[MS-OXORMMS]:
Rights-Managed E-Mail Object Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Preliminary Documentation. This Open Specification provides documentation for past and current releases and/or for the pre-release (beta) version of this technology. This Open Specification is final

documentation for past or current releases as specifically noted in the document, as applicable; it is preliminary documentation for the pre-release (beta) versions. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. As the documentation may change between this preliminary version and the final version of this technology, there are risks in relying on preliminary documentation. To the extent that you incur additional development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability.
04/25/2008	0.2		Revised and updated property names and other technical content.
06/27/2008	1.0		Initial Release.
08/06/2008	1.01		Revised and edited technical content.
09/03/2008	1.02		Updated references.
12/03/2008	1.03		Updated IP notice.
03/04/2009	1.04		Revised and edited technical content.
04/10/2009	2.0		Updated applicable product releases.
07/15/2009	3.0	Major	Revised and edited for technical content.
11/04/2009	4.0.0	Major	Updated and revised the technical content.
02/10/2010	5.0.0	Major	Updated and revised the technical content.
05/05/2010	6.0.0	Major	Updated and revised the technical content.
08/04/2010	6.1	Minor	Clarified the meaning of the technical content.
11/03/2010	6.2	Minor	Clarified the meaning of the technical content.
03/18/2011	6.2	No change	No changes to the meaning, language, and formatting of the technical content.
08/05/2011	6.2	No change	No changes to the meaning, language, or formatting of the technical content.
10/07/2011	6.2	No change	No changes to the meaning, language, or formatting of the technical content.
01/20/2012	7.0	Major	Significantly changed the technical content.
04/27/2012	8.0	Major	Significantly changed the technical content.
07/16/2012	8.1	Minor	Clarified the meaning of the technical content.

Table of Contents

1 Introduction

The Rights-Managed E-Mail Object Protocol is used by the client to create and consume a **rights-managed e-mail message**, which is used to protect e-mail content from inappropriate access, use, and distribution.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

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

code page
flags
handle
little-endian
Unicode

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

Attachment object
binary large object (BLOB)
Hypertext Markup Language (HTML)
mailbox
message body
Message object
metafile
named property
non-Unicode
offline
permission
plain text
plain text message body
property ID
publishing license
recipient
remote operation (ROP)
Rich Text Format (RTF)
rights policy template
rights-managed e-mail message
storage
store
stream
Use License

The following terms are specific to this document:

mapping mode: The way in which logical, device-independent, coordinates are mapped to device-specific coordinates.

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

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the technical documents, which are updated frequently. References to other documents include a publishing year when one is available.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-CFB] Microsoft Corporation, "Compound File Binary File Format".

[MS-DTYP] Microsoft Corporation, "Windows Data Types".

[MS-OFFCRYPTO] Microsoft Corporation, "Office Document Cryptography Structure Specification".

[MS-OXBBODY] Microsoft Corporation, "Best Body Retrieval Algorithm".

[MS-OXCDATA] Microsoft Corporation, "Data Structures".

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

[MS-OXMSG] Microsoft Corporation, "Outlook Item (.msg) File Format".

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

[MS-RMPR] Microsoft Corporation, "Rights Management Services (RMS): Client-to-Server Protocol Specification".

[MS-WMF] Microsoft Corporation, "Windows Metafile Format".

[RFC1950] Deutsch, P., and Gailly, J-L., "ZLIB Compressed Data Format Specification version 3.3", RFC 1950, May 1996, http://www.ietf.org/rfc/rfc1950.txt

[RFC1951] Deutsch, P., "DEFLATE Compressed Data Format Specification version 1.3", RFC 1951, May 1996, http://www.ietf.org/rfc/rfc1951.txt

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

1.2.2 Informative References

[MSDN-DVASP] Microsoft Corporation, "DVASPECT enumeration", http://msdn.microsoft.com/en-us/library/ms690318.aspx

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

[MS-OXCROPS] Microsoft Corporation, "Remote Operations (ROP) List and Encoding Protocol Specification".

[MS-OXGLOS] Microsoft Corporation, "Exchange Server Protocols Master Glossary".

[MS-OXOMSG] Microsoft Corporation, "E-Mail Object Protocol Specification".

[MS-OXORMDR] Microsoft Corporation, "Reminder Settings Protocol Specification".

[MS-OXOTASK] Microsoft Corporation, "Task-Related Objects Protocol Specification".

1.3 Overview

This protocol enables the client to create and consume rights-managed e-mail messages by defining a format for creating and writing an e-mail message with encrypted and compressed content.

When a client creates a rights-managed e-mail message, it encrypts and compresses the contents of the message (body, attachments, and so on) and stores the encrypted, compressed contents as part of the message that is sent to the **recipients (1)**. The client sets certain properties on the message to identify it as rights-managed.

When a client receives a rights-managed e-mail message, it decompresses and decrypts the encrypted **binary large object (BLOB)** and displays the content to the end user if the end user has sufficient **permission** to view the content. In addition, the client disables certain functionality on the rights-managed e-mail message to prevent the recipient (1) from using the message content in an unauthorized manner.

