.4.

MIME readers MUST copy these header field values to the values of the corresponding properties. MIME readers SHOULD ignore these header fields when they appear on a MIME entity that is analyzed as a message or message body, rather than as an attachment.

2.2.2.19 Classification Properties

To set property values related to message classification, MIME clients MUST write the following header field:

```
X-Microsoft-Classified: true
```

In addition, MIME clients MUST write header field values for all of X-Microsoft-Classification, X-Microsoft-ClassDesc, X-Microsoft-Classification-Guid, and X-Microsoft-Classification-Keep.

When the appropriate X-Microsoft-Classified header field is present, MIME readers MUST map or copy all classification header field values to their corresponding property values as specified in the following table. If the X-Microsoft-Classified header field is missing or has a different value, MIME readers SHOULD NOT set any of the five property values listed in the table.

Classification header field	Classification property	Header value mapping
X-Microsoft-Classified	PidLidClassified	"true" => true
X-Microsoft-ClassKeep	PidLidClassificationKeep	"true" => true "false" => false

X-Microsoft-Classification	PidLidClassification	copy string value
X-Microsoft-ClassDesc	PidLidClassificationDescription	copy string value
X-Microsoft-ClassID	PidLidClassificationGuid	copy string value

2.2.2.20 Unified Messaging Properties

To set the values of unified messaging properties, MIME clients MUST write the desired value to the corresponding header field as specified in the following table.

Header field name	Property
X-CallingTelephoneNumber	PidTagSenderTelephoneNumber
X-VoiceMessageSenderName	PidTagVoiceMessageSenderName
X-AttachmentOrder	PidTagVoiceMessageAttachmentOrder
X-CallID	PidTagCallId
X-VoiceMessageDuration	PidTagVoiceMessageDuration; header value MUST be parsed as PtypInteger32
X-FaxNumberOfPages	PidTagFaxNumberOfPages; header value MUST be parsed as PtypInteger32

MIME readers MUST copy header field values to property values as specified in the table.

2.2.2.21 Content-ID

To set the value of the PidTagBodyContentId property, MIME clients MUST write the desired value to a Content-ID header field on a MIME entity that maps to a message body, as specified in section 2.2.3 of this document.

MIME readers MUST copy the value of a Content-ID header field on such a MIME entity to the value of the PidTagBodyContentId property.

MIME clients MAY write either a Content-ID or a Content-Location header field, but SHOULD not write both on a single MIME entity.

2.2.2.22 Content-Base

To set the value of the PidNameContentBase property, MIME clients MUST write the desired value to a Content-Base header field on a MIME entity that maps to a message body, as specified in section 2.2.3 of this document.

MIME readers MUST copy the value of a Content-Base header field on such a MIME entity to the value of the PidNameContentBase property.

To set the value of the PidNameContentBase property, MIME clients MUST write the desired value to a Content-Base header field on a MIME entity that maps to a message object (top-level or attached).

MIME readers MUST copy the value of a Content-Base header field on such a MIME entity to the value of the PidNameContentBase property.

2.2.2.23 Content-Location

To set the value of the PidTagBodyContentLocation property, MIME clients MUST write the desired value to a Content-Location header field on a MIME entity that maps to a message body, as specified in section 2.2.3 of this document.

MIME readers MUST copy the value of a Content-Location header field on such a MIME entity to the value of the PidTagBodyContentLocation property.

2.2.2.24 XRef

To set the value of the PidNameCrossReference property, MIME clients MUST write the desired value to an XRef header field.

MIME readers MUST copy the value of an XRef header field to the value of the PidNameCrossReference property.

2.2.2.25 PidTag TransportMessageHeaders

MIME readers SHOULD copy all header fields, with certain exceptions, from an inbound message to the value of the PidTagTransportMessageHeaders property. With the exception of headers specifically mentioned in section 2.1.2, headers beginning with the reserved name prefixes "X-MS-Exchange-Organization-" and "X-MS-Exchange-Forest-" SHOULD NOT be copied to PidTagTransportMessageHeaders.

Clients SHOULD NOT set the value of PidTagTransportMessageHeaders. This property value SHOULD be set only by MIME readers upon delivery of a message from SMTP, in

which case it SHOULD be set to the header of the top-level message (with exceptions as already specified).

2.2.2.26 Generic Header Fields in PS_INTERNET_HEADERS

To create a named property in the PS_INTERNET_HEADERS property set, whose name is a header field name and whose value is a header field value, MIME clients MUST create a header field with the desired name and value.

For each such header field, MIME readers SHOULD create a named property as follows.

- The PropertyName Guid is %X86.03.02.00.00.00.00.00.00.C0.00.00.00.00.00.00.46.
- The PropertyName name is the header field name.
- The property value is the header field value. If the header field value was encoded according to [RFC2047], MIME readers MUST decode it.

MIME readers MUST NOT create such named properties for any MIME header field that is mapped to a different property, as specified elsewhere in this section. MIME readers SHOULD NOT create such named properties for any MIME header field whose name is listed here:

- Received
- Resent-From
- Resent-Sender
- Resent-Date
- Resent-Message-Id
- Content-Type
- Content-Disposition
- Content-Description
- Content-Transfer-Encoding
- Content-ID
- Content-MD5
- Mime-Version
- Return-Path
- Comments
- AdHoc
- Apparently-To
- Approved
- Control
- Distribution
- Encoding
- FollowUp-To
- Lines

- Bytes
- Article
- Supercedes
- NewsGroups
- NntpPostingHost
- Organization
- Path
- RR
- Summary
- Trace
- Encrypted
- X-MimeOle
- X-MS-Tnef-Correlator
- Any header field whose name begins with "`X-MS-Exchange-Organization-`" or "`X-MS-Exchange-Forest-`", except for
    - X-MS-Exchange-Organization-AuthAs
    - X-MS-Exchange-Organization-AuthDomain
    - X-MS-Exchange-Organization-AuthMechanism
    - X-MS-Exchange-Organization-AuthSource

2.2.3 Body Text

Unlike MIME, which allows an arbitrary number of inline text body parts, message objects distinguish one text body part as the message body.

2.2.3.1 Client Actions

To send the value of PidTagBody as the definitive body text, MIME clients SHOULD create a MIME message in which the first or only element has "`text/plain`" as the value of the content-type header field, and that element's body contains the text. MIME clients SHOULD specify the charset of the message body text on the corresponding MIME element.

To send the value of PidTagHtml as the definitive body text, MIME clients SHOULD create a MIME message in which the first or only MIME element has "`text/html`" as the value of the Content-Type header field, and that element's body contains a well-formed HTML document. Clients SHOULD generate a multipart/alternative structure with a text/plain representation (see Example 1), so that a greater number of clients can process the message.

