

# [MS-OXCSTOR]: Store 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's Open Specification Promise (available here: <http://www.microsoft.com/interop/osp>) or the Community Promise (available here: <http://www.microsoft.com/interop/cp/default.mspix>). 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](mailto: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.

## 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		Revised and edited technical content.
12/03/2008	1.03		Revised and edited technical content.
02/04/2009	1.04		Revised and edited technical content.
04/10/2009	2.0		Updated technical content for new 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.

# Table of Contents

<b>1 Introduction</b>	<b>7</b>
1.1 Glossary	7
1.2 References	8
1.2.1 Normative References	8
1.2.2 Informative References	9
1.3 Overview	9
1.3.1 Private and Public Stores	9
1.3.2 Opening a Connection to a Store	9
1.4 Relationship to Other Protocols	10
1.5 Prerequisites/Preconditions	10
1.6 Applicability Statement	10
1.7 Versioning and Capability Negotiation	10
1.8 Vendor-Extensible Fields	10
1.9 Standards Assignments	10
<b>2 Messages</b>	<b>11</b>
2.1 Transport	11
2.2 Message Syntax	11
2.2.1 Remote Operations	11
2.2.1.1 RopLogon Semantics	11
2.2.1.1.1 Request	11
2.2.1.1.1.1 LogonFlags	11
2.2.1.1.1.2 OpenFlags	12
2.2.1.1.1.2.1 USE_PER_MDB_REPLID_MAPPING Details	13
2.2.1.1.1.3 StoreState	13
2.2.1.1.1.4 EssdnSize	13
2.2.1.1.1.5 Essdn	13
2.2.1.1.2 Redirect Response	13
2.2.1.1.2.1 LogonFlags	14
2.2.1.1.2.2 ServerNameSize	14
2.2.1.1.2.3 ServerName	14
2.2.1.1.3 Success Response for Private Mailbox	14
2.2.1.1.3.1 LogonFlags	14
2.2.1.1.3.2 FolderIds	14
2.2.1.1.3.3 ResponseFlags	15
2.2.1.1.3.4 MailboxGuid	15
2.2.1.1.3.5 ReplId	15
2.2.1.1.3.6 ReplGuid	15
2.2.1.1.3.7 LogonTime	15
2.2.1.1.3.8 GwartTime	15
2.2.1.1.3.9 StoreState	15
2.2.1.1.4 Success Response for Public Folders	16
2.2.1.1.4.1 LogonFlags	16
2.2.1.1.4.2 FolderIds	16
2.2.1.1.4.3 ReplId	16
2.2.1.1.4.4 ReplGuid	16
2.2.1.1.4.5 PerUserGuid	17
2.2.1.1.5 ReturnValue	17
2.2.1.2 RopGetReceiveFolder Semantics	17
2.2.1.2.1 Request	17

2.2.1.2.1.1	MessageClass.....	18
2.2.1.2.2	Response .....	18
2.2.1.2.2.1	FolderId .....	18
2.2.1.2.2.2	ExplicitMessageClass.....	18
2.2.1.2.3	ReturnValue .....	19
2.2.1.3	RopSetReceiveFolder Semantics .....	19
2.2.1.3.1	Request .....	19
2.2.1.3.1.1	FolderId .....	19
2.2.1.3.1.2	MessageClass.....	20
2.2.1.3.2	Response .....	20
2.2.1.3.3	ReturnValue .....	20
2.2.1.4	RopGetReceiveFolderTable Semantics .....	20
2.2.1.4.1	Request .....	21
2.2.1.4.2	Response .....	21
2.2.1.4.2.1	RowCount.....	21
2.2.1.4.2.2	Rows .....	21
2.2.1.4.3	ReturnValue .....	21
2.2.1.5	RopGetStoreState Semantics .....	22
2.2.1.5.1	Request .....	22
2.2.1.5.2	Response .....	22
2.2.1.5.2.1	StoreState.....	22
2.2.1.5.3	ReturnValue .....	22
2.2.1.6	RopGetOwningServers Semantics .....	22
2.2.1.6.1	Request .....	23
2.2.1.6.1.1	FolderId .....	23
2.2.1.6.2	Response .....	23
2.2.1.6.2.1	OwningServersCount .....	23
2.2.1.6.2.2	CheapServersCount .....	23
2.2.1.6.2.3	OwningServers.....	23
2.2.1.6.3	ReturnValue .....	23
2.2.1.7	RopPublicFolderIsGhosed Semantics .....	24
2.2.1.7.1	Request .....	24
2.2.1.7.1.1	FolderId .....	24
2.2.1.7.2	Response .....	24
2.2.1.7.2.1	IsGhosed.....	24
2.2.1.7.2.2	ServersCount.....	24
2.2.1.7.2.3	CheapServersCount .....	24
2.2.1.7.2.4	Servers .....	24
2.2.1.7.3	ReturnValue .....	25
2.2.1.8	RopLongTermIdFromId Semantics .....	25
2.2.1.8.1	Request .....	25
2.2.1.8.1.1	ObjectId.....	25
2.2.1.8.2	Response .....	25
2.2.1.8.2.1	LongTermId.....	25
2.2.1.8.3	ReturnValue .....	25
2.2.1.9	RopIdFromLongTermId Semantics .....	26
2.2.1.9.1	Request .....	26
2.2.1.9.1.1	LongTermId.....	26
2.2.1.9.2	Response .....	26
2.2.1.9.2.1	ObjectId.....	26
2.2.1.9.3	ReturnValue .....	26
2.2.1.10	RopGetPerUserLongTermIds Semantics .....	26
2.2.1.10.1	Request .....	26

2.2.1.10.1.1 DatabaseGuid.....	26
2.2.1.10.2 Response .....	27
2.2.1.10.2.1 LongTermIdCount .....	27
2.2.1.10.2.2 LongTermIds.....	27
2.2.1.10.3 ReturnValue.....	27
2.2.1.11 RopGetPerUserGuid Semantics .....	27
2.2.1.11.1 Request .....	27
2.2.1.11.1.1 LongTermId .....	27
2.2.1.11.2 Response .....	27
2.2.1.11.2.1 DatabaseGuid.....	27
2.2.1.11.3 ReturnValue.....	28
2.2.1.12 RopReadPerUserInformation Semantics .....	28
2.2.1.12.1 Request .....	29
2.2.1.12.1.1 FolderId.....	29
2.2.1.12.1.2 Reserved .....	29
2.2.1.12.1.3 DataOffset .....	29
2.2.1.12.1.4 MaxDataSize .....	29
2.2.1.12.2 Response .....	29
2.2.1.12.2.1 HasFinished .....	29
2.2.1.12.2.2 DataSize.....	29
2.2.1.12.2.3 Data .....	30
2.2.1.12.3 ReturnValue.....	30
2.2.1.13 RopWritePerUserInformation Semantics.....	30
2.2.1.13.1 Request .....	31
2.2.1.13.1.1 FolderId.....	31
2.2.1.13.1.2 HasFinished .....	31
2.2.1.13.1.3 DataOffset .....	31
2.2.1.13.1.4 DataSize.....	31
2.2.1.13.1.5 Data .....	31
2.2.1.13.1.6 ReplGuid.....	31
2.2.1.13.2 Response .....	31
2.2.1.13.3 ReturnValue.....	31
2.2.2 Logon-Specific Properties .....	32
2.2.2.1 Private Mailbox Logon .....	32
2.2.2.2 Public Folders Logon .....	34
<b>3 Protocol Details.....</b>	<b>35</b>
3.1 Client Details.....	35
3.1.1 Abstract Data Model .....	35
3.1.2 Timers .....	35
3.1.3 Initialization .....	35
3.1.4 Higher-Layer Triggered Events.....	35
3.1.4.1 Logging on to a Store.....	35
3.1.4.2 Converting Between LongTermIDs and ShortTermIDs .....	36
3.1.4.3 Syncing Per-User Read/Unread Data for Public Folders .....	36
3.1.4.4 Registering for Notifications .....	37
3.1.5 Message Processing Events and Sequencing Rules.....	37
3.1.6 Timer Events .....	37
3.1.7 Other Local Events .....	37
3.2 Server Details .....	37
3.2.1 Abstract Data Model .....	37
3.2.2 Timers .....	38
3.2.3 Initialization .....	38

3.2.4	Higher-Layer Triggered Events .....	38
3.2.5	Message Processing Events and Sequencing Rules .....	39
3.2.5.1	Processing RopLogon .....	39
3.2.5.1.1	Private Mailbox Logon .....	39
3.2.5.1.2	Public Folders Logon .....	40
3.2.5.2	Processing RopGetReceiveFolder .....	41
3.2.5.3	Processing RopSetReceiveFolder .....	41
3.2.5.4	Processing RopGetReceiveFolderTable .....	42
3.2.5.5	Processing RopGetStoreState .....	42
3.2.5.6	Processing RopGetOwningServers .....	42
3.2.5.7	Processing RopPublicFolderIsGhosted .....	43
3.2.5.8	Processing RopLongTermIdFromId .....	44
3.2.5.9	Processing RopIdFromLongTermId .....	44
3.2.5.10	Processing RopGetPerUserLongTermIds .....	45
3.2.5.11	Processing RopGetPerUserGuid .....	45
3.2.5.12	Processing RopReadPerUserInformation .....	45
3.2.5.12.1	Private Mailbox Specific Behavior .....	45
3.2.5.12.2	Public Folders Specific Behavior .....	45
3.2.5.12.3	Common Behavior .....	45
3.2.5.13	Processing RopWritePerUserInformation .....	47
3.2.5.13.1	Common Behavior .....	47
3.2.5.13.2	Private Mailbox Specific Behavior .....	47
3.2.5.13.3	Public Folders Specific Behavior .....	47
3.2.6	Timer Events .....	47
3.2.7	Other Local Events .....	47
<b>4</b>	<b>Protocol Examples .....</b>	<b>48</b>
4.1	RopLogon for a Private Mailbox .....	48
4.2	RopLogon for Public Folders .....	49
4.3	RopGetReceiveFolder .....	50
4.4	RopSetReceiveFolder .....	51
4.5	RopGetReceiveFolderTable .....	51
4.6	RopIdFromLongTermId .....	51
4.7	RopGetPerUserLongTermIds .....	52
4.8	RopReadPerUserInformation .....	52
4.9	RopWritePerUserInformation .....	53
<b>5</b>	<b>Security .....</b>	<b>54</b>
5.1	Security Considerations for Implementers .....	54
5.2	Index of Security Fields .....	54
<b>6</b>	<b>Appendix A: Product Behavior .....</b>	<b>55</b>
<b>7</b>	<b>Change Tracking .....</b>	<b>57</b>
<b>8</b>	<b>Index .....</b>	<b>60</b>

# 1 Introduction

This document specifies the Store object protocol, which is used by clients to: log on to a private user **mailbox** or **public folders**; read and write mailbox-level **properties** for that user mailbox; perform various housekeeping tasks for that mailbox; and determine the availability of content for public folders.

## 1.1 Glossary

The following terms are defined in [\[MS-OXGLOS\]](#):

- Active Directory**
- active replica**
- ASCII**
- attachment**
- binary large object (BLOB)**
- change number (CN)**
- change number set (CNSET)**
- code page (1)**
- Coordinated Universal Time (UTC)**
- distinguished name (DN)**
- double-byte character set (DBCS)**
- enterprise/site/server distinguished name (ESSDN)**
- EntryID**
- folder**
- folder associated information (FAI)**
- folder ID (FID)**
- Gateway Address Routing Table (GWART)**
- Global Address List (GAL)**
- GUID**
- handle**
- IDSET**
- interpersonal messaging subtree**
- IPM subtree**
- little-endian**
- local replica**
- Logon object**
- LogonID**
- LongTermID**
- mailbox**
- message**
- message class**
- message database (MDB)**
- message ID (MID)**
- named property**
- non-interpersonal messaging subtree**
- non-IPM subtree**
- Out of Office (OOF)**
- property (1)**
- property tag**
- public folder**
- Receive folder**
- remote operation (ROP)**
- remote procedure call (RPC)**

**replica (1)**  
**replica GUID (REPLGUID)**  
**replica ID (REPLID)**  
**ROP request buffer**  
**ROP response buffer**  
**Server object**  
**Session Context Handle (CXH)**  
**ShortTermID**  
**special folder**  
**store**  
**table**  
**Unicode**

The following terms are specific to this document:

**global directory:** A globally accessible database containing entries that correlate servers, databases, and user **mailboxes**. The server uses the correlated data to determine, for a specific user, which server and database to access for a private **mailbox** logon or a **public folder** logon. The global directory also contains other pertinent configuration information that is crucial to the overall operation of the client/server deployment. **Active Directory** can be used for the global directory, but the implementer determines what to use for the global directory.

**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

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

[MS-OXCDATA] Microsoft Corporation, "[Data Structures](#)", April 2008.

[MS-OXCFOLD] Microsoft Corporation, "[Folder Object Protocol Specification](#)", April 2008.

[MS-OXCFXICS] Microsoft Corporation, "[Bulk Data Transfer Protocol Specification](#)", April 2008.

[MS-OXCMAIL] Microsoft Corporation, "[RFC2822 and MIME to E-Mail Object Conversion Protocol Specification](#)", April 2008.

[MS-OXCNOTIF] Microsoft Corporation, "[Core Notifications Protocol Specification](#)", April 2008.

[MS-OXCPRPT] Microsoft Corporation, "[Property and Stream Object Protocol Specification](#)", April 2008.

[MS-OXCROPS] Microsoft Corporation, "[Remote Operations \(ROP\) List and Encoding Protocol Specification](#)", April 2008.



[MS-OXCRPC] Microsoft Corporation, "[Wire Format Protocol Specification](#)", April 2008.

[MS-OXDISCO] Microsoft Corporation, "[Autodiscover HTTP Service Protocol Specification](#)", April 2008.

[MS-OXDSELI] Microsoft Corporation, "[Autodiscover Publishing and Lookup Protocol Specification](#)", April 2008.

[MS-OXGLOS] Microsoft Corporation, "[Exchange Server Protocols Master Glossary](#)", April 2008.

[MS-OXORULE] Microsoft Corporation, "[E-Mail Rules Protocol Specification](#)", April 2008.

[MS-OXWOOF] Microsoft Corporation, "[Out of Office \(OOO\) Web Service Protocol Specification](#)", April 2008.

[MS-UCODEREF] Microsoft Corporation, "Windows Protocols Unicode Reference", July 2007, <http://go.microsoft.com/fwlink/?LinkId=112413>

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

## 1.2.2 Informative References

None.

## 1.3 Overview

### 1.3.1 Private and Public Stores

The client can log on to a private user mailbox for access to that user's mailbox data (**folders**, **messages** and **attachments**). Once logged on, the client can perform operations on the mailbox using the operations specified in this protocol. The client can also simultaneously log on to other users' mailboxes, and granted sufficient permissions by that other user, access that user's mailbox data as well as perform operations on the mailbox. Additionally, the client can simultaneously log on to a public folder **store**.

The content within an entire private mailbox is confined to a single server. The client determines which server to log on to from **global directory** data about the user. If the mailbox has been moved to another server, an attempt to log on to the wrong server will result in an error response from the server, along with a return value providing guidance about which server to try next.