1.4 Relationship to Other Protocols

The Rights-Managed E-Mail Object Protocol relies on the following:

- The Message and Attachment Object Protocol, as described in [MS-OXCMSG], so that the client can obtain a **handle** to the **Message object** and perform property operations on it and handle attachments and perform property operations on **Attachment objects**.
- The E-Mail Object Protocol, as described in [MS-OXOMSG].
- The Rights Management Services (RMS) Client-Server Protocol, as described in [MS-RMPR], to create and consume rights-managed e-mail messages.
- The Compound Binary File Format, as described in [MS-CFB].

1.5 Prerequisites/Preconditions

This protocol assumes that the client has previously logged on to the server and has acquired a handle to the rights-managed e-mail message, as described in [MS-OXCMSG] section 3.1.4.1.

This protocol relies on the RMS Client-to-Server Protocol as described in [MS-RMPR] and therefore assumes that the prerequisites of that protocol are met.

1.6 Applicability Statement

A client can use this protocol to create and consume rights-managed e-mail messages.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The properties specified in this protocol are transported between client and server in the manner specified in [MS-OXCMSG] section 2.1.

2.2 Message Syntax

A rights-managed e-mail message consists of a set of Message object property constraints, including a **Use License**, and an attachment containing an encrypted version of the original message.

The protocol defines several data formats to support rights-managed e-mail messages in addition to those specified in [MS-DTYP].

Unless otherwise specified, rights-managed e-mail Message objects adhere to all property constraints specified in [MS-OXCMSG]. A rights-managed e-mail Message object can also contain other properties, but these properties have no impact on this protocol.<1>

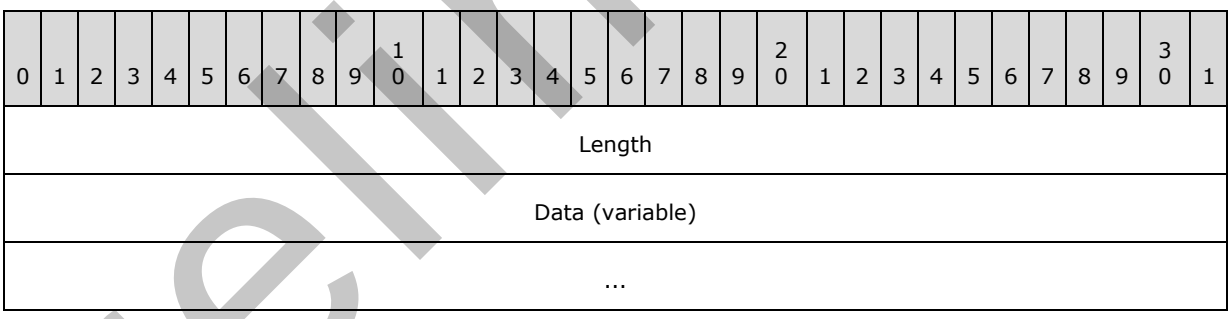
2.2.1 Rights-Managed E-Mail Message Property

The property specified in section 2.2.1.1 is specific to the Rights-Managed E-Mail Object Protocol.

2.2.1.1 PidNameRightsManagementLicense Property

Type: **PtypMultipleBinary** ([MS-OXCDATA] section 2.11.1)

The **PidNameRightsManagementLicense** property ([MS-OXPROPS] section 2.534) is a **named property** that is used to cache the Use License for the rights-managed e-mail message. If the Use License is successfully obtained, this property SHOULD<2> be present on a rights-managed e-mail Message object. If the property is present, the first value of this property MUST contain the compressed Use License for the rights-managed e-mail message. The compression format for the Use License is specified in [RFC1950]. When uncompressed, the resulting data is a length-prefixed **Unicode** string that is formatted as follows.



Length (4 bytes): An unsigned integer that specifies the size of the **Data** field in **WCHAR**s ([MS-DTYP]).

Data (variable): A Unicode string containing the uncompressed Use License data.

2.2.2 Additional Property Constraints

This protocol specifies additional constraints on the property specified in section 2.2.2.1 beyond the constraints specified in [MS-OXCMSG] section 2.2.1.

2.2.2.1 PidNameContentClass Property

Type: **PtypString** ([MS-OXCDATA] section 2.11.1)

The value of the **PidNameContentClass** property ([MS-OXCMSG] section 2.2.1.48) for a rights-managed e-mail message MUST be set to "rpmsg.message".

2.2.3 Attachment Object

A rights-managed e-mail message consists of a wrapper e-mail message with the original e-mail contents encrypted and compressed in an attachment. The attachment that contains the encrypted, compressed contents of the original e-mail message SHOULD be the only attachment on the wrapper e-mail message. If the wrapper e-mail message includes multiple attachments, the attachment that contains the encrypted, compressed contents of the original e-mail message MUST be the first attachment. This attachment has specific property values, as specified in section 2.2.3.1 through section 2.2.3.2, that distinguish it from the other attachments. For details about the procedures for encryption, compression, decompression, and decryption of the original contents in the attachment, see section 3.1.4.

2.2.3.1 PidTagAttachLongFilename Property

Type: **PtypString** ([MS-OXCDATA] section 2.11.1)

The value of the **PidTagAttachLongFilename** property ([MS-OXCMSG] section 2.2.2.10) for a rights-managed e-mail message MUST be set to "message.rpmsg".

2.2.3.2 PidTagAttachMimeTag Property

Type: **PtypString** ([MS-OXCDATA] section 2.11.1)

The value of the **PidTagAttachMimeTag** property ([MS-OXCMSG] section 2.2.2.26) for a rights-managed e-mail message MUST be set to "application/x-microsoft-rpmsg-message".

