[MS-OXOPOST]:

Post Object Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1		Initial Availability.
4/25/2008	0.2		Revised and updated property names and other technical content.
6/27/2008	1.0		Initial Release.
8/6/2008	1.01		Revised and edited technical content.
9/3/2008	1.02		Updated references.
12/3/2008	1.03		Updated IP notice.
4/10/2009	2.0		Updated applicable product releases.
7/15/2009	3.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	4.1.0	Minor	Updated the technical content.
5/5/2010	4.1.1	Editorial	Revised and edited the technical content.
8/4/2010	4.2	Minor	Clarified the meaning of the technical content.
11/3/2010	4.2	No change	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	4.2	No change	No changes to the meaning, language, and formatting of the technical content.
8/5/2011	4.2	No Change	No changes to the meaning, language, or formatting of the technical content.
10/7/2011	4.2	No Change	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	5.0	Major	Significantly changed the technical content.
4/27/2012	6.0	Major	Significantly changed the technical content.
7/16/2012	6.0	No Change	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	6.1	Minor	Clarified the meaning of the technical content.
2/11/2013	6.1	No Change	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	6.2	Minor	Clarified the meaning of the technical content.
11/18/2013	6.2	No Change	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	6.2	No Change	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	6.2	No Change	No changes to the meaning, language, or formatting of the

Date	Revision History	Revision Class	Comments
			technical content.
7/31/2014	6.2	No Change	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	6.3	Minor	Clarified the meaning of the technical content.
3/16/2015	7.0	Major	Significantly changed the technical content.
5/26/2015	7.0	No Change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

The Post Object Protocol enables the representation of a bulletin board post. This protocol extends the Message and Attachment Object Protocol, which is described in [MS-OXCMSG].

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 [RFC2119]. Sections 1.5 and 1.9 are also normative but do not contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are specific to this document:

conversation: A single representation of a send/response series of email messages. A conversation appears in the Inbox as one unit and allows the user to view and read the series of related email messages in a single effort.

Email object: A **Message object** that represents an email message in a message store and adheres to the property descriptions that are described in in [MS-OXOMSG].

Folder object: A messaging construct that is typically used to organize data into a hierarchy of objects containing Message objects and folder associated information (FAI) Message objects.

handle: Any token that can be used to identify and access an object such as a device, file, or a window.

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

message store: A unit of containment for a single hierarchy of Folder objects, such as a mailbox or public folders.

Post object: A **Message object** that represents an entry in a discussion thread stored in a messaging store.

recipient: An entity that can receive email messages.

remote operation (ROP): An operation that is invoked against a server. Each ROP represents an action, such as delete, send, or query. A ROP is contained in a ROP buffer for transmission over the wire.

ROP request: See ROP request buffer.

search key: A binary-comparable key that identifies related objects for a search.

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

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents

in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

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

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

[MS-OXCFOLD] Microsoft Corporation, "Folder Object Protocol".

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

[MS-OXCPRPT] Microsoft Corporation, "Property and Stream Object Protocol".

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

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

[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

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

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

1.3 Overview

The Post Object Protocol allows a user to post a message to a bulletin board in a **message store**. A **Post object** represents a bulletin board post. There are no properties specific to a Post object. A Post object is stored in a **Folder object**. The Post Object Protocol also specifies how a Post object is created and manipulated.

The Post Object Protocol extends the Message and Attachment Object Protocol, described in [MS-OXCMSG], in that it adds constraints to the properties of a **Message object**.

1.4 Relationship to Other Protocols

The Post Object Protocol has the same dependencies as the Message and Attachment Object Protocol, which it extends. For information about the Message and Attachment Object Protocol, see [MS-OXCMSG].

The Post Object Protocol is a peer of the Email Object Protocol, described in [MS-OXOMSG], and uses a subset of the properties specified for an **E-mail object**.

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

1.5 Prerequisites/Preconditions

The Post Object Protocol has the same prerequisites and preconditions as the Message and Attachment Object Protocol, as specified in [MS-OXCMSG].