To send the value of PidTagRtfCompressed as the definitive body text, MIME clients SHOULD create a MIME message containing a TNEF body part, as specified at the beginning of section 2, and write the desired value of PidTagRtfCompressed into the attMsgProps attribute of the TNEF.

2.2.3.2 Determining Which MIME Element Is the Message Body

Rules a MIME reader MUST follow for selecting a message body are both qualifying, or positive, and disqualifying, or negative. In order to qualify as a message body a MIME entity MUST meet at least one of the following conditions:

1. Content-type header field value is "`text/plain`", "`text/html`", "`text/enriched`", or "`text/calendar`".
2. Content-type header field value is "`multipart/alternative`" and at least one child MIME entity is "`text/plain`", "`text/html`", "`text/enriched`", or "`text/calendar`".
3. Content-type header field value is "`multipart/related`", and its first child MIME entity is either "`text/html`", or "`multipart/alternative`" with at least one text/html child MIME entity.

In order to qualify as a message body a MIME entity MUST NOT have a Content-Disposition header field with the value "`Attachment`".

In all cases, it is the text body part and not the containing multipart itself that is mapped to the message body.

MIME readers MUST select the first MIME entity that qualifies according to the above rules as the message body. MIME readers SHOULD then map the content of the selected MIME entity to a message object property value according to the following rules.

1. If the body MIME entity is a single text/plain, copy its content to the value of the PidTagBody property.
2. If the body MIME entity is a single text/html, copy its content to the value of the PidTagHtml property.
3. If the body MIME entity is a single text/enriched, convert its content to HTML and copy the result to the value of the PidTagHtml property.
4. If the body MIME entity is a single text/calendar, parse the iCalendar document and copy the value of the DESCRIPTION property to PidTagBody. If the DESCRIPTION property is missing, MIME readers MAY use the value of the COMMENT property instead. Please refer to [MS-OXCICAL].
5. If the body MIME entity is multipart/alternative, MIME readers SHOULD select the last child entity that has one of the four eligible types and map it as in rules 1-4. However, if the last child entity is text/calendar and one of the preceding entities is text/html, MIME readers MAY map the text/html instead of the DESCRIPTION property of the text/calendar.

6. If the body MIME entity is multipart/related, identify the first child MIME entity that is either text/html or multipart/alternative and map it according to rules 1-5.

2.2.3.2.1 Selecting the Primary Message Text MIME Element

When alternative text MIME elements are present and eligible for use as the message body, as specified in section 2.2.3.2, MIME readers SHOULD choose a MIME element to populate the message body text using the following ranking of content types:

1. text/html
2. text/enriched
3. text/plain
4. text/calendar (but only if the METHOD property value of the text/calendar body part is PUBLISH, REQUEST, REPLY, or CANCEL)

If text/html is selected then MIME readers MUST copy the MIME element body text to the value of the PidTagHtml property, map the charset parameter of the MIME element's Content-Type header field to a codepage, and set the value of the PidTagInternetCodepage property to that codepage. If the charset parameter is not present, MIME readers MAY use the value of a content-type meta tag in the HTML document, but SHOULD verify its validity before using it.

If text/plain is selected then MIME readers MUST convert the plain text to UTF-16LE and write the resulting text to the value of the PidTagBody property. MIME readers SHOULD, in addition, map the value of the charset parameter of the MIME element's Content-Type header field to a codepage and set the value of the PidTagInternetCodepage property to that codepage.

If text/enriched is selected, then MIME readers MUST convert to either text/plain, text/html or RTF, and handle that as previously specified.

If both text/html and text/calendar body parts are present and eligible for use as message body, then instead of writing text to the PidTagHtml property, MIME readers SHOULD convert the HTML text to RTF and write it to the value of the PidTagRtfCompressed property. Alternatively, MIME readers MAY choose to use plain text from a text/plain body part or from data in the text/calendar body part, as specified in [MS-OXCICAL]. MIME readers MUST NOT set the PidTagHtml property on a calendar or meeting message object.

2.2.4 Attachments

During MIME analysis MIME readers SHOULD classify all non-multipart MIME entities and multipart/appledouble MIME entities (that contain appropriate child MIME sub-parts) into 3 categories:

1. MIME entities that can potentially represent the message body

2. MIME entities that represent non-inline attachments

3. MIME entities that represent attachments that can potentially be inline

All MIME entities that can be classified as attachments (2nd or 3rd category) SHOULD be treated by MIME readers as attachment MIME parts, and an entry in attachment table SHOULD be created for each such MIME part. However, depending on the value of content-type MIME header, analysis SHOULD be done differently:

a) message/rfc822 MIME entities SHOULD be treated as embedded message attachments, as specified in section 2.2.4.3.

b) multipart/appledouble, application/applefile and application/mac-binhex40 MIME entities SHOULD be treated as Macintosh attachments, as specified in section 2.2.4.2.

c) message/external-body attachments SHOULD be treated as external body attachments, as specified in section 2.2.5.

d) All other attachments SHOULD be treated as regular file attachments, as specified in section 2.2.4.1.

If no content-type header field is present on a MIME entity, MIME readers SHOULD treat it as text/plain (unless this MIME entity is a sub-part of multipart/digest, in which case the default value for content-type MIME header field is message/rfc822).

2.2.4.1 Regular File Attachment MIME Part Analysis

When creating an attachment object for a regular file attachment, MIME readers MUST set the value of the PidTagAttachMethod property to 0x00000001.

2.2.4.1.1 File name

Attachment file name SHOULD be determined by MIME readers in the following order:

1. If Content-Disposition header field exists on the attachment MIME entity, and a non-empty filename parameter is available on this header, the filename parameter value SHOULD be used, else

2. If Content-Type header field is available on the attachment MIME entity, and a non-empty name parameter is available on this header, the name parameter value SHOULD be used, else

3. If Content-Transfer-Encoding header field is set to "`binhex`", MIME readers SHOULD try to parse MIME part body as MacBinary structure, as specified in section 2.2.4.2.3. Only first 128 bytes of MIME body (decoded with binhex, see [RFC1741]) SHOULD be parsed. If parsing of MacBinary structure succeeds, file name data from this structure SHOULD be used, else

4. If attachment MIME part body is encoded with uuencode (see section 2.3.1), and contains file name data, this file name SHOULD be used, else

5. If Content-Description header field is available on the attachment and its value is non-empty, it SHOULD be used as file name value for an attachment. (Even if a file name for attachment was found in one of the previous steps, this value SHOULD be written to PidTagDisplayName for attachment.)