2.2.4 Data Formats

This protocol references the data formats specified in section 2.2.4.1 through section 2.2.4.5, in addition to several commonly used data types that are specified in [MS-DTYP].

2.2.4.1 LPString Format

The LPString format represents a string that contains a 1-byte positive integer field (**LengthOfString**) indicating the length of the string, followed by Unicode characters whose total length is equal to the **LengthOfString** field. This string is not null-terminated. The length of this string is limited to 255 characters.

2.2.4.2 Non-Unicode LPString Format

The **non-Unicode** LPString format represents a string that contains a 1-byte positive integer (**LengthOfString**) indicating the length of the string, followed by **LengthOfString** non-Unicode characters. This string is not null-terminated. The length of this string is limited to 255 characters.

2.2.4.3 Pipe-Delimited String Format

The pipe-delimited string format represents a Unicode string containing multiple substrings, delimited by the pipe ("|") character. Each substring cannot contain the pipe character. This string

always ends with the pipe character, even if there is only one substring. This is a length-prefixed string with the first byte containing the length of the Unicode characters. The length of this string is limited to 255 characters.

2.2.4.4 Format of the Storage Container

The following figure shows the format of the **storage** container. Some of the streams are optional and will not be present in certain storage containers. The **MailAttachment 0** storage is assumed to be a by value attachment; the format will differ with the attachment type.

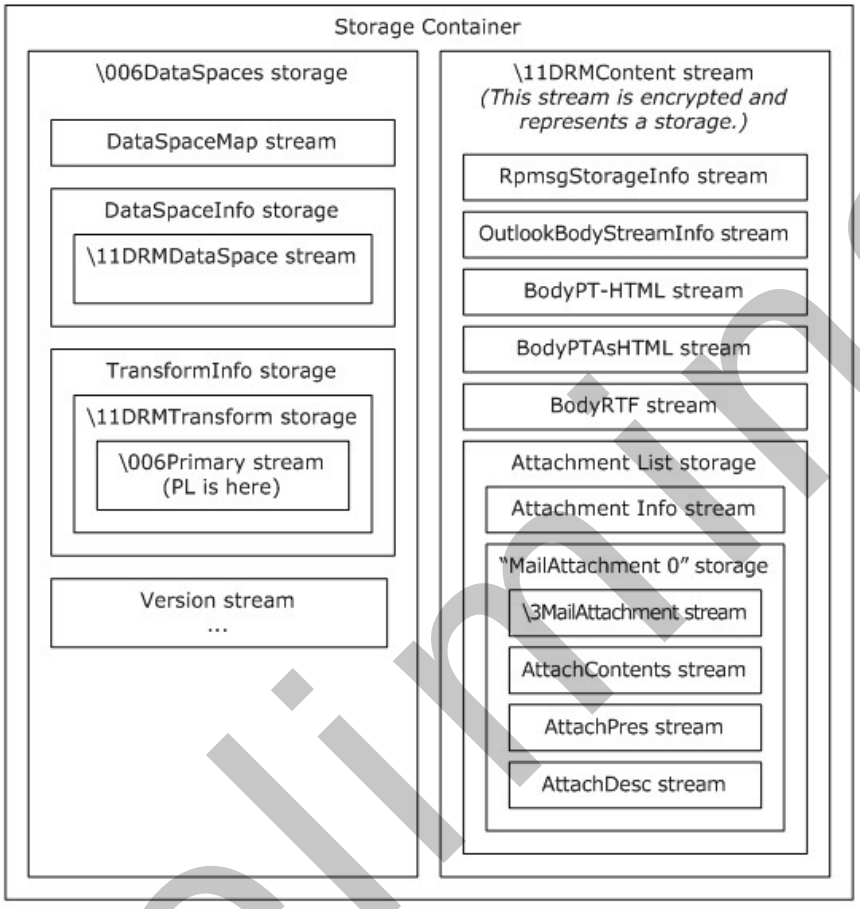


Figure 1: Format of the storage container

The fields listed in the following table MUST be present in the uncompressed storage container.

Stream/storage	Field name	Description	Format
storage	**\006DataSpaces**	Contains data, such as the **publishing license** and transformation information for the document.	Specified in [MS-OFFCRYPTO].

Stream/storage	Field name	Description	Format
stream (1)	**\11DRMContent**	Contains the encrypted **message body (2)** and attachments.	Specified in section 2.2.4.4.1.

2.2.4.4.1 \11DRMContent Storage

The \11DRMContent storage contains the encrypted e-mail message body (2) and attachments. Before encryption, the \11DRMContent storage has the fields specified in the following table.