1.6 Applicability Statement

A client can use this protocol to create and maintain messages that are posted to a bulletin board in a message store.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

This protocol provides no vendor-extensibility beyond what is already specified in [MS-OXCMSG].

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The Post Object Protocol uses the same underlying transport as that used by the Message and Attachment Object Protocol, as specified in [MS-OXCMSG].

2.2 Message Syntax

A Post object can be created and modified by clients and servers. Except where noted, this section defines constraints under which both clients and servers operate.

Clients operate on a Post object by using the Message and Attachment Protocol, as specified in [MS-OXCMSG]. How a server operates on a Post object is implementation-dependent. The results of any such operations MUST be exposed to clients in a manner that is consistent with the Post Object Protocol.

Unless otherwise specified, a Post object adheres to all property constraints specified in [MS-OXPROPS], [MS-OXCMSG], and [MS-OXOMSG].

2.2.1 Post Object Properties

There are no properties that are specific to a Post object.

2.2.2 Additional Property Constraints

The properties specified in the following sections have additional constraints for this protocol beyond what is specified in [MS-OXCMSG] and [MS-OXOMSG].

2.2.2.1 PidTagConversationIndex Property

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

The **PidTagConversationIndex** property ([MS-OXOMSG] section 2.2.1.3) specifies the depth of the reply in a hierarchical representation of Post objects in one **conversation**.

2.2.2.2 PidTagConversationTopic Property

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

The **PidTagConversationTopic** property ([MS-OXOMSG] section 2.2.1.5) contains an unchanging copy of the original subject. This property MUST be set to the same value as that of the **PidTagNormalizedSubject** property ([MS-OXCMSG] section 2.2.1.10) on a new Post object when it is first saved.

2.2.2.3 PidTagIconIndex Property

Type: **PtypInteger32** ([MS-OXCDATA] section 2.11.1)

The **PidTagIconIndex** property ([MS-OXOMSG] section 2.2.1.10) specifies which icon is to be used by a user interface when displaying a group of Post objects. This value MUST be 0x00000001.

2.2.2.4 PidTagMessageClass Property

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

The **PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3) specifies the type of Message object. This value MUST be "IPM.Post" or MUST begin with "IPM.Post.", in addition to meeting the criteria specified in [MS-OXCMSG]. The string is case-insensitive.

2.2.2.5 Sender Properties

The following properties are specified in [MS-OXOMSG] to represent the sender of an E-mail object. They are used in this protocol to represent the creator of a Post object:

- **PidTagSenderAddressType** ([MS-OXOMSG] section 2.2.1.48)
- **PidTagSenderEntryId** ([MS-OXOMSG] section 2.2.1.50)
- **PidTagSenderName** ([MS-OXOMSG] section 2.2.1.51)
- **PidTagSenderSearchKey** ([MS-OXOMSG] section 2.2.1.52)
- **PidTagSentRepresentingAddressType** ([MS-OXOMSG] section 2.2.1.54)
- **PidTagSentRepresentingEntryId** ([MS-OXOMSG] section 2.2.1.56)
- **PidTagSentRepresentingName** ([MS-OXOMSG] section 2.2.1.57)
- **PidTagSentRepresentingSearchKey** ([MS-OXOMSG] section 2.2.1.58)

2.2.2.6 Recipients

A Post object MUST NOT have **recipients**.

3 Protocol Details

3.1 Client Details

The client creates and manipulates a Post object and operates within the client role as specified in [MS-OXCMSG].

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 specified in this document.

This protocol uses the abstract data model that is specified in [MS-OXCMSG] section 3.1.1 with the following adaptations:

- A Post object extends the Message object.
- A Post object is created in the folder chosen by the user.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Creating the Initial Post Object

When the user creates a message to be posted on a bulletin board, the client creates a Message object as specified in [MS-OXCMSG] section 3.1.4.2, sets properties in accordance with the requirements specified in section 2 and [MS-OXCPRPT], and saves the resulting Post object as specified in [MS-OXCMSG] section 3.1.4.3.