The content within the public folders store is typically spread across many different servers, and is replicated among those servers. The client determines which public folder server to log on to by using the global directory information about the user. All the servers that host public folders contain a complete copy of the public folders store's folder hierarchy. However, a specific server does not have to have the contents of any particular public folder. The set of servers that contain content for a specific folder are said to be content replicas for that folder. A client attempting to read folder content from a server that is not a content replica for that folder will result in an error response. The client is then able to use operations specified in this protocol to discover which servers are content replicas for the folder. After making that determination, the client then logs on to one of those servers to read or update the content for that folder.

### 1.3.2 Opening a Connection to a Store

The client first connects to the server in question and establishes a session context as specified in [\[MS-OXCRPC\]](#) section 3.1.4.11. Once that connection is made, the client is then able to follow the

protocol specified in this document to establish a logon session with a private mailbox, or the public folders. After the logon session is established, the client can follow the protocol specified in this document to perform various operations on the user mailbox and make discoveries about where public folder content is located. Note that establishing a session context and subsequently establishing a logon session are the prerequisite steps for all other **remote operations (ROPs)** specified in [\[MS-OXCROPS\]](#).

#### 1.4 Relationship to Other Protocols

The Store Object protocol relies on the Remote Operations (ROP) List and Encoding protocol, as specified in [\[MS-OXCROPS\]](#), and the Wire Format protocol, as specified in [\[MS-OXCRPC\]](#).

All protocols that issue ROPs rely on the Store Object protocol in that they will first successfully complete a [RopLogon](#), as specified in this document.

#### 1.5 Prerequisites/Preconditions

The Store Object protocol assumes that the client has previously connected to the server, as specified in [\[MS-OXCRPC\]](#). All ROPs, except [RopLogon](#), are performed with the assumption that the client has successfully logged on to the server using [RopLogon](#).

#### 1.6 Applicability Statement

The store object represents the connection to a specific mailbox or the public folder store and is identified by a **Logon object handle**. This Logon object handle is used by all other protocols which issue ROPs, including the ROPs described in this protocol.

#### 1.7 Versioning and Capability Negotiation

None.

#### 1.8 Vendor-Extensible Fields

None.

#### 1.9 Standards Assignments

None.

## 2 Messages

### 2.1 Transport

The **ROP request buffers** and **ROP response buffers** specified by this protocol are sent to and received from the server by using the underlying **Remote Procedure Call** transport, as specified in [\[MS-OXCRPC\]](#).

### 2.2 Message Syntax

Unless otherwise specified, unit sizes in this section are expressed in bytes.

#### 2.2.1 Remote Operations

The following sections specify the fields passed in ROP request buffers that are specific to the Store Object protocol.

Before sending a [RopLogon](#) request to the server, the client MUST be connected to the server using the **EcDoConnectEx** RPC as specified in [\[MS-OXCRPC\]](#) section 3.1.4.11. For other ROPs, the client MUST have successfully completed a [RopLogon](#) operation and MUST pass a valid **Server object** value (representing a Logon object obtained from the successful completion of a [RopLogon](#) operation).

For all flag fields specified in the following sections, all bits in the bit field not specified in this document MUST NOT be set by the client and MUST be ignored by the server.

##### 2.2.1.1 RopLogon Semantics

The syntax of the [RopLogon](#) request and response buffer is specified in [\[MS-OXCROPS\]](#) section 2.2.2.1.

[RopLogon](#) establishes a logon session between the client and the server. It is the basis of all further ROPs, and successfully completing a [RopLogon](#) is a prerequisite for performing all other ROPs listed in this specification.

##### 2.2.1.1.1 Request

Several fields are passed from the client to the server, and the presence of some of them depends on flag bits present or absent in other fields. Other flag fields control server-side behavior. Any flag values not defined here MUST NOT be set by a client. If the client uses an undefined flag value, then the server SHOULD reject the operation with **ecError** in the **ReturnValue** field of the response. [<1>](#)

The fields in the request buffer are specified in this section.

##### 2.2.1.1.1.1 LogonFlags

Contains flags that control the behavior of the logon. Individual flag values and their meanings are specified in the following table.

Name	Value	Description
Private	0x01	This bit is set for logon to a private mailbox and is not set for logon to public folders.

Name	Value	Description
Undercover	0x02	This bit is ignored by the server and is returned to the client in the response.
Ghosted	0x04	<p>If the Private bit is set, this bit MUST NOT be set by the client and MUST be ignored by the server.</p> <p>If this bit is not set AND the <b>OpenFlags</b> field does not have any of the following bits set ALTERNATE_SERVER IGNORE_HOME_MDB, the server will use the global directory to find the default public folder database to log on to. If this server does not host that database, the <b>ReturnValue</b> will be ecWrongServer.</p> <p>Otherwise, the server will log on to the public folder database present on the server, if there is one. If there is no public folder database on the server, <b>ReturnValue</b> will be eLoginFailure. For more details about the <b>ReturnValue</b> field, see section <a href="#">2.2.1.1.5</a>.</p> <p>For successful responses, this bit is returned unchanged in the response. For more details, see section <a href="#">2.2.1.1.3.1</a> and section <a href="#">2.2.1.1.4.1</a>.</p>
SplProcess	0x08	This bit is ignored by the server.

### 2.2.1.1.1.2 OpenFlags

Contains additional flags that control the behavior of the logon. Individual flag values and their meanings are specified in the following table.

Name	Value	Description
USE_ADMIN_PRIVILEGE	0x00000001	When set, this bit indicates that the user is requesting administrative access to the mailbox. To grant administrative access to the mailbox, the server MUST confirm that the user has the right to such access. Confirmation is implementation-dependent.
PUBLIC	0x00000002	If set, <a href="#">RopLogon</a> opens the public folders store. Otherwise, <a href="#">RopLogon</a> opens a private user mailbox.
HOME_LOGON	0x00000004	This bit is ignored. <2>
TAKE_OWNERSHIP	0x00000008	This bit is ignored. <3>
ALTERNATE_SERVER	0x00000100	Requests a private server to provide an alternate public server.
IGNORE_HOME_MDB	0x00000200	This bit is used only for public logons. When set, this bit allows the client to log on to a public <b>MDB</b> that is not the user's default public MDB; otherwise, attempts to log on to a public MDB that is not the user's default results in the client being redirected back to the user's default public MDB.
NO_MAIL	0x00000400	Requests a non-messaging logon session. Non-messaging sessions allow clients to access the store,

Name	Value	Description
		but do not allow messages to be sent or received.
USE_PER_MDB_REPLID_MAPPING	0x01000000	For a private-mailbox logon, the client uses this bit to control server behavior as specified in section <a href="#">2.2.1.1.1.2.1</a> . For logons to a public folder store, this bit is ignored.

#### 2.2.1.1.1.2.1 USE\_PER\_MDB\_REPLID\_MAPPING Details

The client uses the USE\_PER\_MDB\_REPLID\_MAPPING bit of the **OpenFlags** field to control how the server maintains the **replica ID (REPLID)**-to-**replica GUID (REPLGUID)** mapping and **named property**-to-**property ID** mapping on an RPC session. [<4>](#)

If the USE\_PER\_MDB\_REPLID\_MAPPING bit is not set, then the client assumes that all logons on an RPC session use the same REPLID/REPLGUID mapping and named property/property ID mapping. Therefore, the server is expected to maintain one REPLID-to-EPRLGUID mapping and one named property-to-property ID mapping for all logon sessions. The server MUST maintain these mappings in one database, called the "reference database". The database that is opened in the first logon operation on the current RPC session is the reference database; a database that is opened in a subsequent logon operation is a secondary database. The REPLIDs and named properties in each secondary-database map MUST be re-mapped through the reference database. The mappings in the reference database are used for all logon sessions on the current RPC session.

If the USE\_PER\_MDB\_REPLID\_MAPPING bit is set, then the client assumes that each logon on an RPC session creates its own mapping. Therefore, the server is not expected to use a reference database for all REPLID/REPLGUID mapping and named property/property ID mapping.

#### 2.2.1.1.1.3 StoreState

Unused. SHOULD be set to zero by the client and MUST be ignored by the server.

#### 2.2.1.1.1.4 EssdnSize

MUST be the length in bytes of the **ESSDN** string, including the terminating NULL character. Clients MUST pass zero for public folders logons.

#### 2.2.1.1.1.5 Essdn

Contains an **ASCII** string that uniquely identifies a mailbox to log on to. The mailbox descriptor in the global directory will contain enough other data to identify the correct server hosting the user's mailbox, as well as how to find that specific mailbox on that server. The string includes the terminating NULL character. The string length (including the terminating NULL character) MUST be equal to the value specified by the **EssdnSize** field.

The string to be used in this field is the value of the legacy **distinguished name (DN)** attribute of the user object that is obtained by using the Autodiscover Publishing and Lookup protocol, as specified in [\[MS-OXDSCLI\]](#).

#### 2.2.1.1.2 Redirect Response

The following return values are included in the [RopLogon](#) response when the value of the **ReturnValue** field is ecWrongServer.

#### 2.2.1.1.2.1 LogonFlags

Composed of the values of the Private, Undercover, and ghosted bits passed to the server in the **LogonFlags** field in the [RopLogon](#) request.

#### 2.2.1.1.2.2 ServerNameSize

Contains the length of the string of the **ServerName** field, including the terminating NULL character.

#### 2.2.1.1.2.3 ServerName

Contains the enterprise/site/server distinguished name (ESSDN) of server for the client to connecting to, as the server included in the request either no longer hosts the requested mailbox (it was moved), or was the wrong server to connect to for access to public folders. The string includes the terminating NULL character. The string length (including the terminating NULL character) **MUST** be equal to the value specified by the **ServerNameSize** field.

#### 2.2.1.1.3 Success Response for Private Mailbox

The following return values are included in the [RopLogon](#) response only when the Private bit is set in the **LogonFlags** field of the [RopLogon](#) request.

##### 2.2.1.1.3.1 LogonFlags

Composed of the values of the Private, Undercover, and ghosted bits passed to the server in the **LogonFlags** field of the [RopLogon](#) request.

##### 2.2.1.1.3.2 FolderIds

Identifies the **folder ID (FID)** of all of the following folders:

- Mailbox Root Folder. All other folders listed here are direct or indirect children of this folder.
- Deferred Action Folder
- Spooler Queue
- Interpersonal Messages Subtree (root of the user-visible portion of the folder hierarchy)
- Inbox
- Outbox
- Sent Items
- Deleted Items
- Common Views
- Schedule
- Search
- Views
- Shortcuts

### 2.2.1.1.3.3 ResponseFlags

Contains flags that provide details about the state of the mailbox. Individual flag values and their meanings are specified in the following table.

Name	Value	Description
Reserved	0x01	MUST be set.
OwnerRight	0x02	If set, the user has Full Owner or View Admin rights for the mailbox.
SendAsRight	0x04	If set, the user has the right to send mail from this mailbox.
OOF	0x10	Indicates whether <b>Out of Office (OOF)</b> is set for the mailbox. For more details about the OOF state, see <a href="#">[MS-OXWOOF]</a> .

### 2.2.1.1.3.4 MailboxGuid

Contains the **GUID** of the mailbox that was logged on to.

### 2.2.1.1.3.5 ReplId

Contains the short form of the value specified in the REPLGUID field.

### 2.2.1.1.3.6 ReplGuid

Contains the GUID used to identify the source of the REPLID to REPLGUID mapping and named property mappings. If the client did not set the **USE\_PER\_MDB\_REPLID MAPPING** bit in the OpenFlags field, this value MUST be identical for all private mailbox logons made to the server on the same RPC session. If the client did set the **USE\_PER\_MDB\_REPLID MAPPING** bit in the OpenFlags field, the server can return values as deemed appropriate by the implementation.

If the server returns the same value for different logons, the server MUST use the same REPLID to REPLGUID and named property mappings for those different logons.

On successive logon attempts to the same mailbox, if the client receives a different value in this return field as compared to a previous logon to the same mailbox, any client-side cached mappings of REPLIDs to REPLGUIDs or named properties MUST be disposed of. [<5>](#)

### 2.2.1.1.3.7 LogonTime

Contains the **Coordinated Universal Time (UTC)** time on the server when the logon was performed. For more details about the format of this field, see [\[MS-OXCROPS\]](#) section 2.2.2.1.2.1.

### 2.2.1.1.3.8 GwartTime

Contains the UTC time when the **Gateway Address Routing Table (GWART)** was last updated on the server. For details about address types, see the details of the [PidTagAddressType](#) property, as specified in [\[MS-OXCMAIL\]](#) section 2.1.1.9.

### 2.2.1.1.3.9 StoreState

If the mailbox currently has any active search folders, this bit field MUST have the STORE\_HAS\_SEARCHES flag set. All other bits MUST NOT be set by the server and SHOULD be ignored by the client.

Name	Value	Description
STORE_HAS_SEARCHES	0x01000000	Indicates whether the mailbox has active search folders being populated. For more details about search folders, see <a href="#">[MS-OXCSTOR]</a> .

#### 2.2.1.1.4 Success Response for Public Folders

The following return values are sent only when the Private bit is not set in the **LogonFlags** field of the [RopLogon](#) request.

##### 2.2.1.1.4.1 LogonFlags

Composed of the values of the Private, Undercover, and ghosted bits passed to the server in the **LogonFlags** field of the [RopLogon](#) request.

##### 2.2.1.1.4.2 FolderIds

Identifies the FID of all of the following folders;

- Public Folders **Root Folder**. All other folders listed here are direct or indirect children of this folder.
- Interpersonal Messages Subtree
- Non-Interpersonal Messages Subtree
- EForms Registry Root Folder
- Free/Busy Data Root Folder
- Offline Address Book Data Root Folder
- EForms Registry for the user's locale
- Local Site's Free/Busy Data Folder
- Local Site's Offline Address Book Data Folder
- NNTP Article Index Folder
- Empty
- Empty
- Empty

##### 2.2.1.1.4.3 ReplId

Contains the short form of the value specified in the REPLGUID field.

##### 2.2.1.1.4.4 ReplGuid

Contains the GUID used to identify the origin of ID and named property mappings. This value is randomly assigned to a database when it is created and is an integral part of all IDs assigned in the database. It is used in forming **LongTermIDs.<6>**



#### 2.2.1.1.4.5 PerUserGuid

This field is not used and is ignored by the client. The server SHOULD set this field to an empty GUID (all zeroes).

#### 2.2.1.1.5 ReturnValue

The following table describes the common return codes that are returned in the **ReturnValue** field of a [RopLogon](#) response. Other return codes are possible, and are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecLoginFailure	0x80040111	A login failure occurred. A possible cause is that a private logon was requested without a mailbox distinguished name (DN), which is specified in the <b>Essdn</b> field. <a href="#">&lt;7&gt;</a>
ecUnknownUser	0x000003EB	The user that is specified by the <b>Essdn</b> field is unknown to the system.
ecUnknownCodePage	0x000003EF	The <b>code page</b> for this session is unknown.
ecMailboxDisabled	0x0000096C	The user account is marked as disabled.
ecMailboxInTransit	0x0000050C	The mailbox is in transit; logon is not allowed
ecWrongServer	0x00000478	The requested message database (MDB) for logon is not the user's home MDB.
ecProfileNotConfigured	0x0000011C	A user tries to log on to a non-home MDB and the USE_ADMIN_PRIVILEGE bit in the <b>OpenFlags</b> field is not set.
ecAccessDenied	0x80070005	The user does not have sufficient permissions to the mailbox.
ecLoginPerm	0x000003F2	A user without owner permission attempted to create a mailbox.
ecServerPaused	0x0000047F	The client has attempted too many logon attempts, each of which resulted in an ecWrongServer response. <a href="#">&lt;8&gt;</a>

#### 2.2.1.2 RopGetReceiveFolder Semantics

The syntax of the [RopGetReceiveFolder](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.2.