Field name	Stream/storage	Description
OutlookBodyStreamInfo	This stream (1) MUST be present in the storage.	This stream (1) contains two consecutive values. The first value is of type **WORD** ([MS-DTYP]) and contains the message body (2) format. If the body format is **plain text**, the value MUST be 0x0001. If the body format is **HTML**, the value MUST be 0x0002. If the body format is **Rich Text Format (RTF)**, the value MUST be 0x0003. The second value is of type **DWORD** ([MS-DTYP]). Its value MUST correspond to the value of the **PidTagInternetCodepage** property ([MS-OXCMSG] section 2.2.1.49.6), if present; otherwise, it MUST be set to the active **code page** of the system.
BodyPT-HTML	This stream (1) MUST be present in the storage.	The contents of this stream (1) are based on the body format as specified in the OutlookBodyStreamInfo stream (1). If the body format is plain text, this stream (1) MUST contain the plain text version of the message body (2) that is present in the **PidTagBody** property ([MS-OXCMSG] section 2.2.1.49.1), as specified in [MS-OXBBODY] section 1.3. If the body format is HTML, this stream (1) MUST contain the HTML version of the message body (2) that is present in the **PidTagHtml** property ([MS-OXCMSG] section 2.2.1.49.9), as specified in [MS-OXBBODY] section 1.3. If the body format is RTF, this stream (1) MUST contain an HTML version of the RTF message body (2).
BodyRtf	If the message body (2) format specified in the OutlookBodyStreamInfo this stream (1) is RTF, this stream (1) MUST be present in the storage.	This stream (1) contains the RTF representation of the message body (2) that is present in the **PidTagRtfCompressed** property ([MS-OXCMSG] section 2.2.1.49.4), as specified in [MS-OXBBODY] section 1.3.
BodyPTAsHTML	If the message body (2) format specified in	This stream (1) contains an HTML version of a **plain text message body**. The client

Field name	Stream/storage	Description
	OutlookBodyStreamInfo is plain text, this stream (1) MUST be present in the storage.	MUST ignore this stream (1) on receipt.
RpmsgStorageInfo	This stream (1) MUST be present in the storage.	This stream (1) contains implementation-specific details. It MUST contain the following byte stream (1): 1F 32 DE 15 02 00 00 00 02 00 00 00 00 00 00 00.
WordMailRightsIndex	This stream (1) SHOULD<3> be present in the storage if the message is a reply to a rights-managed e-mail message; otherwise, it MUST NOT be included.	When replying to a rights-managed e-mail message, the user who is replying cannot copy or print the original message included within/below the reply. To differentiate between this protected and unprotected content in saved e-mail messages, the WordMailRightsIndex stream (1) contains first and last character position pairs that bind content within the message. The first pair represents the beginning and end character positions of the original message. The remaining character pairs represent the bounds of the inline comments in the original message. Multiple pairs exist when inline comments are used. The stream (1) MUST contain a **ULONG** ([MS-DTYP]) to represent the number of pairs, followed by the character position pairs. Each pair consists of two character positions, each of type **ULONG**. The values are stored in the **little-endian** format.
Attachment List	This storage MUST be present if the message has any attachment.	This storage contains the attachment storage collection of the message. For details see section 2.2.4.4.2.

2.2.4.4.2 Attachments to the Rights-Managed E-Mail Message

All attachments in the message MUST be stored in the Attachment List storage. The contents of the attachment MUST be encrypted with the same publishing license as the Message object if the associated application supports rights management. In the typical case, the unmodified bits of the attachment are stored in this storage. However, if the file type of the attachment supports rights management, the attachment is also rights-managed. When rights-management protection is applied to attachments, the same issuance license that is used to protect the message is used for the attachment. The structure of each encrypted attachment conforms to the specifications for its file type. To view a rights-managed attachment, the file has to be opened and unencrypted in its native viewer.

The fields of the Attachment List storage are specified in the following table.

Field name	Stream/storage	Description
Attachment Info	This stream (1) MUST be present.	Specified in section 2.2.4.4.3.

Field name	Stream/storage	Description
MailAttachment N	This storage MUST be present.	N represents the attachment number that starts from zero and is incremented with each attachment, as specified in section 2.2.4.4.4.

2.2.4.4.3 Attachment Info

The attachment info stream (1) provides a table of contents for the Attachment List storage. This stream (1) MUST contain the following fields in the order given:

- The **AttachmentCount** field (**ULONG** ([MS-DTYP])), which gives the number of attachments. If this value is 0xFFFFFFFF, the message body (2) format MUST be RTF. The number of attachments in case of RTF messages is in the **DWORD** ([MS-DTYP]) **NumberOfAttachments** field, as specified in the next table.
- The **Pipe-delimited string** field (section 2.2.4.3) containing the list of attachments in the form of "Mail Attachment N", where N represents the attachment number starting from zero. The format of the Unicode pipe-delimited string is as follows:

MailAttachment 0|MailAttachment 1|...|MailAttachment N|

If the message body (2) format specified in the OutlookBodyStreamInfo stream is RTF, the following information is appended to the stream (1). All values are stored in the little-endian format.

Field name	Format	Description
NumberOfAttachments	**DWORD**	Contains the number of attachments in the RTF message.

The following fields are then repeated for each attachment that is present in the RTF message.

Field name	Format	Description
CharacterPosition	**DWORD**	Contains the location in the RTF stream (1) in which the embedded object appears. This corresponds to the **PidTagRenderingPosition** property ([MS-OXCMSG] section 2.2.2.16).
Objf	**DWORD**	Represents the way the contents of an attachment can be accessed. The possible values are as follows. A value of 0x00000001 MUST be matched to the attachment with the value of the **PidTagAttachMethod** property ([MS-OXCMSG] section 2.2.2.9) set to "afOle". A value of 0x00000004 MUST be matched to the attachment with the value of the **PidTagAttachMethod** property set to "afByValue". A value of 0x00000008 MUST be matched to the attachment with the value of the **PidTagAttachMethod** property set to "afEmbeddedMessage". The **Objf** field also has other client-specific **flags** logically combined with a bitwise OR operation to the above values; these valuesare implementation-specific and can be ignored.
Aspect	**DWORD**	Contains the object's draw aspect. If the Objf value is 0x00000004 or 0x00000008, it is set to **DVASPECT_ICON** (as described in [MSDN-DVASP]). If the value of the **Objf** field is 0x00000001, it is set to **DVASPECT_CONTENT**.