3.1.4.2 Modifying a Post Object

When the user modifies a bulletin board post, the client opens the Post object in the same way that it opens any Message object, as specified in [MS-OXCMSG] section 3.1.4.1. The client then modifies any properties in accordance with the requirements specified in section 2 and [MS-OXCPRPT], and saves the Post object, as specified in [MS-OXCMSG] section 3.1.4.3.

3.1.4.3 Deleting a Post Object

When the user deletes a bulletin board post, the client deletes the Post object in the same way that it deletes any Message object, as specified in [MS-OXCFOLD] section 3.1.4.8.

3.1.4.4 Replying to a Post Object

When the user replies to a bulletin board post, the client creates a new Post object in the same way it creates any Message object, as specified in [MS-OXCMSG] section 3.1.4.2. The new Post object is created in the same Folder object that contains the original Post object. The **PidTagConversationTopic** property (section 2.2.2.2) of the new Post object MUST be copied from the original Post object. The **PidTagConversationIndex** property (section 2.2.2.1) of the new Post object is set to a value that is updated from the **PidTagConversationIndex** property of the original Post object. For details about setting properties and copying properties, see [MS-OXCPRPT].

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 processes a client's requests regarding a Post object and in all other ways operates within the server role as specified in [MS-OXCMSG].

3.2.1 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 specified in this document.

This protocol uses the abstract data model that is specified in [MS-OXCMSG] section 3.2.1 with the following adaptations:

- A Post object extends the Message object.
- A Post object is created in the folder chosen by the user.

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

The server responds to client requests as specified in [MS-OXCMSG] section 3.2.5.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

To post his grocery list of celery and broccoli to a bulletin board, Joe creates a bulletin board post, adds a subject and body, and places it in a folder. The following is a description of what a client might do to accomplish Joe's intentions and the responses a server might return.

To create a Post object, the client sends a **RopCreateMessage** **ROP request** ([MS-OXCROPS] section 2.2.6.2). The server returns a success code and a **handle** to a Message object.

After Joe has input his content for the Post object, the client transmits his data to the server by sending a **RopSetProperties** ROP request ([MS-OXCROPS] section 2.2.8.6).

Property	Property ID	Property type	Value
PidTagIconIndex (section 2.2.2.3)	0x1080	0x0003 (**PtypInteger32** ([MS-OXCDATA] section 2.11.1))	0x00000001
PidTagMessageClass (section 2.2.2.4)	0x001A	0x001F (**PtypString** ([MS-OXCDATA] section 2.11.1)	"IPM.Post"
PidTagNormalizedSubject ([MS-OXCMSG] section 2.2.1.10)	0x0E1D	0x001F	"Grocery List"
PidTagSubjectPrefix ([MS-OXCMSG] section 2.2.1.9)	0x003D	0x001F	""(null)
PidTagConversationTopic (section 2.2.2.2)	0x0070	0x001F	"Grocery List"
PidTagConversationIndex (section 2.2.2.1)	0x0071	0x0102 (**PtypBinary** [MS-OXCDATA] section 2.11.1)	(Set as described in note 1 following this table.)
PidTagHtml ([MS-OXCMSG] section 2.2.1.56.9)	0x1013	0x0102	<html> <head> <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii"> </head> <body lang=EN-US> Celery Broccoli </body> </html>
PidTagSenderName ([MS-OXOMSG] section 2.2.1.51)	0x0C1A	0x001F	"Joe Healy"
PidTagSenderAddressType ([MS-OXOMSG] section 2.2.1.48)	0x0C1E	0x001F	"EX"
PidTagSenderEntryId ([MS-OXOMSG]	0x0C19	0x0102	(Set as described in note 2