[RopGetReceiveFolder](#) is used to determine the delivery folder for messages of a specific message class when they are delivered to a mailbox. This ROP tests the message class string and returns the folder ID (FID) where messages of that class and all subclasses will be delivered. This ROP also returns the specific parent message class configured to deliver to that folder.

##### 2.2.1.2.1 Request

This operation is only valid when the Logon object refers to a private mailbox logon.

### 2.2.1.2.1.1 MessageClass

Contains the message class string to test. The string includes the terminating NULL character. Testing MUST be done case-insensitive. The string MUST meet the following requirements:

- The string uses ASCII encoding.
- The length (including the terminating NULL character) is greater than zero and less than or equal to 255.
- Each character value in the string is in the numeric range of 32 to 126, inclusive.
- The string does not begin with a period (".").
- The string does not end with a period.
- The string does not contain adjacent periods.

### 2.2.1.2.2 Response

#### 2.2.1.2.2.1 FolderId

The FID of the folder where messages of **class MessageClass** will be delivered. The folder MUST be a folder within the user's mailbox.

#### 2.2.1.2.2.2 ExplicitMessageClass

The message class string actually configured for delivery to the folder. The string includes the terminating NULL character. The string MUST meet the following requirements:

- The string uses ASCII encoding.
- The length (including the terminating NULL character) is greater than zero and less than or equal to 255.
- Each character value in the string is in the numeric range of 32 to 126, inclusive.
- The string does not begin with a period (".").
- The string does not end with a period.
- The string does not contain adjacent periods.

The server can return the message class string as originally configured by the client, converted into all upper case, or all lower case. The server MUST return the actual configured message class that is the longest prefix string of the **MessageClass** field (sent in the request). For more details about how the server determines the actual configured message class, see section [3.2.5.2](#).

For example, if the client sends a **MessageClass** of "IPM.Schedule.Meeting.Request", the **ExplicitMessageClass** might be returned as "IPM.Schedule.Meeting", which implies that all messages that share the prefix string (or are a subclass of) "IPM.Schedule.Meeting" will be delivered to this folder. In this same example, if a client sends a **MessageClass** of "IPM.Schedule.Meeting", then the string "IPM.Schedule.Meeting" will be returned in the **ExplicitMessageClass** field.

As a second example, suppose that the client sends a request with either "MY.Class" or "" (an empty string) in the **MessageClass** field. In both cases, the longest prefix substring match is the empty

string. Therefore, the server will return the FID for the user's Inbox and an empty string. If the client requests "IPM.MY.Class", then server will return the FID for the Inbox and "IPM".

As a third example, suppose that the client creates a folder and then uses **RopSetReceiveFolder** to register the message class "MY.Class". If the client queries for the message class "MY.Class.SOMETHING", then the server will return the FID registered for "MY.Class" (in the **FolderId** field) and the string "MY.Class" (in the **ExplicitMessageClass** field).

### 2.2.1.2.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecInvalidParam	0x80070057	The <b>MessageClass</b> value does not conform to the format requirements specified in section <a href="#">2.2.1.2.1.1</a> .
ecNoReceiveFolder	0x00000463	There is no configured <b>Receive folder</b> that exactly matches, nor is a prefix string of the <b>MessageClass</b> value.
ecNotSupported	0x80040102	The ROP was not performed against a private mailbox logon.

### 2.2.1.3 RopSetReceiveFolder Semantics

The syntax of the [RopSetReceiveFolder](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.3.

[RopSetReceiveFolder](#) is used to establish the delivery folder for messages that have a message class string that itself has a prefix of a given string. The request includes a message class string and an FID. As a result, all messages with a message class that matches this given class will be delivered to the folder identified by the FID. Message class matches are determined using a case-insensitive prefix string match. For example, a configured class of "xx.yy" will match a message with a class of "Xx.YY.ZZ".

Multiple message classes are permitted to deliver to the same folder. A client can change an existing receive folder configuration for a message class by simply issuing this ROP with a different value in the **FolderId** field.

The server MUST record, in the UTC time zone, the time the entry was created or modified so that it can be retrieved by using the [RopGetReceiveFolderTable](#) ROP.

#### 2.2.1.3.1 Request

This operation MUST be issued against a private mailbox logon.

##### 2.2.1.3.1.1 FolderId

Contains the FID of the desired delivery folder for the **MessageClass** class and all non-specifically configured subclasses of that class. A value of all zeros means the server MUST remove any previously configured entry for the given message class.

### 2.2.1.3.1.2 MessageClass

Contains the string identifying the **message class** whose delivery folder is being set. The string includes the terminating NULL character. The string MUST comply with all of the following restrictions:

- The string uses ASCII encoding.
- The length (including the terminating NULL character) is greater than zero and less than or equal to 255.
- Each character value in the string is in the numeric range of 32 to 126, inclusive.
- The string does not begin with a period (".").
- The string does not end with a period.
- The string does not contain adjacent periods.

The message class string is compared, case-insensitive, to all existing configured entries. Prefix string comparisons are not performed. If an existing entry matches, the new FID (specified in the **FolderId** field) replaces the currently configured value. Otherwise, a new entry is added.

### 2.2.1.3.2 Response

There are no additional fields other than the **ReturnValue** for this operation.

### 2.2.1.3.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecAccessDenied	0x80070005	The client has attempted to change the receive folder for the "IPM" or "Report.IPM" classes.
ecInvalidParam	0x80070057	The message class string does not conform to the requirements specified in section <a href="#">2.2.1.3.1.2</a> .
ecError	0x80004005	The FID (specified in the <b>FolderId</b> field) is all zeros AND the message class string (specified in the <b>MessageClass</b> field) has a length of zero.
ecNotSupported	0x80040102	The ROP was not performed against a private mailbox logon.

### 2.2.1.4 RopGetReceiveFolderTable Semantics

The syntax of the [RopGetReceiveFolderTable](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.4.

[RopGetReceiveFolderTable](#) is used to obtain a comprehensive list of all configured message classes to delivery folder entries. The return data consists of logical "rows" of data, each row consisting of three "columns" of values. There is one row for each configured entry, and within each row are the message class, FID and last modification time for the entry.

#### 2.2.1.4.1 Request

There are no explicit fields for this operation. The data to retrieve is limited to the mailbox linked to the Logon object passed as part of the normal **ROP request** process. This operation MUST be issued against a private mailbox logon.

#### 2.2.1.4.2 Response

##### 2.2.1.4.2.1 RowCount

The number of rows in the **table**. The rows themselves can be returned in any order.

##### 2.2.1.4.2.2 Rows

An array that contains the rows of the Receive folder table. Each row is returned in either a **StandardPropertyRow** structure or a **FlaggedPropertyRow** structure, both of which are specified in [\[MS-OXCDATA\]](#) sections [2.9.1.1](#) and [2.9.1.2](#), respectively. The value of each structure's **Flag** field indicates which structure is being used: 0x00 for the **StandardPropertyRow** structure; 0x01 for the **FlaggedPropertyRow** structure.

The **ValueArray** field of either **StandardPropertyRow** or **FlaggedPropertyRow** MUST include only the following properties, in the order given, and no other properties.

1. [PidTagFolderId](#) property — A **PtypInteger64** value that specifies the folder ID (FID) of the Receive folder, which is the folder to which messages of the specified message class will be delivered. The Receive folder MUST be a folder that is within the user's mailbox.
2. [PidTagMessageClass](#) property — A **PtypString** value that specifies the message class that is configured for the Receive folder. The string can be all upper case, all lower case, or as originally stored by the client. The string includes the terminating NULL character and MUST meet the following requirements:
  - The string uses ASCII encoding.
  - The length (including the terminating NULL character) is greater than zero and less than or equal to 255.
  - Each character value in the string is in the numeric range of 32 to 126, inclusive.
  - The string does not begin with a period (".").
  - The string does not end with a period.
  - The string does not contain adjacent periods.
3. [PidTagLastModificationTime](#) property — A **PtypTime** value that specifies the time, in Coordinated Universal Time (UTC), when the server created or last modified the row in the Receive folder table.

#### 2.2.1.4.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecNoReceiveFolder	0x00000463	There are no configured receive folder entries.
ecNotSupported	0x80040102	The ROP was not performed against a private mailbox logon.

### 2.2.1.5 RopGetStoreState Semantics

The syntax of the [RopGetStoreState](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.5.

[RopGetStoreState](#) is used to obtain **state** information about the current mailbox. [<9>](#)

#### 2.2.1.5.1 Request

There are no explicit fields for this operation. The data to be retrieved is limited to the mailbox that is linked to the **LogonID** that is passed as part of the ROP request, as specified in [\[MS-OXCROPS\]](#) section 2.2.2.5.1. This operation MUST be issued against a private mailbox logon.

#### 2.2.1.5.2 Response

##### 2.2.1.5.2.1 StoreState

If the mailbox currently has any active search folders, this bit field MUST have the STORE\_HAS\_SEARCHES flag set. All other bits MUST NOT be set.

##### 2.2.1.5.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecNotSupported	0x80040102	The ROP was not performed against a private mailbox logon.

### 2.2.1.6 RopGetOwningServers Semantics

The syntax of the [RopGetOwningServers](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.6.

[RopGetOwningServers](#) is used to obtain the set of servers that host content for a replicated public folder.

When attempting to read content from a public folder on a specific server, the operation can fail with ecNoReplicaHere (0x00000468). This happens if the server that receives the request does not contain a replica copy of the data. In that event, the client issues this ROP to obtain the set of servers that do contain the data. The returned data is an ordered set of server names, sorted by the configured network costs that connect this server to each of the other servers. The cost data can

come from a specific configuration for this server, or can be inferred from other network configuration settings (specific site router costs, for example).<10>

### 2.2.1.6.1 Request

This operation SHOULD be issued against a public folders logon.

#### 2.2.1.6.1.1 FolderId

Contains the FID of the public folder for which to obtain the replica set server names.

### 2.2.1.6.2 Response

#### 2.2.1.6.2.1 OwningServersCount

Identifies the number of strings contained in the **OwningServers** field.

#### 2.2.1.6.2.2 CheapServersCount

Identifies the number of entries at the front of the list that have the same lowest network cost. This value MUST be less than or equal to **OwningServersCount** and MUST be greater than zero if **OwningServersCount** is greater than zero.

#### 2.2.1.6.2.3 OwningServers

Contains an array of ASCII strings. Each entry includes the terminating NULL character. The number of strings MUST be equal to **OwningServersCount**. The entries are sorted by the server's interpretation of the network cost to connect to each of the servers in the list. The source of these network costs can be whatever configuration source the server finds most appropriate.

Each string is the ESSDN of a public folder database that hosts an **active replica** of the content of the folder. Folders can exist in one of several **replica** states, but only those in the Active state are returned. All replica states except Active are implementation-specific.

The server can remove an active replica from the list if it deems it "too expensive" for the client to attempt a connection, or if other configuration settings have identified a particular server as unavailable to clients for some reason. <11> "Too expensive" is defined by the implementation.

### 2.2.1.6.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecNoReplicaAvailable	0x00000469	There are no active replicas for the folder OR the only available replicas have been deemed "too expensive" to reach or are otherwise deemed "unavailable" by the server implementation.
ecNotFound	0x8004010F	The FID could not be found in the public folder database.

### 2.2.1.7 RopPublicFolderIsGhosted Semantics

The syntax of the [RopPublicFolderIsGhosted](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.7.

[RopPublicFolderIsGhosted](#) is used to obtain the replication state for a folder on the current server. Folders can exist in one of several replica states, but all states except the Active state are implementation-specific. The ROP returns TRUE if the server is not an active replica for the folder and FALSE if the server is an active replica.

#### 2.2.1.7.1 Request

This operation SHOULD only be issued against a public folders logon. The server MUST always return FALSE in the **IsGhosted** field if the client issues this operation against a private mailbox logon (for any folder), or the **IPM subtree** or the **non-IPM subtree** root folders of the public store.

##### 2.2.1.7.1.1 FolderId

Contains the FID of the public folder for which to obtain the ghosted state.

#### 2.2.1.7.2 Response

##### 2.2.1.7.2.1 IsGhosted

Contains a Boolean property. TRUE if the server is not an active replica of the folder; otherwise, FALSE. Further return values are included only if **IsGhosted** is TRUE.

##### 2.2.1.7.2.2 ServersCount

Identifies the number of strings contained in the **Servers** field.

##### 2.2.1.7.2.3 CheapServersCount

Identifies the number of entries at the front of the list that have the same lowest network cost. This value MUST be less than or equal to **ServersCount** and MUST be greater than zero if **ServersCount** is greater than zero.

##### 2.2.1.7.2.4 Servers

Contains an array of ASCII strings. Each entry includes the terminating NULL character. The number of strings MUST be equal to **ServersCount**. The entries are sorted by the server's interpretation of the network cost to connect to each of the servers in the list. The source of these network costs can be whatever configuration source the server finds most appropriate.

Each string is the ESSDN of a public folder database that itself hosts an active replica of the content of the folder. Folders can exist in one of several replica states, but only those in the Active state are returned. All replica states except Active are implementation-specific.

The server can remove an active replica from the list if it deems it "too expensive" for the client to attempt a connection, or if other configuration settings have identified a particular server as unavailable to clients for some reason. [<12>](#) "Too expensive" is defined by the implementation.



### 2.2.1.7.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecNoReplicaAvailable	0x00000469	There are no active replicas for the folder OR the only available replicas have been deemed "too expensive" to reach or are otherwise deemed "unavailable" by the server implementation. This error can only occur if the server itself is not an active replica.
ecNotFound	0x8004010F	The FID could not be found in the public folder database.

### 2.2.1.8 RopLongTermIdFromId Semantics

The syntax of the [RopLongTermIdFromId](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.8.

[RopLongTermIdFromId](#) is used to obtain the LongTermID, given the **ShortTermID** (FID or **MID**). For more details about LongTermIDs, see [\[MS-OXCDATA\]](#) section 2.2.1.3.1. For more details about ShortTermIDs, see [\[MS-OXCDATA\]](#) section 2.2.1.1 or [2.2.1.2](#).

#### 2.2.1.8.1 Request

##### 2.2.1.8.1.1 ObjectID

Contains the ShortTermID to map to a LongTermID. The 16-bit REPLID portion of the ID MUST be a valid entry in the REPLID and REPLGUID to-and-from mapping table.

#### 2.2.1.8.2 Response

##### 2.2.1.8.2.1 LongTermId

The same ID, with the REPLID mapped to the associated REPLGUID. The server MUST map the same REPLID to the same REPLGUID every time it is queried. Other servers can map a particular REPLID to a different REPLGUID than this server would, but they too MUST map any particular REPLID to the same REPLGUID value every time they are queried.