MIME readers SHOULD sanitize the resulting file name and display name by removing characters that are not legal in a Windows file name. Invalid characters include:

Description	Code point	Character
Control characters	U+0000 through U+001F	
Double quote	U+0022	"
Forward slash	U+002F	/
Colon	U+003A	:
Left angle bracket	U+003C	<
Right angle bracket	U+003E	>
Pipe	U+0049	|
Backslash	U+005C	\

The following procedure SHOULD then be applied both to attachment file name and display name (if display name is not available, empty string SHOULD be used):

1. Replace all Unicode separator characters with spaces.

2. Separate name into base and extension parts. Extension is defined as trailing part of a name that starts after the last appearance of '.' character (U+002E) in the name, or empty string if name contains no such character.

3. Remove all leading and trailing spaces and leading and trailing '.' (U+002E) characters from both base and extension.

If the extension part of display name is not empty and doesn't match the extension part of file name, it SHOULD be appended to the base part of display name.

If the file name base and/or file name extension is empty, MIME reader SHOULD generate some non-empty string for attachment file name base and/or extension.

After that, if base part of display name is empty, it SHOULD be replaced with base part of file name. Finally, file name base, file extension and display name SHOULD be reassembled from base and extension parts and saved in appropriate properties as follows:

Property	Value
PidTagDisplayName	<display name base>.<file name extension>
PidTagAttachLongFilename	<file name base>.<file name extension>
PidTagAttachExtension	.<file name extension>

The value saved to PidTagAttachLongFilename SHOULD be further processed to form a valid 8.3 file name, and then written to PidTagAttachFilename:

1. Value SHOULD be first separated into name and extension parts, using last '.' character (U+002E) as separator. If no such character is present, or the only appearance of this character is in the beginning of the file name, extension is considered to be empty; separator character itself is not included into name or extension.
2. Replace the following characters with underscore (U+005F):
plus sign '+' (U+002B), comma ',' (U+002C), equals '=' (U+003D), left square bracket '[' (U+005B), right square bracket ']' (U+005D), semicolon ';' (U+003B)
3. Remove the following characters: Space (U+0020), period '.' (U+002E), apostrophe '\'' (U+0027), asterisk '*' (U+002A), question mark '?' (U+003F), as well as characters with UTF8 code greater than 127.
4. If name is empty after removing such characters MIME readers SHOULD generate a non-empty value.
5. Trim name part of the filename to 8 characters, and extension part to 3 characters.
6. If either name or extension was shortened, name part SHOULD be additionally trimmed to 6 characters, and "~1" added to its end.
7. Recombine file name and extension, separated by a single '.' (U+002E).

2.2.4.1.2 Content Type

MIME readers SHOULD save the value of Content-Type MIME header field in PidTagAttachMimeTag property during MIME analysis. The following notes apply for specific values of this header field:

- "`application/ms-tnef`" value SHOULD be replaced with "`application/octet-stream`". This is in the rare case when a TNEF body part is corrupt and cannot be completely processed. Ordinarily a TNEF body part SHOULD NOT be written to an attachment, but analyzed into message object properties and discarded.
- "`application/x-pkcs7-mime`" and "`application/pkcs7-mime`" values: the entire Content-Type header field value, including all parameter names and values, SHOULD be written to the PidNameContentType property value.
- For content-type values starting with "`text/`", if a "`charset`" parameter is present, the parameter value SHOULD be written to the PidTagTextAttachmentCharset property.

2.2.4.1.3 Attachment Creation and Modification Date

If a Content-Disposition MIME header field is present on the attachment MIME part, MIME readers SHOULD use its parameters to set creation and modification dates on the attachment object. If a parameter is missing or its value is not a valid date, the corresponding property value SHOULD NOT be set. Date and time values MUST be translated to UTC time.

Content-Disposition parameter name	Property
creation-date	PidTagCreationTime
modification-date	PidTagLastModificationTime

2.2.4.1.4 Attachment Content-Id, Content-Base, and Content-Location

If a Content-Id MIME header is present on the attachment MIME part, MIME readers SHOULD copy its value to the PidTagAttachContentId property. If this value starts with '<' (U+003C) and/or ends with '>' (U+003E), these characters SHOULD be removed.

If a Content-Location MIME header is present on the attachment MIME part, its value SHOULD be saved in PidTagAttachContentLocation property.

If a Content-Base MIME header is present on the attachment MIME part, its MIME readers SHOULD copy its value to the PidTagAttachContentBase property.

Additionally, if an attachment MIME part is a child of a multipart/related MIME element, and either Content-Id or Content-Location MIME header is present, MIME readers SHOULD mark the message as inline, as specified in section 2.1.4.1.2. MIME readers SHOULD verify

whether the attachment is actually referenced from message body and mark it as inline only if that is the case; but MAY mark it as inline unconditionally.<8>

2.2.4.1.5 Attachment Content-Transfer-Encoding and MIME Part Body

According to [RFC2045], a Content-Transfer-Encoding header might be present on the attachment MIME part. MIME readers SHOULD support following values of this header:

- Base64, see [RFC2045], [RFC4648]
- Quoted-printable, see [RFC2045]
- 7bit, see [RFC2045]
- 8bit, see [RFC3516]
- Binary, see [RFC3030]
- Mac-binhex40
- X-uuencode
- X-uue

According to [RFC2045], if the Content-Transfer-Encoding MIME header is missing, MIME readers MUST behave as if it was set to 7bit.

The attachment's content SHOULD be decoded using the appropriate decoding procedure and saved as the value of the PidTagAttachDataBinary property. MIME readers SHOULD, as a rule, use **RopOpenStream** (as specified in [MS-OXCROPS]) to create this property value.

The encoding values mac-binhex40, x-uuencode and x-uue are non-standard. If a "mac-binhex40" Content-Transfer-Encoding value is encountered, MIME readers SHOULD treat the MIME part body as if it had a Content-Type header field value of "application/mac-binhex40" and process it as specified in section 2.2.4.2.3. However, in the unlikely case of an actual "application/mac-binhex40" Content-Type, MIME readers SHOULD extract only the data fork from the MIME element content and it as the value of the attachment object's PidTagAttachDataBinary property. For X-uuencode and X-uue values, MIME readers SHOULD treat the attachment content as encoded with uuencode (see [IEEE1003.1]). The decoded value SHOULD be written to the attachment object's PidTagAttachDataBinary property value.

2.2.4.1.6 AttachmentContent-ID, Content-Base, and Content-Location

To set the value of the PidTagAttachContentId property, MIME clients MUST write the desired value to a Content-ID header field on a MIME entity that maps to an attachment object, as specified in section 2.2.4of this document.