Field name	Format	Description
SizeAlongXAxis	**DWORD**	Contains the length of the **metafile** that is displayed in the message body (2). Metrics are based on the **mapping mode** of the metafile.
SizeAlongYAxis	**DWORD**	Contains the height of the metafile that is displayed in the message body (2). Metrics are based on the mapping mode of the metafile.

2.2.4.4.4 MailAttachment Structure

The structure of "MailAttachment N" storage depends on the way the contents of the attachment can be accessed. The different ways are specified by the **PidTagAttachMethod** property ([MS-OXCMSG] section 2.2.2.9). A rights-managed e-mail message MUST allow only the following values for the **PidTagAttachMethod** property:

- afByValue
- afEmbeddedMessage
- afOle

Treatment of each type of attachment is described separately in section 2.2.4.4.4.1 through section 2.2.4.4.4.3.

2.2.4.4.4.1 afByValue

The fields of the "MailAttachment N" storage for the attachment for which the value of the **PidTagAttachMethod** property ([MS-OXCMSG] section 2.2.2.9) is set to "afByValue" are specified in the following table.

Field name	Stream/storage	Description
\3MailAttachment	This stream (1) MUST be present.	Contains the attachment number and reference flag stream (1). The elements of this stream (1) are specified in section 2.2.4.4.4.1.1.
AttachPres	This stream (1) MUST be present.	Stores the attachment's icon, as specified in section 2.2.4.4.4.1.2.
AttachDesc	This stream (1) MUST be present.	Contains information about the attachment, as specified in section 2.2.4.4.4.1.3.
AttachContents	This stream (1) MUST be present.	Contains the actual contents of the attachment, as specified in section 2.2.4.4.4.1.4.

Other streams (1) can be present in the storage, but they are client-specific.

2.2.4.4.4.1.1 \3MailAttachment Stream

The formats of the fields in the \3MailAttachment stream (1) are specified in the following table in the order in which they appear. The values are stored in little-endian format.

Field name	Format	Description
Attachment Number	**DWORD** ([MS-DTYP])	Represents the index of the attachment at the time of attaching. This can be set to 0x00000000. Client ignores this value on receipt.

Field name	Format	Description
Reference Flag	**DWORD**	Contains an implementation-specific flag. This can be set to 0x00000000. Client ignores this value on receipt.

2.2.4.4.4.1.2 AttachPres Stream

The AttachPres stream (1) stores the attachment icon for user presentation in the Windows metafile format, as specified in [MS-WMF].

2.2.4.4.4.1.3 AttachDesc Stream

The AttachDesc stream (1) stores information about the attachment. The following table specifies the format of the fields of the AttachDesc stream (1) in the order in which they appear. Some of the fields contain values of Attachment object properties.

Field name	Format	Description
Stream Version	**USHORT** ([MS-DTYP])	Contains the version. When creating a rights-managed e-mail message, this value MUST always be set to 0x0203. The value is stored in the little-endian format.
Long Path Name	non-Unicode LPString (section 2.2.4.2)	SHOULD contain the value of the **PidTagAttachLongPathname** property ([MS-OXCMSG] section 2.2.2.13) of the attachment, if present; otherwise, it MUST be 0x00.
Path Name	non-Unicode LPString	MUST contain the value of the **PidTagAttachPathname** property ([MS-OXCMSG] section 2.2.2.14) of the attachment, if present; otherwise, it MUST be 0x00.
Display Name	non-Unicode LPString	MUST contain the value of the **PidTagDisplayName** property ([MS-OXCMSG] section 2.2.2.4) of the attachment, if present; otherwise, it MUST be 0x00.
Long File Name	non-Unicode LPString	MUST contain the value of the **PidTagAttachLongFilename** property ([MS-OXCMSG] section 2.2.2.10) of the attachment, if present; otherwise, it MUST be 0x00.
File Name	non-Unicode LPString	MUST contain the value of the **PidTagAttachFilename** property ([MS-OXCMSG] section 2.2.2.11) of the attachment, if present; otherwise, it MUST be 0x00.
Extension	non-Unicode LPString	MUST contain the value of the **PidTagAttachExtension** property ([MS-OXCMSG] section 2.2.2.12) of the attachment, if present; otherwise, it MUST be 0x00.
File Creation Time	64-bit value	MUST contain the value of the **PidTagCreationTime** property ([MS-OXCMSG] section 2.2.2.3) of the attachment, if present; otherwise, it MUST be 0x0000000000000000. This is stored in little-endian format.
File Last Modified Time	64-bit value	MUST contain the value of the **PidTagLastModificationTime** property ([MS-OXCMSG] section 2.2.2.2) of the attachment, if present; otherwise, it MUST be 0x0000000000000000. This is stored in little-endian format.
Attach Method	**ULONG** ([MS-DTYP])	MUST contain the value of the **PidTagAttachMethod** property ([MS-OXCMSG] section 2.2.2.9) of the attachment stored in little-endian