### 2.2.1.8.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecNotFound	0x8004010F	The REPLID portion of the ID could not be found in the REPLID and REPLGUID to-and-from mapping table.

### 2.2.1.9 RopIdFromLongTermId Semantics

The syntax of the [RopIdFromLongTermId](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.9.

[RopIdFromLongTermId](#) is used to obtain the ShortTermID, given the LongTermID.

#### 2.2.1.9.1 Request

##### 2.2.1.9.1.1 LongTermId

Contains the LongTermID to map to a ShortTermID. If the REPLGUID portion of the ID is already present in the REPLID and REPLGUID to-and-from mapping table, the associated REPLID is used to form the return value. If the REPLGUID is not present in the mapping table, a new entry is added, and the newly assigned REPLID is used to form the return value.

#### 2.2.1.9.2 Response

##### 2.2.1.9.2.1 ObjectId

The same ID, with the REPLGUID mapped to the associated REPLID. The server MUST map the same REPLGUID to the same REPLID every time it is queried. Other servers can map a particular REPLGUID to a different REPLID than this server would, but they too MUST map any particular REPLGUID to the same REPLID value every time they are queried.

##### 2.2.1.9.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCADATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.

### 2.2.1.10 RopGetPerUserLongTermIds Semantics

The syntax of the [RopGetPerUserLongTermIds](#) request and response is specified in the [\[MS-OXCROPS\]](#) section 2.2.2.10.

[RopGetPerUserLongTermIds](#) is used to obtain the LongTermIDs of folders in a public folders store that contain per-user read/unread data identified by a REPLGUID.

#### 2.2.1.10.1 Request

This ROP MUST be issued against a logon that was made to a private mailbox.

##### 2.2.1.10.1.1 DatabaseGuid

Identifies the replica database for which the client is querying against. This GUID is obtained from the result of a [RopLogon](#) issued against a public store. For more details, see section [2.2.1.1.3.6](#).

## 2.2.1.10.2 Response

### 2.2.1.10.2.1 LongTermIdCount

Identifies the number of entries in the following array. This field can be set to zero.

### 2.2.1.10.2.2 LongTermIds

Contains an array of longTermIDs of folders in the public store for which this user has cached read/unread information. The number of items in this array MUST be the value of the **LongTermIdCount** field.

### 2.2.1.10.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecNotSupported	0x80040102	The ROP was attempted against a public folders logon.

### 2.2.1.11 RopGetPerUserGuid Semantics

The syntax of the [RopGetPerUserGuid](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.11.

[RopGetPerUserGuid](#) is used to obtain the REPLGUID of the cached per-user read/unread data for a specific public folder. The returned GUID value allows the client to correlate the cached data with whatever replica server the client is currently communicating with. Typically, if the cached GUID value does not match the current REPLGUID the client is logged on to for public folder access, it means that the client has been referred to a different server from the one it last cached the data from. The client would then issue a [RopWritePerUserInformation](#) request with the locally cached data so that per-user read/unread information on the new replica will now match what the user last saw when connected to the old replica. For details about [RopWritePerUserInformation](#), see section [2.2.1.13](#).

#### 2.2.1.11.1 Request

This ROP MUST be issued against a logon that was made to a private mailbox.

##### 2.2.1.11.1.1 LongTermId

Contains the LongTermID of the folder to query.

#### 2.2.1.11.2 Response

##### 2.2.1.11.2.1 DatabaseGuid

Contains the REPLGUID of the last database for which relevant read/unread information was obtained. This GUID is obtained from the result of a [RopLogon](#) issued against a public store. For more details, see section [2.2.1.1.4.5](#).

### 2.2.1.11.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecNotSupported	0x80040102	ROP was attempted against a public folders logon.
ecNotFound	0x8004010F	The public folder identified by the value of the LongTermID field could not be found

### 2.2.1.12 RopReadPerUserInformation Semantics

The syntax of the [RopReadPerUserInformation](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.12.

[RopReadPerUserInformation](#) is used to obtain a set of **change numbers (CN)**, each of which is associated with a message that the user has read in a specific folder. Note, this is not a set of message IDs (MIDs), but rather the aggregated values of the [PidTagChangeNumber](#) property ([\[MS-OXCFXICS\]](#) section 2.2.1.2.3) of the messages at the time they were marked as read. Messages that are modified receive a new change number (CN) and hence fall out of the set of read messages. The user will see these modified messages marked as unread. If the user marks a message as read, the current value of that message's [PidTagChangeNumber](#) property is added to the set. If the user marks a message as unread, the current value of that message's [PidTagChangeNumber](#) property is removed from the set.

The **change number set (CNSET)** is serialized into a **binary large object (BLOB)** that is formatted as a serialized **IDSET** with REPLGUIDs, as specified in [\[MS-OXCFXICS\]](#) section 2.2.2.3.2.

The size of the data to be returned can potentially exceed the maximum amount of data that can be communicated in a single ROP. For this reason, the ROP is designed to stream the data to the client by having the client invoke this ROP multiple times. Because the server can cache interim data across client calls, the client MUST complete the entire streaming operation for the data of one folder before commencing streaming operations for another folder on the same server logon. The server cannot distinguish between a client choosing to abort reading data from one folder before commencing reading from another versus doing this by accident. In the event the client does not properly prevent simultaneous access, the server can return data to the client that's potentially confusing and that could lead to corrupted data.

When this ROP is issued against a private mailbox logon, cached information for the folder is retrieved. When issued against a public folders logon, the current read/unread information for a folder is retrieved.

The client can use this ROP in conjunction with [RopWritePerUserInformation](#), which is specified in section [2.2.1.13](#), to move read/unread information from one public folder replica to another. For example, the client could, periodically or on a specific user action, query the public logon for read/unread information for a specific public folder by issuing a [RopReadPerUserInformation](#) request. It would then issue a [RopWritePerUserInformation](#) request against the private mailbox logon, sending the same data back to the server. This effectively saves the read/unread data in the user's mailbox. Later, when the user re-visits the public folder, the client would issue a [RopReadPerUserInformation](#) request against the private mailbox logon to retrieve the cached information for the folder. It would then issue a [RopWritePerUserInformation](#) request to the public

folders logon to save back to the public database. This sequence of operations allows the user to see the same set of unread messages as the last time they visited the folder, even in the event that the client is referred to a different public folder server each time they log on.

#### 2.2.1.12.1 Request

##### 2.2.1.12.1.1 FolderId

Contains the LongTermID of the folder to query.

##### 2.2.1.12.1.2 Reserved

This value MUST be 0x00 and is ignored by the server.

##### 2.2.1.12.1.3 DataOffset

Identifies the offset into the stream of data. This value is the position of the first byte of data to be returned. The value MUST be greater than or equal to zero.

The client MUST NOT set **DataOffset** to an arbitrary value. The value MUST be zero in the first [RopReadPerUserInformation](#) request. If subsequent requests are necessary to retrieve all the data, then the client MUST update **DataOffset** by adding to it the value of the **DataSize** field of the previous [RopReadPerUserInformation](#) response. In other words, if **HasFinished** equals FALSE, then **DataOffset** MUST be updated, as follows, after each [RopReadPerUserInformation](#) response.

$$\text{DataOffset} = \text{DataOffset} + \text{DataSize}$$

##### 2.2.1.12.1.4 MaxDataSize

Identifies the maximum amount of data to be returned to the client in a single [RopReadPerUserInformation](#) response. The server can return less than the requested maximum size. The client can set **MaxDataSize** to zero, which indicates to the server that a default value MUST be used as the maximum size. When multiple [RopReadPerUserInformation](#) requests are necessary to retrieve all of the data, the client can set **MaxDataSize** to a different value in each invocation of the ROP.

#### 2.2.1.12.2 Response

##### 2.2.1.12.2.1 HasFinished

Indicates whether this is the last block of data to be returned. The client SHOULD NOT issue another [RopReadPerUserInformation](#) for the same folder. This value MUST be TRUE if the underlying data has not changed since the last successful download.

##### 2.2.1.12.2.2 DataSize

Contains the size, in bytes, of the data being returned. MUST be less than or equal to **MaxDataSize**. This value MUST be zero if the underlying data has not changed since the last successful download.

### 2.2.1.12.2.3 Data

Contains the actual data being returned. The size MUST be equal to **DataSize**. The client is not expected to interpret this data in any way, but simply provide it unaltered in a future sequence of invocations of [RopWritePerUserInformation](#).

### 2.2.1.12.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecRpcFormat	0x000004B6	The <b>DataOffset</b> value was less than zero.
ecError	0x80004005	The <b>DataOffset</b> value was greater than the data size.
ecNotFound	0x8004010F	The public folder identified by <b>FolderId</b> could not be found.

### 2.2.1.13 RopWritePerUserInformation Semantics

The syntax of the [RopWritePerUserInformation](#) request and response is specified in [\[MS-OXCROPS\]](#) section 2.2.2.13.

[RopWritePerUserInformation](#) is used to establish the set of change numbers of messages the user has read in a specific folder. Note, this is not a set of MIDs, but rather the change numbers of the messages at the time they were read. Messages that are modified receive a new change number, and hence, fall out of the set of read messages. The user will see these modified messages marked as unread.

The format of a serialized change number set is identical to the format of a serialized IDSET with REPLGUIDs and is specified in [\[MS-OXCFXICS\]](#) section 2.2.2.3.2.

The size of the data can potentially exceed the maximum amount of data that can be communicated in a single ROP. For this reason, the ROP is designed to stream the data to the server by having the client invoke this ROP multiple times. Because the server can cache interim data across client calls, the client MUST complete the entire streaming operation for the data of one folder before commencing streaming operations for another folder on the same server logon. The server MUST dispose of partial data if it detects the client has changed target folders before indicating to the server that the first folder's data is complete.

When this ROP is issued against a private mailbox logon, cached information for the folder is saved. When issued against a public folders logon, the current read/unread information is established.

Used in conjunction with [RopReadPerUserInformation](#), the client is able to move read/unread information from one public folder replica to another. For example, the client could, periodically or on a specific user action, query the public logon for read/unread information for a specific public folder by issuing a [RopReadPerUserInformation](#) request. It would then issue a [RopWritePerUserInformation](#) request against the private mailbox logon, sending the same data to the mailbox server. This effectively saves the read/unread data in the user's mailbox. Later, when the user re-visits the public folder, the client would issue a [RopReadPerUserInformation](#) request against the private mailbox logon to retrieve the cached information for the folder. It would then issue a [RopWritePerUserInformation](#) request to the public folders logon to save back onto the public

database. This sequence of operations allows the user to see the same set of unread messages as the last time they visited the folder, even in the event that the client is referred to a different public folder server each time they log on.

### 2.2.1.13.1 Request

#### 2.2.1.13.1.1 FolderId

Contains the LongTermID of the folder for which data is being saved.

#### 2.2.1.13.1.2 HasFinished

Indicates whether this is the last block of data to be written. When the client issues this ROP with this value set to TRUE, the server MUST validate the data before committing it to storage.

#### 2.2.1.13.1.3 DataOffset

Identifies the offset into the stream where this block of data is to be written. This value MUST be equal to the total size of the data previously written.

#### 2.2.1.13.1.4 DataSize

Identifies the size, in bytes, of the data to be written.

#### 2.2.1.13.1.5 Data

Contains the data to write. The size MUST be equal to **DataSize**.

#### 2.2.1.13.1.6 ReplGuid

MUST NOT be present for operations against public folders logons. This field MUST be present when the value of the **DataOffset** field is zero. This field MUST NOT be present when **DataOffset** is not zero. Identifies which public database was the source of this data. The value is the REPLGUID of the last database for which relevant read/unread information was obtained. This GUID is obtained from the result of a [RopLogon](#) issued against a public store. For more details, see section [2.2.1.13.6](#).

### 2.2.1.13.2 Response

There are no fields other than the **ReturnValue** field for this ROP.

#### 2.2.1.13.3 ReturnValue

All ROPs have an error return code. Upon error-free return, this return code MUST be zero. Additional output values then follow. The most common error return values are provided in the following table. Other possible error codes are specified in [\[MS-OXCDATA\]](#) section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecError	0x80004005	The <b>DataOffset</b> didn't match the size of the data written so far OR the <b>FolderId</b> didn't match the value on the previous call, AND THEN <b>DataOffset</b> wasn't zero.

Name	Value	Meaning
ecFmtError	0x000004ED	The data cumulatively written could not be parsed as a proper serialized IDSET with REPLGUIDs, as specified in <a href="#">[MS-OXCFXICS]</a> section 2.2.2.3.2.
ecNotFound	0x8004010F	The public folder identified by <b>FolderId</b> could not be found.

## 2.2.2 Logon-Specific Properties

The following properties are available on Logon objects. A Logon object is obtained by issuing a [RopLogon](#) request, and receiving a successful response. Some logon properties are read-only. Some logon properties are write-only. Some properties can be deleted by the client. Some properties are available only on public folder logons. Some properties are available only on private mailbox logons. When the client attempts to read a non-readable property, write a non-writable property, delete a non-deletable property, manipulate private-only properties on a public logon, or manipulate public-only properties on a private logon, the server MUST return ecPropSecurityViolation (0x80070005) for the property. For details about how error codes are returned for [RopSetProperties](#), [RopGetPropertiesSpecific](#), and [RopDeleteProperties](#) see [\[MS-OXCROPS\]](#) and [\[MS-OXCPRPT\]](#).

To read any of the readable properties, issue a [RopGetPropertiesSpecific](#) ROP with the Logon object obtained from a successful invocation of [RopLogon](#). To write any of the writable properties, issue a [RopSetProperties](#) ROP with the Logon object obtained from a successful invocation of [RopLogon](#). To delete any of the deletable properties, issue [RopDeleteProperties](#) with the Logon object obtained from a successful invocation of [RopLogon](#). For more details about how to issue [RopSetProperties](#), [RopGetPropertiesSpecific](#), or [RopDeleteProperties](#) requests, see [\[MS-OXCROPS\]](#) and [\[MS-OXCPRPT\]](#).

### 2.2.2.1 Private Mailbox Logon

The following table lists the properties that are available on a private mailbox logon. The operations (GET/read, SET/write, delete) that are allowed on each property are indicated by an "X" in the appropriate column. For details about these operations, see section [2.2.2](#).

Name	Get	Set	Delete
<a href="#">PidTagExtendedRuleSizeLimit</a>	X		
<a href="#">PidTagMaximumSubmitMessageSize</a>	X		
<a href="#">PidTagProhibitReceiveQuota</a>	X		
<a href="#">PidTagProhibitSendQuota</a>	X		
<a href="#">PidTagStoreState</a>	X		
<a href="#">PidTagComment</a>	X	X	
<a href="#">PidTagContentCount</a>	X		
<a href="#">PidTagDeleteAfterSubmit</a>	X	X	X
<a href="#">PidTagDisplayName</a>	X	X	
<a href="#">PidTagMailboxOwnerEntryId</a>	X		
<a href="#">PidTagMailboxOwnerName</a>	X		
<a href="#">PidTagMessageSize</a>	X		



Name	Get	Set	Delete
<a href="#">PidTagMessageSizeExtended</a>	X		
<a href="#">PidTagOutOfOfficeState</a>	X	X	
<a href="#">PidTagUserEntryId</a>	X		
<a href="#">PidTagSentMailSvrEID</a>	X	X	X
<a href="#">PidTagCodePageId</a>		X	
<a href="#">PidTagLocaleId</a>		X	
<a href="#">PidTagSortLocaleId</a>		X	

The following table describes each of the properties that are available on a private mailbox logon.