MIME readers MUST copy the value of a Content-ID header field on such a MIME entity to the value of the PidTagAttachContentId property.

To set the value of the PidTagAttachContentBase property, MIME clients MUST write the desired value to a Content-Base header field on a MIME entity that maps to an attachment object, as specified in section 2.2.4 of this document.

MIME readers MUST copy the value of a Content-Base header field on such a MIME entity to the value of the PidTagAttachContentBase property.

To set the value of the PidTagAttachContentLocation property, MIME clients MUST write the desired value to a Content-Location header field on a MIME entity that maps to an attachment object, as specified in section 2.2.4 of this document.

MIME readers MUST copy the value of a Content-Location header field on such a MIME entity to the value of the PidTagAttachContentLocation property.

2.2.4.2 Apple File Formats

[RFC1740] and [RFC1741] specify using MIME content-types of multipart/appledouble, application/applefile and application/mac-binhex40 to encode files originating from Macintosh OS, to preserve additional data that might be available for these files in that OS. MIME readers SHOULD preserve this additional data for attached files to enable full support of Macintosh-based client applications.

In particular, the attachment object content stored in PidTagAttachDataBinary MUST contain a MacBinary stream as described in [MacBin]. This stream format incorporates both the resource and data forks, as well as certain metadata.

Note, that MIME analysis of application/applefile attachments is specified differently depending on whether or not the application/applefile MIME entity is a sub-part of multipart/appledouble.

2.2.4.2.1 Multipart/Appledouble

A MIME element with content-type multipart/appledouble MIME readers MUST, according to [RFC1740], have two child MIME elements. The "header part" MUST have a content-type of application/applefile; the "data part" MAY have any MIME content-type except application/applefile or another multipart content-type.

As a MIME reader copies data from a multipart/appledouble MIME entity to an attachment object, it MUST analyze the three parts in the following sequence:

1. The data part (typically the first child of multipart/appledouble)
2. The header part (typically the second child of multipart/appledouble)
3. The multipart/appledouble envelope itself

Property values set as a result of MIME header analysis of a particular MIME part MUST overwrite property values set as a result of previous MIME part analysis.

The procedure of header field analysis for any part of a multipart/appledouble MIME part is similar to the procedure for ordinary file attachments specified in section 2.2.4.1, with the following additions:

1. MIME readers MUST set the value of the PidTagAttachMimeTag property to "`multipart/appledouble`".
2. MIME readers MUST set the value of the PidTagAttachEncoding property to the following byte sequence (in hexadecimal): %x2A.86.48.86.F7.14.03.0B.01.
3. MIME readers MUST copy the value of the Content-Type header field on the data part to the value of the PidNameAttachmentMacContentType property.
4. MIME readers SHOULD copy the entire MIME body of the header part to the value of the attachment object's PidNameAttachmentMacInfo property. MIME readers SHOULD also parse this data as AppleSingle structure, specified in [RFC1740], and combine it with the MIME body from the data part to form a MacBinary structure, which SHOULD then be written to the attachment object's PidTagAttachDataBinary property.
5. MIME readers MUST copy file creator and file type information taken from the MacBinary representation of the attachment, to the value of the PidTagAttachAdditionalInformation property with special formatting as follows. The file creator and type fields are both unsigned 32-bit integers in big-endian format.
    a. A single byte, value 0x2E (colon character)
    b. The file creator, encoded by the rule below
    c. A single byte, value 0x2E (colon character)
    d. The file type, encoded by the rule below
    e. A single byte, value 0x00
    f. Encoding is done from the highest-order byte to the lowest-order byte, using the following scheme:
        - Single bytes with values for '\' (%x5C), ':' (%x3A), and ';' (%x3B) are replaced with two-byte sequences '\\' (%x5C.5C), '\:' (%x5C3A), and '\;' (%x%c3B) respectively.
        - Single bytes with values less than 32, greater than 251, or equal to 127 are encoded by a backslash (%x5C) followed by the byte value in octal, padded with zeroes to 3 digits. So for example, a 0x01 byte is encoded as '\001', and 0xFF is encoded as '\377'.

If parsing of the header part fails, MIME readers SHOULD reject the entire message as not MIME compliant.

If the AppleSingle structure from the header part contains a file name for this attachment, it SHOULD be used as the file name only if no file name was found during processing of the MIME headers. <9>

2.2.4.2.2 Application/Applefile

This section specifies MIME analysis for MIME parts with content-type application/applefile which are not sub-parts of a MIME part with content-type multipart/appledouble.

The procedure of MIME header analysis for application/applefile attachments is the same as for the procedure for ordinary file attachments specified in section 2.2.4.1, with one exception:

- MIME reader MUST set the value of the PidTagAttachMimeTag property to "`application/applefile`".

Processing of MIME content SHOULD include parsing AppleSingle structure, defined in [RFC1740]. MIME readersSHOULD use the data from this structure to fill PidTagAttachDataBinary property and PidNameAttachmentMacInfo property with appropriate structures, as specified in section 2.2.4.2.1.

If MIME body data doesn't match the definition of AppleSingle structure (see [RFC1740]), MIME readers MAY choose to try to interpret the body of this MIME part as a MacBinary structure. If this succeeds, MIME readers SHOULD copy the resulting MacBinary structure to the value of the PidTagAttachDataBinary property, and PidNameAttachmentMacInfo SHOULD be filled with appropriate data from the MacBinary structure. The value of the PidNameAttachmentMacInfo property SHOULD be application/applefile data containing only the header and resource fork sections. But if the MIME reader fails to parse the MIME body, the entire message SHOULD be rejected as not MIME compliant.

If AppleSingle or MacBinary structure contains a file name for this attachment, it SHOULD be used only if no file name was found during analysis of the attachment's MIME headers. <10>

This remainder of this section specifies how MIME readers SHOULD map elements from AppleSingle format, which can have content-type of multipart/appledouble, or application/applefile, to MacBinary data in the value of the PidTagAttachDataBinary property.

The general structure of AppleSingle format is described in [RFC1740]. Is short, this data structure contains a header part, followed by some number of entries. Each of these entries is identified by a number (AppleSingleEntryId, unsigned 32 bit integer), which defines the internal structure of its binary data. The value of each AppleSingleEntryId, along with definition of the structure of each entry, is specified by [RFC1740]. Custom entries are also allowed in this format.

The MacBinary structure is described in [MacBin]. It consists of five parts; each part MUST be padded to a 128 byte boundary, and all parts except the header are optional <11>

1. Header
2. Additional header data
3. Actual file data (data fork)
4. Resource fork
5. Comment

The structure of the MacBinary header, with comments on usage of each field by MIME readers, is shown in the following table. All offsets and lengths are in bytes, and all integers use big-endian byte ordering.