Property	Property ID	Property type	Value
section 2.2.1.50)			following this table.)
PidTagSenderSearchKey ([MS-OXOMSG] section 2.2.1.52)	0x0C1D	0x0102	(Set as described in note 3 following this table.)
PidTagSentRepresentingName ([MS-OXOMSG] section 2.2.1.57)	0x0042	0x001F	"Joe Healy"
PidTagSentRepresentingAddressType ([MS-OXOMSG] section 2.2.1.54)	0x0064	0x001F	"EX"
PidTagSentRepresentingEntryId ([MS-OXOMSG] section 2.2.1.56)	0x0041	0x0102	(Set as described in note 2 following this table.)
PidTagSentRepresentingSearchKey ([MS-OXOMSG] section 2.2.1.58)	0x003B	0x0102	(Set as described in note 3 following this table.)

Notes

1. The **PidTagConversationIndex** property is set with a depth of 1 and has the following binary contents:

```
0000: 01 C8 73 2D A1 0A 3E B3-EE 24 90 F4 45 BE 97 10
0010: 90 B2 A5 07 7A 13
```

2. The values of the **PidTagSenderEntryId** and **PidTagSentRepresentingEntryId** properties are identical because Joe isn't posting this on behalf of another user. These properties have the following 125-byte value:

```
0000: 00 00 00 00 DC A7 40 C8-C0 42 10 1A B4 B9 08 00 ......@..B......
0010: 2B 2F E1 82 01 00 00 00-00 00 00 00 2F 4F 3D 46 +/.........../O=F
0020: 49 52 53 54 20 4F 52 47-41 4E 49 5A 41 54 49 4F IRST ORGANIZATIO
0030: 4E 2F 4F 55 3D 45 58 43-48 41 4E 47 45 20 41 44 N/OU=EXCHANGE AD
0040: 4D 49 4E 49 53 54 52 41-54 49 56 45 20 47 52 4F MINISTRATIVE GRO
0050: 55 50 20 28 46 59 44 49-42 4F 48 46 32 33 53 50 UP (FYDIBOHF23SP
0060: 44 4C 54 29 2F 43 4E 3D-52 45 43 49 50 49 45 4E DLT)/CN=RECIPIEN
0070: 54 53 2F 43 4E 3D 4A 48-45 41 4C 59 00          TS/CN=JHEALY.
```

3. The values of **PidTagSenderSearchKey** and **PidTagSentRepresentingSearchKey** properties are identical because Joe isn't posting this on behalf of another user. The contents of these properties are used as Joe's **search key**. These properties have the following 100-byte value:

```
0000: 45 58 3A 2F 4F 3D 46 49-52 53 54 20 4F 52 47 41 EX:/O=FIRST ORGA
0010: 4E 49 5A 41 54 49 4F 4E-2F 4F 55 3D 45 58 43 48 NIZATION/OU=EXCH
0020: 41 4E 47 45 20 41 44 4D-49 4E 49 53 54 52 41 54 ANGE ADMINISTRAT
0030: 49 56 45 20 47 52 4F 55-50 20 28 46 59 44 49 42 IVE GROUP (FYDIB
0040: 4F 48 46 32 33 53 50 44-4C 54 29 2F 43 4E 3D 52 OHF23SPDLT)/CN=R
0050: 45 43 49 50 49 45 4E 54-53 2F 43 4E 3D 4A 48 45 ECIPIENTS/CN=JHE
0060: 41 4C 59 00                                     ALY.
```

When Joe is ready to save his changes, the client commits the properties on the server by sending a **RopSaveChangesMessage** ROP request ([MS-OXCROPS] section 2.2.6.3) and then releases the handle for the Message object by sending a **RopRelease** ROP request ([MS-OXCROPS] section 2.2.15.3).

The values of some properties will change during the processing of the **RopSaveChangesMessage ROP**, but the properties specified in section 2.2.2 will not change.

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to the Post Object Protocol. General security considerations pertaining to the underlying transport apply, as described in [MS-OXCMSG].

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

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

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

7 Change Tracking

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

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

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

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

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

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

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

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

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

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

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

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

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

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

8 Index

S

T

V