Name	Description
<a href="#">PidTagExtendedRuleSizeLimit</a>	Maximum size, in bytes, the user is allowed to accumulate for a single "extended" rule. For details of extended rules, see <a href="#">[MS- OXORULE]</a> section 2.2.4.
<a href="#">PidTagMaximumSubmitMessageSize</a>	Maximum size, in kilobytes, of a message a user is allowed to submit for transmission to another user. An unset value or a value of -1 indicates that there is no limit.
<a href="#">PidTagProhibitReceiveQuota</a>	Maximum size, in kilobytes, a user is allowed to accumulate in their mailbox, before no further mail will be delivered. An unset value or a value of -1 indicates that there is no limit.
<a href="#">PidTagProhibitSendQuota</a>	Maximum size, in kilobytes, a user is allowed to accumulate in their mailbox, before the user can no longer submit any more mail. An unset value or a value of -1 indicates that there is no limit.
<a href="#">PidTagStoreState</a>	For a full description of the <b>StoreState</b> field, see section <a href="#">2.2.1.1.3.9</a> .
<a href="#">PidTagComment</a>	A mailbox comment.
<a href="#">PidTagContentCount</a>	Cumulative count of non- <b>FAI</b> messages in the mailbox.
<a href="#">PidTagDeleteAfterSubmit</a>	Indicates whether a transport deletes all submitted mail after transmission. An unset value or a value of FALSE indicates that the mail is not deleted.
<a href="#">PidTagDisplayName</a>	The mailbox display name.
<a href="#">PidTagMailboxOwnerEntryId</a>	The <b>EntryID</b> in the <b>Global Address List (GAL)</b> of the owner of the mailbox.
<a href="#">PidTagMailboxOwnerName</a>	Display name of the owner of the mailbox.
<a href="#">PidTagMessageSize</a>	Cumulative size, in bytes, of all content in the mailbox. Value is limited to 32 bits and becomes undefined if the content size exceeds four gigabytes.
<a href="#">PidTagMessageSizeExtended</a>	Cumulative size, in bytes, of all content in the mailbox.

Name	Description
<a href="#">PidTagOutOfOfficeState</a>	Indicates whether the user is OOF. Setting to FALSE causes accumulated OOF history maintained by the rules engine to be cleared for all folders and OOF rules. Setting to TRUE allows rules set to run only when the user is out of the office to run. For more details, see [MS-OXORULE].
<a href="#">PidTagUserEntryId</a>	Address book EntryID of the user logged on to the mailbox.
<a href="#">PidTagSentMailSvrEID</a>	The structure identifying the Sent Items folder. An unset value indicates that the server won't move sent items to a Sent Items folder after transmission. For more details, see [MS-OXCDATA] section 2.13.1.2.
<a href="#">PidTagCodePageId</a>	Establishes the client code page for <b>Unicode</b> to <b>double-byte character set (DBCS)</b> string conversion. For details, see [MS-UCODEREF] section 2.2.1.
<a href="#">PidTagLocaleId</a>	Establishes the language locale for translating system-generated messages, such as delivery reports. For more details, see [MS-LCID].
<a href="#">PidTagSortLocaleId</a>	Establishes the language locale for sorting the contents of tables. For more details, see [MS-LCID].

### 2.2.2.2 Public Folders Logon

The following table lists the properties that are available on a public folders logon. The operations (GET/read, SET/write, delete) that are allowed on each property are indicated by an "X" in the appropriate column. For details about these operations, see section 2.2.2.

Name	Get	Set	Delete
<a href="#">PidTagUserEntryId</a>	X		
<a href="#">PidTagAddressBookMessageId</a>	X		

The following table describes each of the properties that are available on a public folders logon.

Name	Description
<a href="#">PidTagUserEntryId</a>	Address book EntryID of the user logged on to the public folders.
<a href="#">PidTagAddressBookMessageId</a>	Short-term MID of the first message in the local site's offline address book public folder, if it exists and has a <b>local replica</b> . The property <b>MUST</b> have an error value of ecNotFound (0x8004010F) if there is no local site offline address book public folder, the server can't open the folder, the server can't access the message, or there is no local replica of the folder.

## 3 Protocol Details

### 3.1 Client Details

None.

#### 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 client SHOULD maintain a cached copy of the mapping between REPLIDs and REPLGUIDs used for ShortTermIDs and longTermIDs.

The client SHOULD maintain a cache of per user data currently stored in the private store. This enables the client to only sync per user data when a change has been made.

#### 3.1.2 Timers

None.

#### 3.1.3 Initialization

None.

#### 3.1.4 Higher-Layer Triggered Events

##### 3.1.4.1 Logging on to a Store

The client logs on a store using [RopLogon](#) before attempting to do any additional ROPs on the store.

When performing a [RopLogon](#), the client MUST generate a logon ID to be used in the request (see [\[MS-OXCROPS\]](#) section 2.2.2.1 for more details on the ROP request). This logon ID is then used in all subsequent ROP requests that use this logon (see [\[MS-OXCROPS\]](#) for more details). This value MUST be unique per RPC **Session Context Handle (CXH)** (per call to the **EcDoConnectEx** method). For more details about **EcDoConnectEx**, see [\[MS-OXCRPC\]](#) section 3.1.4.11.

If the server returns an error code of `ecWrongServer`, the client SHOULD re-attempt the logon by setting the `ESSDN` field to the value returned by the **ServerName** field in the redirect response. This requires creating a new RPC connection to the server specified by the **ServerName** field and re-attempting the logon using that connection. For details about creating the RPC connection, see [\[MS-OXCRPC\]](#).

If the server returns an error code of `ecUnknownUser` or `ecLoginFailure`, the client SHOULD use the Autodiscover HTTP Service protocol [\[MS-OXDISCO\]](#) in order to attempt to retrieve updated user and server information. [<13>](#) If successful, the client can attempt to logon again by releasing the previous RPC connection and creating a new RPC connection with the information supplied by the Autodiscover HTTP Service protocol. If the client is not successful at retrieving updated information or no changes are detected, the client MUST fail the logon.

If the client is unable to establish an RPC connection to a public folder store, it can request a redirection to an alternative public folder store from the private store. The client can use an existing RPC connection to the private store, or create a new one. To request a redirection to an alternative public folder store the client issues a [RopLogon](#) request to the private store. The client MUST set the ALTERNATE\_SERVER flag in the OpenFlags field of the [RopLogon](#) request. The logon request will return ecWrongServer and redirect the client to an alternate server. When issuing the logon request to the alternate server, the client MUST clear the ALTERNATE\_SERVER flag and set the IGNORE\_HOME\_MDB flag in OpenFlags.

If the RPC session to the server is lost and then reconnected (see [MS-OXCRPC] for more details on RPC connections), the existing logon is invalid. The client MUST logon again by calling [RopLogon](#) (the client can reuse the existing LogonID). Additionally, all objects (folders, messages, tables) opened using the original logon are now invalid and MUST be re-opened.

After logging on, the client SHOULD cache the REPLGUID. If a reconnect occurs, the new REPLGUID returned by [RopLogon](#) MUST be compared to the cached value. If the GUIDs are different, the client MUST dispose of all local caches of server information. This includes any open Server objects, caches of data mappings, or caches of special FIDs. The effect MUST be similar to actually exiting the client application, and restarting from the beginning of the process.

### 3.1.4.2 Converting Between LongTermIDs and ShortTermIDs

ShortTermIDs use a 16-bit REPLID in place of a REPLGUID. ShortTermIDs MUST NOT be persisted in any storage that could be accessed on a different logon session for the same mailbox. Any ShortTermIDs cached in non-persistent storage MUST be forgotten (deleted from non-persistent storage) if the client reconnects to the server, issues a [RopLogon](#), and the return value of REPLGUID is different than the value obtained from a previous [RopLogon](#). To persist IDs in any long-term storage, the client MUST first convert the ID to a LongTermID.

The client MUST use [RopLongTermIdFromId](#) to convert a short-term ID into a long-term ID. This ROP succeeds only if the REPLID portion of the given short-term ID exists in the REPLID and REPLGUID to-and-from mapping table. For more details about how the server processes [RopLongTermIdFromId](#), see section [3.2.5.8](#).

The client MUST use [RopIdFromLongTermId](#) either to create a short-term ID, given a long-term ID, or to convert a long-term ID into a short-term ID. If the REPLGUID portion of the long-term ID exists in the REPLID and REPLGUID to-and-from mapping table, then the associated REPLID is returned; otherwise, a new REPLID is created, the given REPLGUID and the new REPLID are added to the table, and the new REPLID is returned to the client. For more details about how the server processes [RopIdFromLongTermId](#), see section [3.2.5.9](#).

### 3.1.4.3 Syncing Per-User Read/Unread Data for Public Folders

Public folder data is replicated across multiple servers, with each server maintaining per-user read/unread data for each folder. The read/unread information is valid for that server only. If a subsequent logon results in the client being redirected to a different replica server, it is the client's responsibility to synchronize the current read/unread data to the new server.

For each folder, the client issues a [RopReadPerUserInformation](#) against the public store (this is not necessary if the folder has not been modified) to retrieve the per-user read/unread data. This data is then stored in the private store using [RopWritePerUserInformation](#). While the goal of these operations is to keep per-user data from the public store in sync with the private store, the actual time when this operation takes place is up to the client (every time an item is marked read/unread, whenever a folder switch occurs, whenever the store is released).

When a folder is subsequently reopened in a later logon session, the client MUST check to see if the replica server has changed. This is done by issuing a [RopGetPerUserGuid](#) against the private store and comparing the REPLGUID returned (in the **DatabaseGuid** field) to the public store's REPLGUID, which is returned by [RopLogon](#) (in the **PerUserGuid** field). If the REPLGUIDs match (or [RopGetPerUserGuid](#) doesn't find the REPLGUID), then the public folder is in sync. If the REPLGUIDs don't match, the client MUST synch the read/unread data from the private store up to the public store. This is done in the reverse manner as the previous sync: the data is retrieved from the private store by using [RopReadPerUserInformation](#) and sent to the public store using [RopWritePerUserInformation](#).

When syncing using [RopReadPerUserInformation](#) and [RopWritePerUserInformation](#), it is important to note that the size of the return data potentially exceeds the maximum amount of data that can be communicated in a single ROP. For this reason, the operation is designed to stream the data to the client by having the client invoke these ROPs multiple times. Because the server will cache interim data across client calls, the client MUST complete the entire streaming operation for the data of one folder before commencing streaming operations for another folder on the same server logon.

#### **3.1.4.4 Registering for Notifications**

The client can register for notifications for a store. For a full description of notifications, see [\[MS-OXCNOTIF\]](#).

#### **3.1.5 Message Processing Events and Sequencing Rules**

The client MUST log on a store using [RopLogon](#) before attempting to do any additional ROPs on the store.

#### **3.1.6 Timer Events**

None

#### **3.1.7 Other Local Events**

None

### **3.2 Server Details**

#### **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 described in this document.

The server will maintain several tables of data in order to satisfy the various ROPs that a client can invoke. These tables include: a REPLID and REPLGUID to-and-from mapping table, a mailbox table, a per-user data table, and a receive folder table.

The REPLID and REPLGUID to-and-from mapping table contains rows of 16-bit REPLID values coupled with 128-bit REPLGUID values. When a client invokes [RopIdFromLongTermId](#), this table is searched for the REPLGUID portion of the ID passed by the client. If a row with that REPLGUID is found, the associated REPLID<14> is used to formulate the returned ShortTermID. If it is not found, a new row is added with the REPLGUID and a newly assigned REPLID, and that new REPLID is used to formulate the returned ShortTermID. The newly assigned REPLID MUST be unique within the

table. When a client invokes [RopLongTermIdFromId](#), this table is searched for the REPLID portion of the ID passed by the client. If a row is found, the associated REPLGUID is used to formulate the return LongTermID. If a row is not found, the ROP fails. Zero is an invalid REPLID value, and the server MUST NOT assign a REPLID with a value of zero.

The mailbox table is used for logging on to a private mailbox. The table contains one row for each mailbox in the database. It contains columns to specify the root and other **special folders** of the mailbox, the access permissions to the mailbox, an identifying GUID that matches the owner of the mailbox, along with other metadata, such as last logon time, various item counts and aggregate sizes within the mailbox, and so on. When a client invokes [RopLogon](#), the client passes an identifying moniker for the mailbox. The server then looks up the moniker in a global directory. The entry in the global directory will indicate the proper server to log on to for this user's mailbox, and will contain other relevant data used to find the mailbox on that server, in the mailbox table. The proper row in the mailbox table is then found, and then the user's access is checked. If the logon is allowed, various special FIDs are obtained from the table and returned to the client.

The per-user data table contains the read/unread information for various public folders on a specific public folder replica server. The table maintains the mailbox GUID, the REPLGUID, the folder LongTermID, and change number set of read items. The [RopGetPerUserLongTermIds](#), [RopGetPerUserGuid](#), [RopReadPerUserInformation](#), [RopWritePerUserInformation](#) ROPs each make use of the data in this table.

The Receive folder table contains rows of messages class strings and associated FIDs. [<15>](#)The delivery process will use the message class string on the incoming e-mail to look up the appropriate folder to which to deliver that message. The [RopGetReceiveFolder](#) and [RopSetReceiveFolder](#) ROPs each make use of the data in this table.

### 3.2.2 Timers

None.

### 3.2.3 Initialization

When a database is created, the database MUST be assigned a new randomly-generated REPLGUID.

When the REPLID and REPLGUID to-and-from mapping table is created, a single new entry MUST be added, consisting of the database REPLGUID and a newly assigned REPLID. When a database is restored from backup, the server MUST assign a new randomly-generated REPLGUID to the database, and add this new REPLGUID, along with a new REPLID, to the REPLID and REPLGUID to-and-from mapping table.

When a mailbox is created, the following entries MUST be added to the receive folder table for the new mailbox:

- "" (empty string) – Inbox in the new mailbox
- "IPM" – Inbox in the new mailbox
- "Report.IPM" – Inbox in the new mailbox
- "IPC" – Root folder of the new mailbox

### 3.2.4 Higher-Layer Triggered Events

None.

## 3.2.5 Message Processing Events and Sequencing Rules

Except for [RopLogon](#), all ROPs listed in the following sections have the client prerequisite of successfully completing a [RopLogon](#) operation. [RopLogon](#) requires that the client has successfully connected to the server by initiating a normal RPC session (calls to the **EcDoConnectEx** method). For more details about **EcDoConnectEx**, see [\[MS-OXCRPC\]](#) section 3.1.4.11.

### 3.2.5.1 Processing RopLogon

If **OpenFlags** does not have the PUBLIC bit set, this logon is going to a private mailbox. Otherwise, this logon is going to the public folders.