Field offset	Field length	Description
0	1	Old version number, MUST be zero
1	1	Length of file name, unsigned byte; MUST be less than 64
2	63	File name, in ASCII; characters beyond the length specified in byte 1 MUST be ignored
65	4	File type data, normally expressed as four characters
69	4	File creator data, normally expressed as four characters
73	1	Finder flags, bits 15:8
74	1	Pad, MUST be 0
75	2	Icon vertical location, unsigned 16-bit integer
77	2	Icon horizontal location, unsigned 16-bit integer
79	2	File's folder ID
812	1	File protected flag, low order bit
82	1	Pad, MUST be 0

83	4	Data fork length, signed 32-bit integer, zero if there is no data fork
87	4	Resource fork length, signed 32-bit integer, zero if there is no resource fork
91	4	File creation date, signed 32-bit integer representing a number of seconds since (or before, if negative) midnight, 01/01/2000, UTC.
95	4	File modification date, signed 32-bit integer representing a number of seconds since (or before, if negative) midnight, 01/01/2000, UTC.
99	2	Comment length, unsigned 16-bit integer, MUST be 0
101	1	Finder flags, bits 7:0
102	4	Signature. MIME reader MUST set this value to "mbin" (%x4d.62.69.6E).
106	1	File name script identifier. MIME reader SHOULD set to 0.
107		Extended finder flags, MIME reader SHOULD set to 0.
108	12	Zero fill
120	2	Secondary header length, MUST be 0.
122	1	MacBinary version number. MUST be set to 130 (0x0102), indicating MacBinary III, when MIME reader creates the MacBinary structure.
123	1	Minimum MacBinary version supported by this structure. MUST be set to 129 (0x0101), indicating MacBinary II, when MIME reader creates the MacBinary structure.
124	2	CRC of previous 124 bytes. MIME writers SHOULD NOT validate this value. MIME readers SHOULD calculate this value SHOULD as CRC of the whole header (126 bytes), with last 2 bytes set to 0.

Bytes 126:127		Zero fill

When processing AppleSingle data, MIME readers MUST map AppleSingle fields to MacBinary fields as specified in the following table.

AppleSingleEntryId and type	MacBinary field	Comment
1, data fork	Bytes 83:86 – length; MacBinary data fork part	This mapping SHOULD only be used by MIME readers in MIME analysis of a stand-alone application/applefile, because (according to [RFC1740]) data fork SHOULD be in a separate MIME part in multipart/appledouble case.
2, resource fork	Bytes 87:90 – length; MacBinary resource fork part	If length is 0, MIME writers SHOULD NOT create this entry in AppleSingle.
3, ASCII string	Byte 1 – length; Bytes 2:64 – ASCII string value (only length bytes used)	File name. Note that MacBinary limits this string to 63 bytes. Excess bytes MUST be truncated.
8, ASFileDates structure, create	Bytes 91:94	File creation date, MIME readers SHOULD map it for AppleSingle to MacBinary conversion.
8, ASFileDates structure, modify	Bytes 91:94	File modification date, MIME readers SHOULD map it for AppleSingle to MacBinary conversion.
8, ASFileDates structure,	None	MIME writers SHOULD set to 0 on conversion to

access		AppleSingle.
8, ASFileDates structure, backup	None	MIME writers SHOULD set to 0 on conversion to AppleSingle.
9, ASFinderInfo structure, ioFlFndrInfo.fdType	Bytes 65:68	File type information
9, ASFinderInfo structure, ioFlFndrInfo.fdCreator	Bytes 69:72	File creator information
9, ASFinderInfo structure, ioFlFndrInfo.fdFlags	Byte 73 – bits 15:8 Byte 101 – bits 7:0	File finder flags word. MIME readers SHOULD map this element for AppleSingle to MacBinary conversion.
9, ASFinderInfo structure ioFlFndrInfo.fdLocation.v	Bytes 75:76	Icon vertical location MIME readers SHOULD map this element for AppleSingle to MacBinary conversion.
9, ASFinderInfo structure, ioFlFndrInfo.fdLocation.h	Bytes 77:78	Icon horizontal location. MIME readers SHOULD map this element for AppleSingle to MacBinary conversion.
9, ASFinderInfo structure, ioFlFndrInfo.fdFldr	Bytes79:80	File folder ID. MIME readers SHOULD map this element for AppleSingle to MacBinary conversion.
9, ASFinderInfo structure, ioFlXFndrInfo	None	MIME writers SHOULD fill with zeroes on conversion to AppleSingle.
10, ASMacInfo structure, filler	None	MIME writers SHOULD fill with zeroes on conversion to AppleSingle.
10, ASMacInfo structure, ioFlAttrib, bit 1	Byte 81, low order bit	Protected flag. MIME readers SHOULD map this element for AppleSingle to MacBinary conversion.

Conversion from a **full** AppleSingle structure, found in a stand-alone application/applefile MIME element that is not a child of multipart/appledouble, to a **reduced** AppleSingle structure that SHOULD be used as a child of multipart/appledouble, is done simply by removing the entry with AppleSingleEntryId equal to 1 (the data fork) and adjusting the AppleSingle header accordingly.

2.2.4.2.3 Application/Mac-binhex40

This section specifies MIME analysis for MIME parts with content-type application/mac-binhex40, as specified in [RFC1741].

The procedure of MIME header analysis for application/mac-binhex40 attachments is the same as for the procedure for ordinary file attachments specified in section 2.2.4.1, with the following exceptions:

1. MIME readers MUST set the value of the PidTagAttachMimeTag property to "`application/mac-binhex40`".
2. The value of the Content-Transfer-Encoding header field MUST be ignored. MIME readers MUST use BinHex decoding, as specified in [RFC1741], instead.

Processing of the MIME body SHOULD include parsing a MacBinary structure from the decoded content, as specified in [MacBin]. MIME readers SHOULD use the header and resource fork data from this structure to fill PidNameAttachmentMacInfo property with appropriate data, as specified in section 2.2.4.2.1. The entire decoded MIME body SHOULD be written to the value of the PidTagAttachDataBinary property.

If parsing of the MacBinary data fails, the entire message SHOULD be rejected by the MIME reader as not MIME compliant.

MIME readers SHOULD copy the attachment file name extracted from the MacBinary structure to the value of PidTagAttachFilename, but only if no file name was found during analysis of the MIME headers. <12>

2.2.4.3 Attached Messages

If an attachment MIME part has its content-type MIME header set to message/rfc822 (or no content-type header is present, and this MIME part is a sub-part of multipart/digest MIME part), MIME readers SHOULD treat this attachment as embedded message attachment, and set the value of the attachment object's PidTagAttachMethod property to 5.