Field name	Format	Description
		format.
Content ID	LPString (section 2.2.4.1)	MUST contain the value of the **PidTagAttachContentId** property ([MS-OXCMSG] section 2.2.2.26), if present; otherwise it MUST be 0x00.
Content Location	LPString	MUST contain the value of the **PidTagAttachContentLocation** property ([MS-OXCMSG] section 2.2.2.26), if present; otherwise, it MUST be 0x00.
Long Path Name	LPString	SHOULD contain the value of the **PidTagAttachLongPathname** property of the attachment, if present; otherwise, it MUST be 0x00.
Path Name	LPString	MUST contain the value of the **PidTagAttachPathname** property of the attachment, if present; otherwise, it MUST be 0x00.
Display Name	LPString	MUST contain the value of the **PidTagDisplayName** property of the attachment, if present; otherwise, it MUST be 0x00.
Long File name	LPString	MUST contain the value of the **PidTagAttachLongFilename** property of the attachment, if present; otherwise, it MUST be 0x00.
File Name	LPString	MUST contain the value of the **PidTagAttachFilename** property of the attachment, if present; otherwise, it MUST be 0x00.
Extension	LPString	MUST contain the value of the **PidTagAttachExtension** property of the attachment, if present; otherwise, it MUST be 0x00.
Image Preview Small	LPString	MUST contain the file name that contains the small image preview of the attachment, if present; otherwise, it MUST be 0x00.
Image Preview Medium	LPString	MUST contain the file name of the medium image preview of the attachment, if present; otherwise, it MUST be 0x00.
Image Preview Large	LPString	MUST contain the file name of the large image preview of the attachment, if present; otherwise, it MUST be 0x00.
Rendered	**LONG** ([MS-DTYP])	MUST be 0x00000001 if the attachment is rendered inline (valid only for HTML images). Otherwise, it MUST be set to 0x00000000. This value is stored in little-endian format.
Flags	**LONG**	Contains certain implementation-specific flags that correspond to the attachment. When creating a rights-managed e-mail message, this value can be set to 0x00000000. This value is stored in little-endian format.

2.2.4.4.4.1.4 AttachContents Stream

The AttachContents stream (1) stores the actual bits of the attachment, as specified in the following table.

Field name	Format	Description
attachment	Binary	The attachment contents are stored here. This is the same as the value of the

Field name	Format	Description
	data	**PidTagAttachDataBinary** property ([MS-OXCMSG] section 2.2.2.7).

2.2.4.4.4.2 afEmbeddedMessage Storage Structure

The "MailAttachment N" storage structure for attachments that have a value of "afEmbeddedMessage" for the **PidTagAttachMethod** property ([MS-OXCMSG] section 2.2.2.9) is the same as that for attachments that have a value of "afByValue" for the **PidTagAttachMethod** property, with the following exceptions:

- In the AttachDesc stream (1), the **Attach Method** field MUST be set to 0x0005 to represent that the attachment is an embedded message.
- The AttachContents stream (1) MUST be replaced by a .msg storage file, which is the embedded message converted into storage. The format of this file is specified in [MS-OXMSG].
- On receipt of the rights-managed e-mail message with .msg attachments, the client creates an embedded message Attachment object from the contents of the .msg storage file.

2.2.4.4.4.3 PidTagAttachTag Property Value

The "afOle" value applies to RTF message body (2) formats only.

If the value of the attachment's **PidTagAttachTag** property ([MS-OXCMSG] section 2.2.2.15) is "afStorage", the "MailAttachment N" storage MUST contain the value of the **PidTagAttachDataBinary** property ([MS-OXCMSG] section 2.2.2.7) converted to a compound file storage, as specified in [MS-CFB].

If the value of the **PidTagAttachTag** property is not "afStorage", the "MailAttachment N" storage MUST contain a copy of the **PidTagAttachDataObject** property ([MS-OXCMSG] section 2.2.2.8).

2.2.4.5 Format of the message.rpmsg Attachment

As specified in section 2.2.3.1 and section 3.1.4.1.2, the original message is stored in a message attachment called message.rpmsg. The attachment MUST contain a prefix that is formatted as follows:

CHAR STRING prefix - Value is "\x76\xE8\x04\x60\xC4\x11\xE3\x86" in little-endian format.

The prefix is followed by one or more blocks of data. The uncompressed content is divided into segments of 4,096 bytes. Each block of the compressed stream (1) contains the fields specified in the following table.

Field name	Format	Description
ULCheck	**DWORD** ([MS-DTYP])	Marks the beginning of the attachment prefix. The value of this block is 0x00000FA0.
SizeAfterInflation	**DWORD**	Size of the uncompressed data segment. Usually 4,096; the last block can be less.
SizeBeforeInflation	**DWORD**	Size of the compressed data segment.
CompressedDataSegment	Specified in	The compressed bits of the storage container that is

Field name	Format	Description
	[RFC1950]	specified in section 3.1.4.

The client compresses the bits for each successive block, as specified in [RFC1950] and [RFC1951].