#### 3.2.5.1.1 Private Mailbox Logon

Look up the ESSDN (specified in the **Essdn** field of the request) in the global directory to get that user's configuration information. If lookup fails specifically because the ESSDN could not be found, the server MUST fail the operation with a **ReturnValue** of 0x000003EB. If lookup fails for any other reason, the server MUST fail the operation with a **ReturnValue** of 0x80040111.

If the user has no configured mailbox database, the ROP MUST fail with a **ReturnValue** of 0x000003EB.

If the database indicated by the user's configured mailbox database is currently offline, the operation MUST fail with a **ReturnValue** of 0x80040111.

If the user's configured mailbox is not hosted by this server, the server MUST determine the name of the correct server hosting the user's mailbox and fail the ROP with a **ReturnValue** of 0x00000478. For details about properly forming the response when a **ReturnValue** of 0x00000478 is sent, see section [2.2.1.1.2](#).

If the user who has connected to the server is the same as the owner of the mailbox being logged on to, add the TAKE\_OWNERSHIP flag to the **OpenFlags** field. This will be used in further processing.

For the USE\_PER\_MDB\_REPLID\_MAPPING bit of the **OpenFlags** field, servers SHOULD NOT implement the behavior that is described in section [2.2.1.1.2.1](#) and SHOULD implement the following behavior: [<16>](#) [<17>](#)

- If the logon is the first on the RPC session, or if the logon is additional on the RPC session and it is to the same mailbox that is associated with the first logon, then the server ignores the USE\_PER\_MDB\_REPLID\_MAPPING bit of the **OpenFlags** field.
- If the logon is additional on the RPC session, and it is to a mailbox that is different from the mailbox that is associated with the first logon, then the server inspects the USE\_PER\_MDB\_REPLID\_MAPPING bit of the **OpenFlags** field to see if it is set. If the USE\_PER\_MDB\_REPLID\_MAPPING bit is not set, then the server MUST fail the ROP with a **ReturnValue** of 0x00000478. For more details about properly forming the response when a **ReturnValue** of 0x00000478 is sent, see section [2.2.1.1.2](#). If the USE\_PER\_MDB\_REPLID\_MAPPING bit is set, then the server takes no special action.
- For each mailbox, the server maintains a replica ID (REPLID)-to-replica GUID (REPLGUID) mapping and named property-to-property ID mapping regardless of the value of the USE\_PER\_MDB\_REPLID\_MAPPING bit.

The server MUST verify the user logging on to the mailbox has permissions to it. First, the server MUST check if the user logging on has owner permission. If not, and the user logging on matches

the owner of the mailbox, the ROP MUST fail with a **ReturnValue** of 0x80070005. Next, the server MUST check if the user logging on has send-as permission. If the **OpenFlags** field has the TAKE\_OWNERSHIP flag set and the user has owner permissions on the mailbox, the user will be granted full owner permission, send-as permission, and administrative "view" permission on the mailbox. Otherwise, the connecting user is considered a delegate. Delegate logons get send-as rights only if the permissions on the mailbox expressly grant the user those rights. Delegate logons do not get full ownership rights. Full owners of the mailbox are not required to have security settings checked before performing any non-administrative operations on the mailbox.

The server MUST then find the mailbox in the mailbox table. If the mailbox is not present in the table and the user has full owner rights on the mailbox, the server MUST create the mailbox. That process includes creating the default folders and establishing the proper Receive folder values. For details about setting Receive folder values, see section [3.2.3](#). The server MUST NOT create the mailbox if the user hasn't got full owner permission. In that case, the ROP MUST fail with a **ReturnValue** of 0x000003F2. Other failures to find the user in the mailbox table (beyond a "not found" error) MUST fail the operation with a **ReturnValue** of 0x80040111.

The server MUST then determine the appropriate FIDs to return to the user. For details, see section [2.2.1.1.3.2](#).

The server is now ready to accept further ROP commands from the client on behalf of this logon session. The server can now update auditing information about the logon. Auditing information can include last logon time, user identity, and so forth, as determined by the implementer.

### 3.2.5.1.2 Public Folders Logon

The server MUST confirm the user logging on to the public folders has a mailbox in the organization. The server MUST perform the following:

1. Determine the mailbox database hosting the connected user's mailbox. The user is determined from the underlying RPC connection. The **Essdn** field specifies a mailbox to log on to for private mailbox logons only. This field will be empty for public folder logons.
2. Determine from the global directory (for that mailbox database) the preferred public folder database to use.
3. Determine the server that database is hosted upon.

If the **OpenFlags** field has the ALTERNATE\_SERVER bit set, the server MUST search for another public folder database server in the organization. The process by which another public folder database is chosen is up to the implementation, however, the server SHOULD choose a public folder database server that is not the configured preferred server. If a suitable server cannot be found, the operation MUST fail with a **ReturnValue** of 0x80040111. Otherwise, the operation MUST fail with a **ReturnValue** of 0x00000478. For details about properly forming the return values when a **ReturnValue** of 0x00000478 is sent, see section [2.2.1.1.2](#).

If the server being queried does not host the preferred public folder database, and the **LogonFlags** field does not have the ghosted bit set, and the **OpenFlags** field does not have the IGNORE\_HOME\_MDB bit set, the operation MUST fail with a **ReturnValue** of 0x00000478. For details about properly forming the return values when a **ReturnValue** of 0x00000478 is sent, see section [2.2.1.1.2](#).

If this server doesn't host a public folder database at all, or the database is not presently accessible, the operation MUST fail with a **ReturnValue** of 0x80040111.



The server MUST then determine the appropriate FIDs to return to the user. For more details, see section [2.2.1.1.4.2](#).

The server is now ready to accept further ROP commands from the client on behalf of this logon session.

### 3.2.5.2 Processing RopGetReceiveFolder

The server MUST verify that the operation is being performed against a private mailbox logon, and not a public folders logon. If the operation is performed against a public folders logon, the server MUST fail the operation with a **ReturnValue** of 0x80040102.

The server MUST validate the value of the **MessageClass** field, as specified in [2.2.1.2.1.1](#). If the value does not conform to the requirements, then the server MUST fail the operation with a **ReturnValue** of 0x80070057.

The server then MUST scan the receive folder table to find the entry that has the longest case-insensitive prefix string match to **MessageClass**.[<18>](#) The search MUST be scoped to the relevant mailbox. The server then MUST retrieve the actual message class string from the Receive folder table, and the associated FID. The Receive folder table is primed for the mailbox at creation time, as specified in section [3.2.3](#).

If no entry in the table can be matched, the server MUST fail the operation with a **ReturnValue** of 0x00000463.

Upon success, the server returns the actual configured message class string and the FID. The server can case-fold the string to all upper case, all lower case, or leave the string as stored.

### 3.2.5.3 Processing RopSetReceiveFolder

The server MUST verify the operation is being performed against a private mailbox logon, and not a public folders logon. If the operation is performed against a public folders logon, the server MUST fail the operation with a **ReturnValue** of 0x80040102.

The server MUST validate the **MessageClass** field as specified in [2.2.1.3.1.2](#). If the **MessageClass** field does not conform to the requirements, the server MUST fail the operation with a **ReturnValue** of 0x80070057.

If the **MessageClass** field is a case-insensitive match to either "IPM" or "Report.IPM", the server MUST fail the operation with a **ReturnValue** of 0x80070005.

If the **MessageClass** field is a zero-length string, or **FolderId** field is all zeros, the server MUST fail the operation with a **ReturnValue** of 0x80004005.

The server MUST search the Receive folder table using a case-insensitive string comparison for an exact match to the **MessageClass** field. The search MUST be scoped to the relevant mailbox. If a match is found, the value of the **FolderId** field MUST replace the FID stored in the table on that row. (The details about the content of a table row are provided following this paragraph.) If a match is not found, a new row is added with the **MessageClass** and **FolderId** field values. The server can case-fold the value of the **MessageClass** field to upper case or lower case, or leave the value unchanged before storage. After modifying or inserting the new row, the "Last-modification Time" column for that row MUST be set to the current system-time of the server, adjusted to Coordinated Universal Time (UTC).

Each row of the table contains at least the following three columns, with each column corresponding to a property. Any other columns included in the table, and the storage format for the table, are determined by the implementer.

1. "Folder ID" column ([PidTagFolderId](#) property) — Contains the folder ID (FID) of the Receive folder, which is the folder to which messages of the specified message class will be delivered. The Receive folder MUST be a folder that is within the user's mailbox.
2. "Message Class" column ([PidTagMessageClass](#) property) — Contains a string that specifies the message class that is configured for the Receive folder.
3. "Last-modification Time" column ([PidTagLastModificationTime](#) property) — Contains the current system-time, in UTC, when the entry was created or last modified.

#### 3.2.5.4 Processing RopGetReceiveFolderTable

The server MUST verify the operation is being performed against a private mailbox logon, and not a public folders logon. If the operation is performed against a public folders logon, the server MUST fail the operation with a **ReturnValue** of 0x80040102.

The server MUST return all rows of the Receive folder table for the relevant mailbox. The **Rows** field of the [RopGetReceiveFolderTable](#) response contains either a **StandardPropertyRow** structure or a **FlaggedPropertyRow** structure for each row of the Receive folder table. If there is an error retrieving any data of a row from the Receive folder table, the server returns the row formatted as a **FlaggedPropertyRow** structure; otherwise, the server returns the row formatted as a **StandardPropertyRow** structure. For more details about these structures, see [\[MS-OXCDATA\]](#) section 2.9.1 and its sub-sections.

The server can convert message class values to all upper case or all lower case or return the value as stored.

#### 3.2.5.5 Processing RopGetStoreState

Servers SHOULD NOT implement this ROP and SHOULD return a NotSupported error. Servers MAY implement this ROP as specified in this section. [<19>](#)

If the server implements this ROP, it has the following behavior:

- The server MUST verify the operation is being performed against a private mailbox logon, and not a public folders logon. If the operation is performed against a public folders logon, the server MUST fail the operation with a **ReturnValue** of 0x80040102.
- If the mailbox has any persisted search folders, the server MUST set the STORE\_HAS\_SEARCHES flag in the return value. If the mailbox does not have any persisted search folders, the server MUST NOT set the STORE\_HAS\_SEARCHES flag in the return value.
- The server MUST NOT set any other flags in the return value.

#### 3.2.5.6 Processing RopGetOwningServers

If the operation is performed against a private mailbox store, the server can fail the operation, or it can compute a correct answer for the client.

Each public folder has associated configuration information, including which servers are configured to actually hold content of the folder. This specific configuration indicates one of several potential states for each replica server. An "Active" replica contains content and is expected to serve that

content to clients. Servers in other replica states do not serve content to clients. These other replica states are implementation-specific, but possible definitions are as follows:

- An "Inactive" replica contains content, but is not going to serve that content to clients.
- A "Deleted" replica once contained content and presently does not.

Each server in the organization's network has a tangible communication cost due to the following implementation-dependent factors: network hardware costs, the cost of the network connectors (various WAN versus LAN costs and so forth), and the perceived cost of using the network for certain applications, and so on.

The server **MUST** retrieve the current replica information for the specific public folder specified by the **FolderId** field. This replica information is a list, each entry consisting of at least a server identifier and the replication state for this folder on that server (Active/Inactive/Deleted/etc). The server **MUST** obtain the network cost for each server in the list, if that cost information isn't already in the list. The network cost values are expressed relative to the server servicing the request, not the client making the request.

The server **MUST** remove entries from the list that are not active replicas. The server can remove entries from the list which, at its discretion, it determines to be too expensive for the client to reach. The server can remove entries if another configuration indicates that the client be prohibited from attempting a connection, if such a configuration exists. If the resulting trimmed list is empty, the operation **MUST** fail with a **ReturnValue** of 0x00000469.

The server **MUST** sort the list according to the cost information, least expensive items sorting to the front of the list. Servers with the same cost can appear in any order, but the server **SHOULD** ensure that the same list values sort to the same order every time.

The server **MUST** count the number of entries at the front of the list that all have the same cost value. The resultant value will be the number of cheapest, equally costed servers (the **CheapServersCount** value returned in the response).

The current total list length constitutes the **OwningServersCount** value returned in the response. The list contents of server identifiers constitute the value in the **OwningServers** field. The server **MUST** map whatever identifier moniker for each server it has into an ESSDN string to return to the client.

### 3.2.5.7 Processing RopPublicFolderIsGhosed

If the operation is performed against a private mailbox store, the server can fail the operation, or it can compute a correct answer for the client.

Each public folder has associated configuration information, including which servers are configured to actually hold content of the folder. This specific configuration indicates one of several potential states for each replica server. An "Active" replica contains content and is expected to serve that content to clients. Servers in other replica states do not serve content to clients. These other replica states are implementation-specific, but possible definitions are as follows:

- An "Inactive" replica contains content, but is not going to serve that content to clients.
- A "Deleted" replica once contained content and presently does not.

Each server in the organization's network has a tangible communication cost due to the following implementation-dependent factors: network hardware costs, the cost of the network connectors (various WAN versus LAN costs, and so forth), and the perceived cost of using the network for certain applications, and so on.

The server MUST retrieve the current replica information for the specific public folder specified by the **FolderId** field. This replica information is a list, each entry consisting of at least a server identifier and the replication state (Active, Inactive, Deleted) for this folder on that server. The server MUST obtain the network cost for each server in the list, if that cost information isn't already in the list. The network cost values are expressed relative to the server, not the client making the request.

The server MUST return an **IsGhosed** value of TRUE if the queried server is not listed as an active replica of the folder. The **IsGhosed** value MUST be FALSE if the queried server is listed as an active replica of the folder.

The server MUST remove entries from the list which are not active replicas. The server can remove entries from the list which, at its discretion, it determines to be too expensive for the client to reach. The server can remove entries if another configuration indicates that the client be prohibited from attempting a connection, if such configuration exists. If the resulting trimmed list is empty, the operation MUST be failed with a **ReturnValue** of 0x00000469. A client MUST interpret this **ReturnValue** value as implying an **IsGhosed** value of TRUE.

The server MUST sort the list according to the cost information, least expensive items sorting to the front of the list. Servers with the same cost can appear in any order, but the server SHOULD ensure that the same list values sort to the same order every time.

The server MUST count the number of entries at the front of the list that all have the same cost value. The resultant value will be the number of cheapest, equally costed servers (the **CheapServersCount** return value).

The current total list length constitutes the **ServersCount** return value. The list contents of server identifiers constitute the value of the **Servers** field. The server MUST map whatever identifier moniker for each server it has into an ESSDN string to return to the client.

### 3.2.5.8 Processing RopLongTermIdFromId

The server MUST verify whether the REPLID portion of the given ID exists in the REPLID and REPLGUID to-and-from mapping table. The server MUST NOT otherwise attempt to confirm that the given ID is a change number for an existing or former object, is an identifier for an existing or former object, or has ever been assigned for any reason.

If the given REPLID is not in the REPLID and REPLGUID to-and-from mapping table, the operation MUST fail with a **ReturnValue** of 0x8004010F.

The server MUST construct the returned LongTermID by composing the REPLGUID recovered from the REPLID and REPLGUID to-and-from mapping table for the REPLID portion of the given ID, copying the global counter portion of the given ID to the global counter portion of the LongTermID, and by setting the padding bytes of the LongTermID to zero.