MIME analysis for MIME headers SHOULD be performed by server in the same way as for ordinary file attachments, with the following exception: procedure of extracting display name and file name for attachment is different.

The display name for embedded message attachments is extracted from MIME part header fields in the following order:

1. If a Content-Type MIME header field is available on the attachment MIME part, and non-empty name parameter is available on this header, its value SHOULD be used, else
2. If a Content-Disposition MIME header field is available on the attachment MIME part, and non-empty filename parameter is available on this header, its value SHOULD be used, else
3. If a Content-Description MIME header field is available on the attachment, and its value is non-empty, it SHOULD be used, else
4. If a Subject header field is available on the attachments, and its value is non-empty, it SHOULD be used, else
5. the MIME reader SHOULD generate a name of its choosing.

The resulting value SHOULD be written to the PidTagDisplayName property on the attachment, and then processed further to obtain a valid file name:

6. All Unicode separator characters in file name SHOULD be replaced with space characters.
7. All trailing and starting space and '.' characters SHOULD be removed.
8. The file name is then separated into base and extension parts. To do this, server SHOULD look for the last occurrence of any of these characters:
   backslash, "\\", U+005C
   forward slash, "/", U+002F
   colon, ":", U+003A
   period, ".", U+002E

   If a "." (U+002E) character is the last one found, then the part of the filename that precedes this character is considered to be base, and the rest is considered to be extension. In all other cases, extension is considered to be empty string, and base part is considered to be the same as whole file name.

The resulting file name value SHOULD be written to the PidTagAttachLongFilename property, and the resulting extension value SHOULD be saved in PidTagAttachExtension property. The file name SHOULD then be processed further to obtain a valid 8.3 file name:

9. Value SHOULD be first separated into base and extension parts, using last "." character as separator (if no such character is present, extension is considered to be empty; separator character itself is not included into name or extension).
10. "+", ",", "=", "[", "]", ";" characters SHOULD be replaced with "_" character.
11. Space, ".", "\",”\”, "*", "<", ">", "?", ":", "|" characters, as well as characters with UTF8 code greater than 127, SHOULD be removed

12. If base becomes empty string, some default non-empty value SHOULD be used

13. Base part of the file name SHOULD be trimmed to 8 characters, extension part – to 3 characters

14. If either name of extension changed, base part SHOULD be additionally trimmed to 6 characters, and "~1" SHOULD be added to its end

15. File name is saved in the <base>.<extension> format.

The resulting file name SHOULD be written to PidTagAttachFilename.

MIME part body for this attachment SHOULD be used for further MIME analysis that SHOULD result in assigning values to the properties of embedded message from this attachment. This MIME analysis is performed in a way similar to ordinary MIME messages, with the following exceptions:

1. X-MS-Exchange-Organization-Original-Sender MIME header value SHOULD be saved in PidTagQuarantineOriginalSender property, if present

2. Unknown MIME headers, starting with "X-MS-Exchange-Organization-" or "X-MS-Exchange-Forest-", SHOULD NOT be excluded from analysis.

3. After MIME analysis is done for the embedded message, PidTagMessageFlags property SHOULD be modified: IsDraft flag (0x8) SHOULD be removed, and IsRead flag (0x1) SHOULD be set.

2.2.5 External Body Attachments

Attachment MIME parts with content-type MIME header field set to "`message/external-body`" SHOULD be analyzed in the same way as ordinary file attachments, with exceptions as described below.

If the content-type MIME header field has no access-type parameter, or if the value of that parameter is not "`anon-ftp`", MIME readers SHOULD save the entire MIME part in PidTagAttachDataBinary property on the attachment.

Otherwise, the following differences in MIME analysis apply:

- Different file name extraction logic SHOULD be applied, see below.
- Server SHOULD ignore the MIME part body. Instead, a specially formatted URL data string is saved in PidTagAttachDataBinary property in ASCII format, as specified below.
- In this case MIME readers expect the name, site, directory and mode parameters to be present in the Content-Type MIME header. Clients SHOULD NOT create MIME that doesn't meet this criteria.

The URL data string to save in PidTagAttachDataBinary property is constructed using the following template:

"[InternetShortcut]\r\nURL=ftp://"	site parameter value	"/"	directory parameter value	"/"	name parameter value	";type=a", if mode parameter is set to "ascii", ";type=i", if it is set to "image", empty string otherwise

File name extraction logic is similar to the one for ordinary file attachments, with the following exceptions:

1. MIME readers MUST use the name parameter value from the Content-Type MIME header as a value of attachment file name. The file extension (a part of file name after the last appearance of '.' character) SHOULD be replaced with "url"; if original file name has no extension, ".url" SHOULD be added at the end of file name string.
2. The sanitizing logic specified in section 2.2.4.1.1 SHOULD NOT be applied in this case.
3. The file name value constructed in Step 1SHOULD be used as attachment display name as well.
4. All additional filtering logic specified in section 2.2.4.1.1 still applies in this case.
5. The procedure of calculating a value for PidTagAttachFilename property is the same as for embedded message attachments.

2.3 Additional Content Types

2.3.1 Analysis of Non-MIME Content

Internet message content that lacks a MIME-Version header field MAY still be supported by MIME readers. The absence of a MIME-Version header field makes the payload of SMTP or elsewhere non-MIME, with different behavior for inline attachments; header fields such as Content-Type, Content-Disposition and such have no special meaning. MIME readers MAY, nevertheless, assume the presence of a MIME-Version field and treat header fields such as Content-Type in the usual way.