3 Protocol Details

3.1 Client Details

The role of the client is to create a rights-managed e-mail message by setting properties to distinguish the message from a normal message and to identify and consume a rights-managed e-mail message when it is received

3.1.1 Abstract Data Model

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

The following abstract data model (ADM) types are defined in this section:

Mailbox

Rights-Managed E-Mail Message Object

3.1.1.1 Per Mailbox

Mailboxes are represented by the **Mailbox** ADM object type. The following ADM objects are maintained for each **Mailbox** ADM object type:

Mailbox.RightsManagedMessageObject: An abstract representation of a rights-managed e-mail message.

3.1.1.2 Per Rights-Managed E-Mail Message Object

Rights-managed e-mail messages are represented by the **RightsManagedMessageObject** ADM object type. The following ADM objects are maintained for each **RightsManagedMessageObject** ADM object type:

RightsManagedMessageObject.License: The Use License for the rights-managed e-mail message.

RightsManagedMessageObject.EncryptedMessage: The original message, encrypted and stored as an attachment on the rights-managed e-mail message.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

When a higher layer triggers the creation of rights-managed e-mail message, the original message, along with its attachments that are to be rights-managed, is encrypted and packaged in a storage container. This storage container is then compressed and stored as an attachment to a wrapper

message that is marked with the specific property, as specified in section 2.2.2, which results in a rights-managed e-mail message. The attachment is also specified with certain properties, as specified in section 2.2.3, that distinguish it from a regular attachment.

3.1.4.1 Creating a Rights-Managed E-Mail Message

To create a rights-managed e-mail message, the client encrypts and compresses the original message as specified in section 3.1.4.1.1, and then creates the wrapper message as specified in section 3.1.4.1.2.

3.1.4.1.1 Encrypting and Compressing the Original Message

To encrypt and compress the contents of a message, the higher layer creates a rights-managed e-mail message. This initiates a handshaking session between the client and the Rights Management Services (RMS) server, resulting in the generation of required certificates by the RMS server for creation and consumption of rights-managed content. This process is specified in [MS-RMPR].

When the client obtains the certificates that are required to create the rights-managed content:

- A storage container that is based on a storage structure referred to as the "compound file", as specified in [MS-CFB], MUST be created with the format as specified in section 2.2.4.4.
- The following fields of the original message MUST be encrypted before they are included in the container:
  - **message body**: Depending on the body format of the message, the message body (2) is contained in one of these properties: **PidTagBody** ([MS-OXPROPS] section 2.688), **PidTagBodyHtml** ([MS-OXCMSG] section 2.2.1.49.1), **PidTagRtfCompressed** ([MS-OXCMSG] section 2.2.1.49.4), **PidTagRtfInSync** ([MS-OXCMSG] section 2.2.1.49.5), or the combination of the **PidTagRtfCompressed** and **PidTagRtfInSync** properties.
  - **attachments**: If any attachments are present in the original message, they MUST be encrypted.
- The publishing license MUST be obtained for the encrypted content, as specified in [MS-RMPR], and packaged in the storage container.
- This storage container MUST then be compressed, as specified in [RFC1951], to reduce its size. The format of the storage container is specified in section 2.2.4.4.

3.1.4.1.2 Creating the Wrapper E-Mail Message

A wrapper e-mail message (that is, a Message object as specified in [MS-OXCMSG]) for the rights-managed e-mail message is created with an attachment that is formatted as specified in section 2.2.4.5. The **PidTagAttachLongFilename** property ([MS-OXCMSG] section 2.2.2.11) of the attachment is set to "message.rpmsg" and the **PidTagAttachMimeTag** property ([MS-OXCMSG] section 2.2.2.26) of the attachment is set to "application/x-microsoft-rpmsg-message". The message.rpmsg attachment SHOULD be the only attachment created on the wrapper e-mail message. If multiple attachments are created on the wrapper e-mail message, the message.rpmsg attachment MUST be the first attachment and all subsequent attachments will be ignored when the client opens a rights-managed e-mail message, as specified in section 3.1.4.2. The **PidNameContentClass** property ([MS-OXCMSG] section 2.2.1.48) of the wrapper message is set to "rpmsg.message".

3.1.4.2 Opening a Rights-Managed E-Mail Message

To open a rights-managed e-mail message, the client scans the message for the properties specified in section 2.2.1 and 2.2.2 and scans the first attachment for the properties specified in section 2.2.3. The client uses the values of these properties to determine whether a message is rights-managed. Following this determination, the client proceeds to open the message, as specified in section 3.1.4.2.1.

3.1.4.2.1 Decompressing and Decrypting the Message

To decompress a message, a client MUST obtain the storage container as specified in section 2.2.4.4 by processing the "message.rpmsg" attachment as specified in section 2.2.4.5, decompressing the compressed bits in each successive data block as specified in [RFC1950] and [RFC1951]. If the **PidNameRightsManagementLicense** property (section 2.2.1.1) is present, the Use License has already been obtained. If it is not present, the client MUST obtain the required Use License from the server by using the publishing license in the container, as specified in [MS-RMPR] section 3.4.4.1.

The client SHOULD<4> then cache the Use License in the **PidNameRightsManagementLicense** property if the Use License is not already present. A user can open the message **offline** if the Use License is cached. When the message is opened, the client SHOULD store the Use License in the RMS License **store** so that the license can be used to open any rights-managed attachments in the message.

By using the Use License, the messaging client decrypts the encrypted content as specified in [MS-RMPR].

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server is responsible for issuing the various certificates and licenses required for the creation and consumption of rights-managed e-mail messages. The role and details of the server in rights management are specified in [MS-RMPR].