### 3.2.5.9 Processing RopIdFromLongTermId

The server MUST NOT perform any specific validation of the passed LongTermID value. The server MUST search the REPLID and REPLGUID to-and-from mapping table for the REPLGUID portion of the LongTermID field. If a row is not found, the server MUST add a new row consisting of the REPLGUID portion of the LongTermID field and a newly assigned REPLID value. The new REPLID value MUST be unique in the REPLID and REPLGUID to-and-from mapping table.

The server MUST construct the returned ID by composing the REPLID recovered from the REPLID and REPLGUID to-and-from mapping table for the REPLGUID portion of the LongTermID, and by copying the global counter portion of the LongTermID to the returned ID.

The server MUST ignore the content of the padding bytes in the LongTermID field.

### 3.2.5.10 Processing RopGetPerUserLongTermIds

The server MUST verify the operation is being performed against a private mailbox logon, and not a public folders logon. If the operation is performed against a public folders logon, the server MUST fail the operation with a **ReturnValue** of 0x80040102.

The server MUST search the per-user data table for the mailbox for entries identified by the **DatabaseGuid** field in the request. For each entry in the table, the server MUST collect the associated public folder long-term ID. The total number of long-term IDs collected is specified in the **LongTermIdCount** field and the aggregated list of long-term IDs constitutes the value of the **LongTermIds** field.

The server can return the list of long-term IDs in any order. The server can return an empty list.

### 3.2.5.11 Processing RopGetPerUserGuid

The server MUST verify the operation is being performed against a private mailbox logon, and not a public folders logon. If the operation is performed against a public folders logon, the server MUST fail the operation with a **ReturnValue** of 0x80040102.

The server MUST search the per-user data table for the mailbox for the only row with an FID equal to the one specified by the **LongTermId** field in the request. The server MUST return the associated REPLGUID value in the **DatabaseGuid** field.

### 3.2.5.12 Processing RopReadPerUserInformation

This operation can be issued against either a private mailbox logon or a public folders logon.

#### 3.2.5.12.1 Private Mailbox Specific Behavior

The server MUST search the per-user data table for the mailbox for the only row with an FID equal to the value of the **FolderId** field. The server MUST retrieve from that row the stored change number set of read items. After the change number set is retrieved, the server's behavior, as specified in section [3.2.5.12.3](#), is the same for both private mailboxes and public folders.

#### 3.2.5.12.2 Public Folders Specific Behavior

The server MUST first determine that the persisted read/unread information for the user is up to date. If the server is maintaining any in-memory caches of the per-user read/unread information, the data for the current user MUST now be flushed to disk.

The server MUST search the per-user data table for the only row with an FID equal to the value of the **FolderId** field and the user ID equal to the logged on user. The server MUST retrieve from that row the stored change number set of read items. After the change number set is retrieved, the server's behavior, as specified in section [3.2.5.12.3](#), is the same for both private mailboxes and public folders.

#### 3.2.5.12.3 Common Behavior

The change number set MUST be serialized into a binary large object (BLOB) that is formatted as a serialized IDSET with REPLGUID structure, as specified in [\[MS-OXCFXICS\]](#) section 2.2.2.3.2. The server then returns the BLOB in the **Data** field of the response.

The size of the BLOB can potentially exceed the maximum amount of data that can be communicated in a single [RopReadPerUserInformation](#) response. For this reason, the [RopReadPerUserInformation](#) ROP is designed to stream the data to the client by having the client invoke the ROP multiple times. In other words, the client can send multiple [RopReadPerUserInformation](#) requests to retrieve the BLOB in segments.

On each invocation of [RopReadPerUserInformation](#), the server MUST inspect the value of the **MaxDataSize** field of the [RopReadPerUserInformation](#) request because the value can be different in each request. In certain cases, the server MUST adjust the value of **MaxDataSize**. For more details about inspecting and adjusting the value, see the summary in this section. After the server has inspected and, if necessary, adjusted the value of **MaxDataSize**, the server MUST compare the value to the size of the remaining BLOB segment. If the adjusted **MaxDataSize** value is less than the size of the remaining BLOB segment, then the server MUST set **HasFinished** to FALSE to indicate to the client that some data remains to be retrieved.

The **DataOffset** field in the request contains an index into the BLOB. In other words, the value of **DataOffset** specifies the position within the BLOB of the first byte of data to be returned to the client. The value of **DataOffset** is always zero in the first [RopReadPerUserInformation](#) request. The client updates **DataOffset** based on the number of bytes received in the previous response so that **DataOffset** always points to the first byte of the next BLOB segment to be returned.

Summary:

1. The **MaxDataSize** field of the request specifies the maximum number of bytes that can be returned in a single [RopReadPerUserInformation](#) response. The server MUST adjust the **MaxDataSize** value in certain cases, as specified in item 2 of this summary.
2. When the client retrieves a BLOB in segments, the client can set **MaxDataSize** to a different value in each [RopReadPerUserInformation](#) request that is used to retrieve the BLOB. Therefore, the server MUST examine the value of **MaxDataSize** on each invocation of [RopReadPerUserInformation](#) as follows.
  1. The server MUST compare the value of **MaxDataSize** to zero. If **MaxDataSize** equals 0, then the server MUST adjust the value of **MaxDataSize** to a suitable default value, which is determined by the implementation. [<20>](#)
  2. The server SHOULD compare the value of **MaxDataSize** to some suitable maximum value, as determined by the implementation. If **MaxDataSize** > [server's suitable maximum], then the server SHOULD adjust the value of **MaxDataSize** to the suitable maximum value. [<21>](#)
  3. The server MUST compare the adjusted value of **MaxDataSize** to the size of the remaining BLOB segment. If [size of remaining BLOB segment] > [adjusted **MaxDataSize**], then the server MUST set **HasFinished** to FALSE to indicate to the client that additional requests are necessary to retrieve all of the remaining portions of the BLOB. The size of the remaining BLOB segment is equal to the size of the entire BLOB minus the value of **DataOffset**.
3. The **DataSize** field specifies the actual number of bytes that are returned in the response. The value of **DataSize** MUST NOT exceed the adjusted value of the **MaxDataSize** field. For details about adjusting **MaxDataSize**, see item number 2 of this summary. The server MUST set **DataSize** to the lesser of the following two values:
  1. The adjusted value of **MaxDataSize**.
  2. The size of the remaining BLOB segment. This is the size of the portion of the BLOB that remains to be sent to the client and is equal to the size of the entire BLOB minus the value of **DataOffset**.

4. The server **MUST** set **HasFinished** to TRUE if **DataOffset** plus **DataSize** equals the size of the entire BLOB. In other words, when the server sends the last segment of the BLOB, **HasFinished** **MUST** be set to TRUE.

### 3.2.5.13 Processing RopWritePerUserInformation

This operation can be issued against either a private mailbox logon or a public folders logon.

#### 3.2.5.13.1 Common Behavior

Each invocation of this ROP accumulates data from the client until the client makes a final call with **HasFinished** set to TRUE. The server **MUST** aggregate the data across multiple invocations and it **MUST** validate the entire data set before persisting to permanent storage.

The server **MUST** determine if the current invocation is a continuation of a previous invocation by examining the **FolderId** and **DataOffset** fields. If the FID has changed since the last invocation, or the **DataOffset** value does not equal the amount of data already written, the server **MUST** assume the previous operation was aborted and **MUST** dispose of any accumulated data. In addition, if the current invocation's **DataOffset** isn't zero, the ROP **MUST** fail with a **ReturnValue** of 0x80004005.

Once the client invokes this ROP with **HasFinished** set to TRUE, the server **MUST** validate the accumulated data and verify it is a properly formed serialized IDSET with REPLGUID as specified in [\[MS-OXCFCICS\]](#) section 2.2.2.3.2. If the data is not properly formed, the ROP **MUST** fail with a **ReturnValue** of 0x000004ED.

After performing the specific behavior in the following sections, the server **MUST** record, in UTC, the current system time on the appropriate row in the table.

#### 3.2.5.13.2 Private Mailbox Specific Behavior

The server **MUST** search the per-user data table of the mailbox for the only row with an FID equal to the value of the **FolderId** field. If the row exists, the REPLGUID field and accumulated change number information **MUST** replace any existing values in the table. If the row does not presently exist, a new row for the mailbox and folder **MUST** be added, setting the REPLGUID field and accumulated change number information onto that row.

#### 3.2.5.13.3 Public Folders Specific Behavior

The server **MUST** search the per-user data table for the only row with a user ID equal to user ID associated with the session logon and an FID equal to the value of the **FolderId** field. If the row exists, the accumulated change number information **MUST** replace any existing values in the table. If the row does not exist, a new row for the user and folder **MUST** be added, setting the accumulated change number information onto that row.

### 3.2.6 Timer Events

None.

### 3.2.7 Other Local Events

None.

## 4 Protocol Examples

This section provides a sample sequence of ROP requests and responses that a client and a server might exchange as the client logs on to a user mailbox or public folders, reads or writes mailbox-level properties, or determines the availability of content for public folders. Note that the examples listed here only show the relevant portions of the specified ROPs; these portions are not the final byte sequences that get transmitted over the wire. Also note that the data for a multi-byte field appear in **little-endian** format, with the bytes in the field presented from least significant to most significant. Generally speaking, these ROP requests are packed with other ROP requests, compressed and packed in one or more RPC calls as specified in [\[MS-OXCRPC\]](#) section 3.1.7.4. These examples assume the client has already successfully connected to the server. For more details, see [\[MS-OXCRPC\]](#) section 4.1.

The byte sequences are shown in the following format with each byte's value expressed as a two-digit hexadecimal number.

```
0080: 45 4d 53 4d 44 42 2e 44-4c 4c 00 00 00 00 00 00
```

The value, 0080, at the far left is the byte sequence's offset from the beginning of the buffer. Following the offset is a colon and then a series of up to 16 byte values. Here, the first byte value (45) in the series is located 0x80 bytes (128 bytes) from the beginning of the buffer. The seventh byte value (2e) in the series is located 0x86 bytes (134 bytes) from the beginning of the buffer. The dash between the eighth byte (44) and ninth byte (4c) has no semantic value, and serves only to distinguish the eight byte boundary for readability purposes.

Each set of byte sequences is followed by one or more lines interpreting it.

The following example shows how a **property tag** and its property value are represented in a buffer and interpreted directly from it (according to the PropertyValue structure format specified in [\[MS-OXCDATA\]](#) section 2.13.2). The data appears in the buffer in little-endian format.

```
0020: 03 00 76 66 0a 00 00-00
```

[0020-0023] Property tag: 0x66760003 ([PidTagRuleSequence](#))

[0024-0027] Property value: 10

Generally speaking, interpreted values will be shown in their native format, interpreted appropriately from the raw byte sequence as specified in the appropriate section. Here, the byte sequence "0a 00 00 00" has been interpreted as a **PtypInteger32** with a value of 10 because the type of the [PidTagRuleSequence](#) property is **PtypInteger32**. Property data types are specified in [\[MS-OXCDATA\]](#) section 2.12.1.

### 4.1 RopLogon for a Private Mailbox

[RopLogon](#) request for a private mailbox:

```
0000: 01 0c 04 00 01 00 00 00-00 68 00 2f 6f 3d 46 69
0010: 72 73 74 20 4f 72 67 61-6e 69 7a 61 74 69 6f 6e
0020: 2f 6f 75 3d 45 78 63 68-61 6e 67 65 20 41 64 6d
0030: 69 6e 69 73 74 72 61 74-69 76 65 20 47 72 6f 75
0040: 70 20 28 46 59 44 49 42-4f 48 46 32 33 53 50 44
0050: 4c 54 29 2f 63 6e 3d 52-65 63 69 70 69 65 6e 74
```



0060: 73 2F 63 6E 3D 41 64 6D-69 6E 69 73 74 72 61 74  
0070: 6F 72 00

[0000-0000] **LogonFlags** — Private

[0001-0004] **OpenFlags** — HOME\_LOGON, TAKE\_OWNERSHIP, NO\_MAIL, USE\_PER\_MDB\_REPLID\_MAPPING

[0005-0008] **StoreState** — Value is ignored by the server.

[0009-000A] **EssdnSize** — The size of the ESSDN field is 0x68 bytes long.

[000B-0072] **ESSDN**

[RopLogon](#) success response for a private mailbox:

0000: 01 01 00 00 00 00 78 27-1A 01 00 00 00 00 78 27  
0010: 1C 01 00 00 00 00 78 27-1D 01 00 00 00 00 78 27  
0020: 1B 01 00 00 00 00 78 27-1E 01 00 00 00 00 78 27  
0030: 1F 01 00 00 00 00 78 27-20 01 00 00 00 00 78 27  
0040: 21 01 00 00 00 00 78 27-24 01 00 00 00 00 78 27  
0050: 25 01 00 00 00 00 78 27-22 01 00 00 00 00 78 27  
0060: 23 01 00 00 00 00 78 27-26 07 F7 F8 91 A5 1C 34  
0070: 16 41 8C 48 9D B0 1A 86-F5 0B 01 00 4D 77 D4 64  
0080: 83 49 70 4F 9B 8B 46 E6-35 BB 78 AB 0D 10 0F 01  
0090: 0A 03 D8 07 60 53 1A C2-BE 82 C8 01 00 00 00 01

[0000-0000] **LogonFlags** — Private

[0001-0068] **FolderIds** — As follows:

[0001-0008] Mailbox Root Folder FID

[0009-0010] Deferred Action Folder FID

[0011-0068] <more FIDs>

[0069-0069] **ResponseFlags** — SendAsRight, OwnerRight, Reserved

[006A-0079] **MailboxGuid**

[007A-007B] **ReplId**

[007C-008B] **ReplGuid**

[008C-0093] **LogonTime** — 2008/03/10 Mon 15:10:13

[0094-009B] **GwartTime** — 2008/03/10 14:55:19

[009C-009F] **StoreState** — STORE\_HAS\_SEARCHES

## 4.2 RopLogon for Public Folders

[RopLogon](#) request for public folders:

0000: 00 06 04 00 01 00 00 00-00 00 00

[0000-0000] **LogonFlags** — Log on to public folders

[0001-0004] **OpenFlags** — PUBLIC, HOME\_LOGON, NO\_MAIL, USE\_PER\_MDB\_REPLID\_MAPPING

[0005-0008] **StoreState** — Value is ignored.

[0009-000A] **EssdnSize** — No ESSDN is given for public logons.

[RopLogon](#) success response for public folders:

```
0000: 00 01 00 00 00 00 00 00-06 01 00 00 00 00 00 00
0010: 01 01 00 00 00 00 00 00-02 01 00 00 00 00 00 00
0020: 03 01 00 00 00 00 00 00-04 01 00 00 00 00 00 00
0030: 05 00 00 00 00 00 00 00-00 03 00 00 00 00 00 00
0040: 07 03 00 00 00 00 00 00-08 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00-00 01 00 70 5B CA BF 1E
0070: F9 98 41 89 7D 47 9E 09-45 FD 2F 95 DA FE 74 E0
0080: F7 4C 4C 81 EF 83 BA 85-0B E8 E4
```

[0000-0000] **LogonFlags** — Log on to public folders

[0001-0050] **FolderIds** — As follows:

[0001-0008] Public Folders FID

[0009-0010] IPM Subtree FID

[0011-0050] <other FIDs>

[0051-0068] Unused — Set to zero.