Here is an example of such a message.

```
From: <user1@example.com>
To: <user2@example.com>
Subject: Example Legacy 822 message with attachment.
Date: Mon, 10 Mar 2008 14:36:46 -0700

this is a test message

begin 664 Flag.png
MB5!.1PT*&@H````-24A$4@```!0````-"`8```"I4$Y>````!F)+1T0`_P#_
M`/^@O:>3````"7!(67,```L3```+$P$`FIP8````!W1)344'V`,+`!8G&XK)
MCP```1M)1$%4.,NMDTUJPE`4A;\7\Z,ET8$-07`@BE,1.Q)T#2*X!-V+6^@J
M7(2.Q14X<J+6@29$DZBOHTJA)GW0GO&Y'^=>SA6\O4O^49J*R;4#7#OX.W#2
MGU,I'6EZ6YK>EDKIR*0_SP2*M)6=_(6"D1!$%F%L`O!BQMA6Q#DQ\"]Y]82M
MZH919\G.=QXP@#`VV?D.H\Z25G6CGM#(W4ANN<S5TCP_$FI",AW.L/1K*LS2
MKTR',S0AU1+FM#NW>W8!TCQ/IQKNGG%OD0H;]Q8TW+WZ#0M&@A`2VXKX"&SN
M4CS.\6H'!)&%E()S8J@E/"<&86PR:*^HE0]TZVNZ]36U\H%!>T48FT]AF3W\
F+J]X`F![*O[Z*;K*.ZF`OO0)9G1H4-$`H@T`````245.1*Y"8((`
`
end

begin 664 Flag.png
MB5!.1PT*&@H````-24A$4@```!0````-"`8```"I4$Y>````!F)+1T0`_P#_
M`/^@O:>3````"7!(67,```L3```+$P$`FIP8````!W1)344'V`,+`!8G&XK)
MCP```1M)1$%4.,NMDTUJPE`4A;\7\Z,ET8$-07`@BE,1.Q)T#2*X!-V+6^@J
M7(2.Q14X<J+6@29$DZBOHTJA)GW0GO&Y'^=>SA6\O4O^49J*R;4#7#OX.W#2
MGU,I'6EZ6YK>EDKIR*0_SP2*M)6=_(6"D1!$%F%L`O!BQMA6Q#DQ\"]Y]82M
MZH919\G.=QXP@#`VV?D.H\Z25G6CGM#(W4ANN<S5TCP_$FI",AW.L/1K*LS2
MKTR',S0AU1+FM#NW>W8!TCQ/IQKNGG%OD0H;]Q8TW+WZ#0M&@A`2VXKX"&SN
M4CS.\6H'!)&%E()S8J@E/"<&86PR:*^HE0]TZVNZ]36U\H%!>T48FT]AF3W\
F+J]X`F![*O[Z*;K*.ZF`OO0)9G1H4-$`H@T`````245.1*Y"8((`
`
end
```

MIME writers SHOULD NOT generate messages in this format; MIME SHOULD be generated instead. MIME readers SHOULD analyze messages in this format into header fields, plain-text body, and attached files (possibly including a TNEF attachment).

2.3.2 Message/Partial

The message/partial content type is not supported. MIME readers MUST reject messages containing MIME entities with a message/partial content-type header field. This is to prevent virus scanning from being defeated by splitting up attachment content.

2.3.3 Multipart/Digest

This content type is treated exactly as multipart/mixed, except that the assumed Content-Type for body parts with no Content-Type header field SHOULD be message/rfc822 rather than text/plain.

3 Structure Examples

This example illustrates a very simple e-mail message in both MIME and message object formats. First, here is the message in MIME format.

```
Received: from mailer01.example.com by mailer02.example.com
  with Microsoft SMTP Server; Mon, 11 Feb 2008 14:45:44 -0800
From: <user1@example.com>
To: <user2@example.com>; <user3@example.com>
Subject: test message
Date: Mon, 11 Feb 2008 14:45:32 -0800
Message-ID: <000001c86cff$cf0dd670$ae62379d@mail.example.com>
MIME-Version: 1.0
Content-Type: text/plain
Content-Transfer-Encoding: 7bit
Importance: normal
Priority: normal

this is a test message
```

Represented as a message object, the simple message above would look something like this. The message itself has several properties, and it contains recipients each with several properties of its own; the recipients are shown as a separate table here.

Message Property	Type	Value

PidTagMessageDeliveryTime	PtypTime	%xF9.2D.82.D6.FF.6C.C8.01
PidTagSentRepresentingName	PtypString	Test user 1
PidTagSentRepresentingAddressType	PtypString	SMTP
PidTagSentRepresentingEmailAddress	PtypString	user1@example.com
PidTagSubject	PtypString	test message
PidTagClientSubmitTime	PtypeTime	%x00.8E.AD.CE.FF.6C.C8.01
PidTagInternetMessageId	PtypString	<000001c86cff$cf0dd670$ae62379d@mail.example.com>
PidTagImportance	PtypInteger32	1
PidTagPriority	PtypInteger32	0
PidTagBody	PtypString	this is a test message
PidTagInternetCodepage	PtypInteger32	28591
PidTagObjectType	PtypInteger32	5
PidTagMessageFlags	PtypInteger32	0x23

Row ID	Recipient Property	Type	Value
38714304	PidTagDisplayName	PtypString	Test user 2

38714304	PidTagAddressType	PtypString	EX
38714304	PidTagEmailAddress	PtypString	/O=Example1/OU= Administrative Group/cn=Recipients/cn=user2
38714304	PidTagSmtpAddress	PtypString	user2@example.com
38714304	PidTagRecipientType	PtypInteger32	1
38714304	PidTagObjectType	PtypInteger32	6
38714305	PidTagDisplayName	PtypString	Test user 3
38714305	PidTagAddressType	PtypString	EX
38714305	PidTagEmailAddress	PtypString	/O=Example1/OU= Administrative Group/cn=Recipients/cn=user3
38714305	PidTagSmtpAddress	PtypString	user3@example.com
38714305	PidTagRecipientType	PtypInteger32	1
38714305	PidTagObjectType	PtypInteger32	6

While obviously less compact than the MIME format, the message object format makes strongly typed data available. Both client and server code can sort, find, and process messages according to specific criteria such as "all unread messages," "all messages tagged as Personal," or "all calendar items occurring in the week of 2/12/2008, sorted by start time."

4 Security Considerations

4.1 Unsolicited Commercial E-mail (Spam)

A significant business has evolved around the sending of Unsolicited Commercial E-mail (colloquially referred as SPAM). Unlike physical bulk mail where there exist built-in restrictions on who can send and what they have to pay, the general structure of SMTP allows anonymous sources to send e-mail virtually without restriction. Attempts are

being made to reduce the volume of it making it to a person's mailbox, but care has to be taken to not affect legitimate senders.

Part of the success of this industry is the fact that people impute importance to unverifiable things (see [MS-OXCSPAM]). For example the purported sender of a piece of mail (considering most mail is not digitally signed), is commonly used by people to attach importance and priority. If the mail appeared to come from that person's boss there is a higher probability that the employee would act on the message. In this case care SHOULD be taken when receiving mail over unauthenticated transports that just because the routing address of the sender matches a valid employee or contact that the address stored on the message object remain as the external routing address. Thus not replaced with its Address Book equivalent and conveying more importance to the content.

4.2 Information Disclosure

Content that is sent can contain hints about the source network's topology and structure. In MIME this can be discerned from the Received headers (every SMTP server and potentially the client's machine are listed by its network address). In addition, if the optional algorithm offered in [RFC2822 section 3.6.4] to generate a unique Message-Id header value is implemented, the client's network address and internal domain name is exposed. Alternately a GUID can be used to satisfy the unique identifier, circumventing the first data exposure. The aforementioned problem also exists for Content-Id header and Boundary parameter values. It is suggested that a GUID be used here as well.