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

None.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 Creating a Rights-Managed E-Mail Message

A user creates a rights-managed e-mail message and saves it. The following example describes how a client issues this request, and how the server responds.

Before manipulating rights-managed Message objects, the client requests that the server map named properties to **property IDs** by using the **RopGetPropertyIDsFromNames remote operation (ROP)** ([MS-OXCROPS] section 2.2.8.1).

Property	Property Set GUID	Name or ID
PidNameContentClass ([MS-OXCMSG] section 2.2.1.48)	{00020386-0000-0000-c000-000000000046}	Content class

The server returns the property IDs in response to the **RopGetPropertyIDsFromNames** ROP. The actual IDs are at the discretion of the server.

Property	Property ID
PidNameContentClass	0x806C

To create a rights-managed object, the client uses the **RopCreateMessage** ROP ([MS-OXCROPS] section 2.2.6.2). The server returns a success code and a handle to the Message object. The client then transmits the data to the server by using the **RopSetProperties** ROP ([MS-OXCROPS] section 2.2.8.6).

Property	Property ID	Type	Value
PidNameContentClass	0x806C	0x001F	rpmsg.message

In order to create the message.rpmsg attachment, the client creates the Attachment object by using the **RopCreateAttachment** ROP ([MS-OXCROPS] section 2.2.6.13). Then the client writes out the contents into the attachment by using the **RopOpenStream** ROP ([MS-OXCROPS] section 2.2.9.1) and the **RopSetStreamSize** ROP ([MS-OXCROPS] section 2.2.9.6) followed by the **RopWriteStream** ROP ([MS-OXCROPS] section 2.2.9.3). The client also requests that the server return specific attachment properties, which are then set by using the **RopSetProperties** ROP.

Property	Property ID	Type	Value
PidTagAttachMimeTag ([MS-OXCMSG] section 2.2.2.26)	0x370E	0x001F	application/x-microsoft-rpmsg-message
PidTagAttachLongFilename ([MS-OXCMSG] section 2.2.2.11)	0x3703	0x001F	message.rpmsg

The client saves the Attachment object by using the **RopSaveChangesAttachment** ([MS-OXCROPS] section 2.2.6.15).

When the user is ready to save his changes, the client commits the properties on the server by using the **RopSaveChangesMessage** ROP ([MS-OXCROPS] section 2.2.6.3) and then releases the object by using the **RopRelease** ROP ([MS-OXCROPS] section 2.2.15.3).

The values of some properties will change during the processing of the **RopSaveChangesMessage** ROP, but none of the properties that are specified in this document will change.

5 Security

5.1 Security Considerations for Implementers

The key used to encrypt the content, which is generated by the client, has to be different every time a rights-managed e-mail message is created and whenever any component of the **rights policy template** changes. Security considerations of the client and server also figure in this protocol and are described in [MS-RMPR].

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

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

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

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

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

<1> Section 2.2: Office Outlook 2007 and Outlook 2010 set the following properties (in addition to those specified in sections 2.2.1.1 and 2.2.2.1) on a new rights-managed e-mail message, regardless of user input: **PidLidAgingDontAgeMe** ([MS-OXCMSG] section 2.2.1.33), **PidLidCurrentVersion** ([MS-OXCMSG] section 2.2.1.34), **PidLidCurrentVersionName** ([MS-OXCMSG] section 2.2.1.35), **PidLidPrivate** ([MS-OXCMSG] section 2.2.1.15), **PidLidSideEffects** ([MS-OXCMSG] section 2.2.1.16), **PidTagAlternateRecipientAllowed** ([MS-OXCMSG] section 2.2.1.36), **PidTagClientSubmitTime** ([MS-OXOMSG] section 2.2.3.11), **PidTagDeleteAfterSubmit** ([MS-OXOMSG] section 2.2.3.8), **PidTagImportance** ([MS-OXCMSG] section 2.2.1.11), **PidTagMessageDeliveryTime** ([MS-OXOMSG] section 2.2.3.9), **PidTagPriority** ([MS-OXCMSG] section 2.2.1.12), **PidTagReadReceiptRequested** ([MS-OXOMSG] section 2.2.1.28), **PidTagSensitivity** ([MS-OXCMSG] section 2.2.1.13), **PidLidReminderDelta** ([MS-OXORMDR] section 2.2.1.3), **PidLidReminderSet** ([MS-OXORMDR] section 2.2.1.1), and **PidLidTaskMode** ([MS-OXOTASK] section 2.2.3.2).

<2> Section 2.2.1.1: Office Outlook 2003 does not use the **PidNameRightsManagementLicense** property (section 2.2.1.1) to cache the Use License.

<3> Section 2.2.4.4.1: The WordMailRightsIndex stream (1) is not included in Office Outlook 2003 when replying to a rights-managed e-mail message.

<4> Section 3.1.4.2.1: Office Outlook 2003 does not use this property to cache the Use License.

7 Change Tracking

This section identifies changes that were made to the [MS-OXORMMS] protocol document between the April 2012 and July 2012 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

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

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

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

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

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

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

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

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

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
2.2.4.4.4.1.3 AttachDesc Stream	Removed product behavior notes regarding the Long Path Name field.	N	Product behavior note removed.
2.2.4.5 Format of the message.rpmsg Attachment	Clarified the meaning of the ULCheck block.	N	Content updated.

8 Index