[0069-006A] **ReplId**

[006B-007A] **ReplGuid**

[007B-008A] **PerUserGuid**

### 4.3 RopGetReceiveFolder

[RopGetReceiveFolder](#) request:

0000: 00

[0000-0000] **MessageClass** <empty string>

[RopGetReceiveFolder](#) response:

0000: 01 00 00 00 00 00 78 27 1E-00

[0000-0007] **FolderId**

[0008-0008] **ExplicitMessageClass** <empty string>

#### 4.4 RopSetReceiveFolder

[RopSetReceiveFolder](#) request:

```
0000: 01 00 00 00 00 78 27 1A-49 50 4D 2E 53 6F 6D 65
0010: 4D 65 73 73 61 67 65 43-6C 61 73 73 00
```

[0000-0007] **FolderId**

[0008-001C] **MessageClass**

[RopSetReceiveFolder](#) response:

No response.

#### 4.5 RopGetReceiveFolderTable

[RopGetReceiveFolderTable](#) request:

No fields in the request.

[RopGetReceiveFolderTable](#) response:

```
0000: 04 00 00 00 00 01 00 00-00 00 78 27 1E 00 5E FF
0010: 54 5F C0 82 C8 01 00 01-00 00 00 00 78 27 1A 49
0020: 50 43 00 32 EF 56 5F C0-82 C8 01 00 01 00 00 00
0030: 00 78 27 1E 49 50 4D 00-32 EF 56 5F C0 82 C8 01
0040: 00 01 00 00 00 00 78 27-1E 52 45 50 4F 52 54 2E
0050: 49 50 4D 00 32 EF 56 5F-C0 82 C8 01
```

[0000-0003] **RowCount** (4 rows being returned)

[0004-0004], [0016-0016], [002B-002B], [0040-0040] Error Flag Indicator (no error). For more details, see [\[MS-OXCDATA\]](#) section 2.10.

[0005-000C], [0017-001E], [002C-0033], [0041-0048] **FolderId**

[000D-000D], [001F-0022], [0034-0037], [0049-0053] **MessageClass**

[000E-0015], [0023-002A], [0038-003F], [0054-005B] **LastModification**

#### 4.6 RopIdFromLongTermId

[RopIdFromLongTermId](#) request:

```
0000: 70 5B CA BF 1E F9 98 41-89 7D 47 9E 09 45 FD 2F
0010: 00 00 00 00 00 12 00 00
```

[0000-000F] **LongTermID REPLGUID**

[0010-0015] **LongTermID** counter

[0016-0017] **LongTermID** padding

[RopIdFromLongTermId](#) response:

```
0000: 05 00 00 00 00 00 00 12
```

[0000-0001] **ObjectId REPLID**

[0002-0007] **ObjectId** counter

#### 4.7 RopGetPerUserLongTermIds

[RopGetPerUserLongTermIds](#) request:

```
0000: 4D 77 D4 64 83 49 70 4F-9B 8B 46 E6 35 BB 78 AB
```

[0000-000F] **DatabaseGuid**

[RopGetPerUserLongTermIds](#) response:

```
0000: 00 00
```

[0000-0001] **LongTermIdCount** (no IDs being returned)

#### 4.8 RopReadPerUserInformation

[RopReadPerUserInformation](#) request:

```
0000: 70 5B CA BF 1E F9 98 41-89 7D 47 9E 09 45 FD 2F
0010: 00 00 00 00 00 12 00 00-00 00 00 00 00 00 00
```

[0000-0017] **FolderId**

[0018-0018] **Reserved**

[0019-001C] **DataOffset**

[001D-001E] **MaxDataSize**

[RopReadPerUserInformation](#) response:

```
0000: 01 18 00 D8 44 AE 73 F9-61 5D 4F B3 C6 9A 7C 31
0010: FE C1 23 06 00 00 00 78-2B 33 00
```

[0000-0000] **HasFinished**

[0001-0002] **DataSize**

[0003-001A] **Data**

## 4.9 RopWritePerUserInformation

[RopWritePerUserInformation](#) request:

```
0000: 70 5B CA BF 1E F9 98 41-89 7D 47 9E 09 45 FD 2F
0010: 00 00 00 00 00 12 00 00-01 00 00 00 00 18 00 D8
0020: 44 AE 73 F9 61 5D 4F B3-C6 9A 7C 31 FE C1 23 06
0030: 00 00 00 78 2B 33 00 D8-44 AE 73 F9 61 5D 4F B3
0040: C6 9A 7C 31 FE C1 23
```

[0000-0017] **FolderId**

[0018-0018] **HasFinished**

[0019-001C] **DataOffset**

[001D-001E] **DataSize**

[001F-0036] **Data**

[0037-0046] **ReplGuid**

[RopWritePerUserInformation](#) response:

No response.

## 5 Security

### 5.1 Security Considerations for Implementers

There are no special security considerations specific to the Store Object protocol. General security considerations pertaining to the underlying RPC-based transport apply. For details, see [\[MS-OXCROPS\]](#).

### 5.2 Index of Security Fields

None.

## 6 Appendix A: Product Behavior

The information in this specification is applicable to the following product versions. References to product versions include released service packs.

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

Exceptions, if any, are noted below. If a service pack number appears with the product version, behavior changed in that service pack. The new behavior also applies to subsequent service packs of the product unless otherwise specified.

Unless otherwise specified, any statement of optional behavior in this specification 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 product does not follow the prescription.

[<1> Section 2.2.1.1.1:](#) Exchange 2003 and Exchange 2007 do not reject the operation when the client uses an undefined flag value.

[<2> Section 2.2.1.1.2:](#) Exchange 2003 and Exchange 2007 use the HOME\_LOGON bit as follows: When the bit is set in a public folder logon, per-user read/unread information is tracked. This bit is ignored in a private mailbox logon.

[<3> Section 2.2.1.1.2:](#) Exchange 2003 and Exchange 2007 use the TAKE\_OWNERSHIP bit as follows: If set, the bit grants SendAs permission to an admin logon. If not set, SendAs permission is not granted.

[<4> Section 2.2.1.1.2.1:](#) Outlook 2003 does not use this bit and always assumes that all logons on an RPC session use the same REPLID/REPLGUID mapping and named property/property ID mapping.

[<5> Section 2.2.1.1.3.6:](#) Outlook 2003, Outlook 2007, and Outlook 2010 cache the REPLGUID returned by [RopLogon](#). If a reconnect occurs and the REPLGUID changes, Outlook 2003, Outlook 2007, and Outlook 2010 fail the logon and prompt the user to restart the application.

[<6> Section 2.2.1.1.4.4:](#) Outlook 2003, Outlook 2007, and Outlook 2010 cache the REPLGUID returned by [RopLogon](#). If a reconnect occurs and the REPLGUID changes, Outlook 2003, Outlook 2007, and Outlook 2010 fail the logon and prompt the user to restart the application.

[<7> Section 2.2.1.1.5:](#) If the user doesn't exist in the Active Directory forest, Exchange 2007 and Exchange 2010 return ecUnknownUser and Exchange 2003 returns ecLoginFailure.

[<8> Section 2.2.1.1.5:](#) Exchange 2003 and Exchange 2007 return ecServerPaused after five attempts within any 10-second period to log on to a mailbox that is not hosted on the server.

[<9> Section 2.2.1.5:](#) Exchange 2010 does not support [RopGetStoreState](#), but Exchange 2003 and Exchange 2007 do support it.

<10> [Section 2.2.1.6](#): Exchange 2003, Exchange 2007, and Exchange 2010 will successfully complete a [RopGetOwningServers](#) operation when issued against a private mailbox logon. The results are undefined.

<11> [Section 2.2.1.6.2.3](#): Exchange 2003 queries the transport engine for cost information and removes servers that have a connection cost of "infinite". Exchange 2007 and Exchange 2010 query **Active Directory** for cost information and remove servers that have a connection cost over 500. The source used to determine connection costs and the algorithm used to determine the servers that are to be removed are implementation-defined.

<12> [Section 2.2.1.7.2.4](#): Exchange 2003 queries the transport engine for cost information and removes servers that have a connection cost of "infinite". Exchange 2007 and Exchange 2010 query Active Directory for cost information and remove servers that have a connection cost over 500. The source used to determine connection costs and the algorithm used to determine the servers that are to be removed are implementation-defined.

<13> [Section 3.1.4.1](#): Autodiscover HTTP Service protocol [\[MS-OXDISCO\]](#) is not supported by Outlook 2003.

<14> [Section 3.2.1](#): Exchange 2003, Exchange 2007, and Exchange 2010 assign REPLIDs in increasing sequential order, starting with the number 1.

<15> [Section 3.2.1](#): Exchange 2003, Exchange 2007, and Exchange 2010 maintain a single Receive folder table per database. The data within the table are scoped to each mailbox by including a per-mailbox moniker on each row. Conceptually, there is a single Receive folder table per mailbox.

<16> [Section 3.2.5.1.1](#): Exchange 2003 always ignores the USE\_PER\_MDB\_REPLID\_MAPPING bit and behaves as if the bit is not set, as described in section [2.2.1.1.1.2.1](#).

<17> [Section 3.2.5.1.1](#): Exchange 2007 implements the behavior that is described in section [2.2.1.1.1.2.1](#).

<18> [Section 3.2.5.2](#): Exchange 2003, Exchange 2007, and Exchange 2010 case-fold all message class strings in the Receive folder table to upper case. All message class strings returned from [RopGetReceiveFolder](#) and [RopGetReceiveFolderTable](#) are sent to the client in upper case.

<19> [Section 3.2.5.5](#): Exchange 2003 and Exchange 2007 implement **RopGetStoreState**.

<20> [Section 3.2.5.12.3](#): Exchange (all versions) uses 4096 for the default value.

<21> [Section 3.2.5.12.3](#): Exchange (all versions) uses 4096 for the maximum value.



## 7 Change Tracking

This section identifies changes made to [MS-OXCSTOR] protocol documentation between February 2010 and May 2010 releases. Changes are classed as major, minor, or editorial.

**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.
- A protocol is deprecated.
- The removal of a document from the documentation set.
- Changes made for template compliance.

**Minor** changes do not affect protocol interoperability or implementation. Examples are updates to fix technical accuracy or ambiguity at the sentence, paragraph, or table level.

**Editorial** changes apply to grammatical, formatting, and style issues.

**No changes** means that the document is identical to its last release.

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

- New content added.
- Content update.
- 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 always have the revision type "Editorially updated."

Some important terms used in revision 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.

Changes are listed in the following table. If you need further information, please contact [protocol@microsoft.com](mailto:protocol@microsoft.com).

Section	Tracking number (if applicable) and description	Major change (Y or N)	Revision Type
<a href="#">1.1 Glossary</a>	55264 Removed the product names "Exchange" and "Exchange Server" from the definition of the term "global directory" and re-worded the definition as necessary.	N	Content update.
<a href="#">1.2.1 Normative References</a>	55268 Removed reference to [MS-DTYP].	N	Content update.
<a href="#">1.3 Overview</a>	Updated the section title.	N	Content updated for template compliance.
<a href="#">2.1 Transport</a>	54973 Changed reference from [MS-OXCROPS] to [MS-OXCRPC].	N	Content update.
<a href="#">2.2.1 Remote Operations</a>	55268 Changed "below" to "in the following sections".	N	Content update.
<a href="#">2.2.1.1.1 LogonFlags</a>	54975 Deleted references from the description of the Undercover bit.	N	Content update.
<a href="#">2.2.1.1.1.2 OpenFlags</a>	49243 Revised the description of the USE_PER_MDB_REPLID_MAPPING bit and moved the product behavior note to section 2.2.1.1.1.2.1.	N	Product behavior note removed.
<a href="#">2.2.1.1.1.2 OpenFlags</a>	54980 Deleted the OVERRIDE_HOME_MDB flag.	N	Content update.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Revision Type
<a href="#">2.2.1.1.1.2.1 USE_PER_MDB_REPLID_MAPPING Details</a>	49243 Clarified how the USE_PER_MDB_REPLID_MAPPING bit is used by the client to control server behavior.	Y	Content update.
<a href="#">2.2.1.1.1.2.1 USE_PER_MDB_REPLID_MAPPING Details</a>	49243 Added a product behavior note from section 2.2.1.1.1.2.	N	New product behavior note added.
<a href="#">2.2.1.1.1.2.1 USE_PER_MDB_REPLID_MAPPING Details</a>	49243 Removed two product behavior notes.	N	Product behavior note removed.
<a href="#">2.2.1.1.1.3 StoreState</a>	55196 Clarified the description.	N	Content update.
<a href="#">2.2.1.1.5 ReturnValue</a>	54980 Updated the description of the ecWrongServer return code.	N	Content update.
<a href="#">2.2.1.13.2 Response</a>	54988 Clarified the contents of the response.	N	Content update.
<a href="#">3.2.5 Message Processing Events and Sequencing Rules</a>	55268 Changed "below" to "in the following sections".	N	Content update.
<a href="#">3.2.5.1.1 Private Mailbox Logon</a>	49243 Updated server behavior related to the USE_PER_MDB_REPLID_MAPPING bit of the OpenFlags field.	Y	Content update.
<a href="#">3.2.5.1.1 Private Mailbox Logon</a>	49243 Added product behavior notes to specify Exchange 2003 and Exchange 2007 behavior related to the USE_PER_MDB_REPLID_MAPPING bit of the OpenFlags field.	N	New product behavior note added.
<a href="#">3.2.5.6 Processing RopGetOwningServers</a>	54986 Clarified that a server's communication cost is due to implementation-dependent factors.	N	Content update.
<a href="#">3.2.5.7 Processing RopPublicFolderIsGhosed</a>	54986 Clarified that a server's communication cost is due to implementation-dependent factors.	N	Content update.
<a href="#">3.2.5.13.1 Common Behavior</a>	55268 Changed "below" to "in the following sections".	N	Content update.

## 8 Index

### A

[Applicability](#) 10

### C

[Capability negotiation](#) 10

[Change tracking](#) 57

Client

[overview](#) 35

### E

Examples

[overview](#) 48

### F

[Fields – vendor-extensible](#) 10

### G

[Glossary](#) 7

### I

Implementer

[security considerations](#) 54

[Index of security parameters](#) 54

[Informative references](#) 9

[Introduction](#) 7

### M

Messages

[overview](#) 11

Messaging

[transport](#) 11

### N

[Normative references](#) 8

### O

[Overview](#) 9

### P

[Parameters – security index](#) 54

[Preconditions](#) 10

[Prerequisites](#) 10

[Product behavior](#) 55

### R

References

[informative](#) 9

[normative](#) 8

[Relationship to other protocols](#) 10

### S

Security

[implementer considerations](#) 54

[overview](#) 54

[parameter index](#) 54

Server

[overview](#) 37

[Standards Assignments](#) 10

### T

[Tracking changes](#) 57

[Transport](#) 11

### V

[Vendor-extensible fields](#) 10

[Versioning](#) 10