Additionally when sending recipient data other than the properties outlined above, implementations SHOULD be aware that internal data can be exposed. For example office numbers and phone numbers could be cached on each recipient. Where this is an issue is on embedded messages which are transported via TNEF (see [MS-OXTNEF]) as it has the ability to carry more recipient information than is available with MIME headers.

Care SHOULD also be taken when receiving mail to deal with some information disclosure issues. If the e-mail message leverages any feature that requires the client to "fetch" additional resources when displaying it, the act of fetching can expose a) that the recipient is an actual employee and b) the date and time that it was read. Examples of this are external bodies (see below), HTML stylesheets, and images.

4.3 Content-Type Versus File Extension Mismatch

Various clients accept the Content-Type received from the server but then verify the content. This can include checking the file extension or looking for a thumbprint at the start of the file and then mapping this data back to a verified Content-Type. If the file extension or thumbprint does not match the stated Content-Type, a Content-Type value derived from the file extension or thumbprint is used instead. This behavior is actually codified in [RFC 1521], which allows the sender to set the Content-Type to

"`application/octet-stream`" (or not set it at all). The recipient is then responsible for correctly determining the type of content via alternate means.

In addition it was found that various clients incorrectly set the Content-Type either by mistake or intentionally. Support to address the former has existed for quite some time but has opened a path to potentially thwart, policy scanning and protection applications running on the server.

Therefore MIME readers MAY want to correct mislabeled Content-Type header values so that server Policy Agents and clients can trust its value. Clients SHOULD offer a mechanism to

a) Suppress correcting the Content-Type header,

b) Block attachments by type or extension, and/or

c) Offer some sort of security barrier before running any script, or binary.

The previous is particularly important if the sender is unauthenticated.

4.4 Do Not Support Message/Partial

A Content-Type of "message/partial" allows large messages to be sent in pieces and re-assembled by the client. It was originally designed to work around transmission failures during slow delivery causing the complete message to be resent from scratch, and to work around message size restrictions of implementations of protocols like SMTP. With increased bandwidth speeds, and greater connectivity, the long transmission times are more a thing of the past. Continued support for this Content-Type allows an avenue for content that is inappropriate to reach (or leave) the e-mail client's computer. This could include things such as "Information disclosure" of proprietary information, unsolicited bulk mail (SPAM), and computer virus attachments.

E-mail servers attempt to protect their users from inappropriate content by implementing Policy applications that run as part of the protocol. For them to work efficiently, the complete content MUST be incorporated into one message. For this reason servers SHOULD prohibit sending or receiving messages with a Content-Type of "message/partial".

4.5 Considerations for Message/External-Body

The original MIME RFC's [RFC1521] allowed the body of an entity to be referenced externally versus requiring it to be inline. The current RFC [RFC2046 section 5.2.3] documents the form of this construct but the security implications are as follows.

1. The blind retrieval of the content by the client can disclose information about the recipient (see above).

2. The authentication mechanism tied to the retrieval (Access-Type parameter) can result in a Pop-up dialog box, leading the user to expose credential information.

3. The server (Policy or delivery application) attempting to check the content opens up a Denial of Service vector for the remote host to tie up server resources.

4.6 Preventing Denial of Service Attacks

4.6.1 Submission Limits

Servers MAY limit the size of received messages to limit resource consumption. Such limits can be different for authenticated versus anonymous senders.

4.6.2 Complexity of Nested Entities

It is possible to represent very complex hierarchies with MIME and to add superfluous entity layers (Content-Type: multipart/). Servers and Clients SHOULD protect themselves from stack overflows or heap starvation. This MAY involve limiting the nesting depth of attachments and body parts within a single message.

4.6.3 Number of embedded messages.

A server implementing this protocol converts MIME content into a Message Object representation. This causes each embedded message to be mapped individually and all attachments included therein. One implementation of this is to recursively handle each attached embedded message, but care SHOULD be taken not to encounter a stack overflow by doing so.

4.6.4 Compressed Attachments.

Analyzing each attachment on the server is a concern where decompression is required. It is possible to encounter compressed content that requires large volumes of disk space, memory, or other resources leading to a denial of service.

5 Appendix A: Office/Exchange Behavior

The information in this specification is applicable to the following versions of Office/Exchange:

- Office 2003 with Service Pack 3 applied
- Exchange 2003 with Service Pack 2 applied
- Office 2007 with Service Pack 1 applied
- Exchange 2007 with Service Pack 1 applied

Exceptions, if any, are noted below. Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies Office/Exchange behavior in accordance with the SHOULD or SHOULD NOT prescription.

Unless otherwise specified, the term MAY implies Office/Exchange does not follow the prescription.

<1> Section 2.1.3.2: This is for compatibility with the Exchange 2003 SP2 TNEF reader, which loses body text if only a plain text body is encoded in TNEF.

<2> Section 2.1.3.6: Exchange 2007 SP1 does not check whether an apparently inline attachment is, in fact, referenced from the message body.

<3> Section 2.1.4.1: Exchange 2007 SP1 does not exclude attached message objects or attachments to plain-text messages from being considered inline.

<4> Section 2.1.4.1.1: Exchange 2007 SP1 does not exclude non-OLE attachments in an RTF message from being considered inline.

<5> Section 2.1.4.1.2: The MIME writer in Exchange 2007 SP1 only checks condition 1, not 2 or 3, when classifying attachments as inline.

<6> Section 2.1.4.3: Exchange 2007 SP1 does not ignore the presence of secondary header data in a MacBinary stream; it fails if secondary header data is present.

<7> Section 2.1.4.3: Exchange 2007 SP1 does not ignore the presence of additional data in a MacBinary stream; it fails if additional data is present.

<8> Section 2.2.4.1.4: The MIME reader in Exchange 2007 SP1 does not verify that inline attachment candidates are in fact referenced from the message body before marking them as inline.

<9> Section 2.2.4.2.1: Exchange 2007 SP1 gets the attachment file name from AppleSingle data, instead of preferring a file name found in MIME headers.

<10> Section 2.2.4.2.2: Exchange 2007 SP1 gets the attachment file name from AppleSingle or MacBinary data, instead of preferring a file name found in MIME headers.

<11> Section2.2.4.2.2: Exchange 2007 SP1 does not support a MacBinary structure with additional header data or comment part present.

<12> Section 2.2.4.2.3: Exchange 2007 SP1 gets the attachment file name from MacBinary data, instead of preferring a file name found in MIME headers.

6 Index