[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.mspx). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

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.
08/04/2010	7.0	Major	Significantly changed the technical content.
11/03/2010	8.0	Major	Significantly changed the technical content.
03/18/2011	8.0	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction 7
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

2 Messages 11
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 11
2.2.1.1.1.3 StoreState 12
2.2.1.1.1.4 EssdnSize 12
2.2.1.1.1.5 Essdn 12
2.2.1.1.2 Redirect Response 13
2.2.1.1.2.1 LogonFlags 13
2.2.1.1.2.2 ServerNameSize 13
2.2.1.1.2.3 ServerName 13
2.2.1.1.3 Success Response for Private Mailbox 13
2.2.1.1.3.1 LogonFlags 13
2.2.1.1.3.2 FolderIds 13
2.2.1.1.3.3 ResponseFlags 14
2.2.1.1.3.4 MailboxGuid 14
2.2.1.1.3.5 ReplId 14
2.2.1.1.3.6 ReplGuid 14
2.2.1.1.3.7 LogonTime 14
2.2.1.1.3.8 GwartTime 14
2.2.1.1.3.9 StoreState 15
2.2.1.1.4 Success Response for Public Folders 15
2.2.1.1.4.1 LogonFlags 15
2.2.1.1.4.2 FolderIds 15
2.2.1.1.4.3 ReplId 15
2.2.1.1.4.4 ReplGuid 15
2.2.1.1.4.5 PerUserGuid 16
2.2.1.1.5 ReturnValue 16
2.2.1.2 RopGetReceiveFolder Semantics 16
2.2.1.2.1 Request 16
2.2.1.2.1.1 MessageClass 17

2.2.1.2.2 Response 17
2.2.1.2.2.1 FolderId 17
2.2.1.2.2.2 ExplicitMessageClass 17
2.2.1.2.3 ReturnValue 17
2.2.1.3 RopSetReceiveFolder Semantics 18
2.2.1.3.1 Request 18
2.2.1.3.1.1 FolderId 18
2.2.1.3.1.2 MessageClass 18
2.2.1.3.2 Response 18
2.2.1.3.3 ReturnValue 18
2.2.1.4 RopGetReceiveFolderTable Semantics 19
2.2.1.4.1 Request 19
2.2.1.4.2 Response 19
2.2.1.4.2.1 RowCount 19
2.2.1.4.2.2 Rows 19
2.2.1.4.3 ReturnValue 20
2.2.1.5 RopGetStoreState Semantics 20
2.2.1.5.1 Request 20
2.2.1.5.2 Response 20
2.2.1.5.2.1 StoreState 20
2.2.1.5.3 ReturnValue 20
2.2.1.6 RopGetOwningServers Semantics 21
2.2.1.6.1 Request 21
2.2.1.6.1.1 FolderId 21
2.2.1.6.2 Response 21
2.2.1.6.2.1 OwningServersCount 21
2.2.1.6.2.2 CheapServersCount 21
2.2.1.6.2.3 OwningServers 21
2.2.1.6.3 ReturnValue 22
2.2.1.7 RopPublicFolderIsGhosted Semantics 22
2.2.1.7.1 Request 22
2.2.1.7.1.1 FolderId 22
2.2.1.7.2 Response 22
2.2.1.7.2.1 IsGhosted 22
2.2.1.7.2.2 ServersCount 22
2.2.1.7.2.3 CheapServersCount 22
2.2.1.7.2.4 Servers 23
2.2.1.7.3 ReturnValue 23
2.2.1.8 RopLongTermIdFromId Semantics 23
2.2.1.8.1 Request 23
2.2.1.8.1.1 ObjectId 23
2.2.1.8.2 Response 23
2.2.1.8.2.1 LongTermId 23
2.2.1.8.3 ReturnValue 24
2.2.1.9 RopIdFromLongTermId Semantics 24
2.2.1.9.1 Request 24
2.2.1.9.1.1 LongTermId 24
2.2.1.9.2 Response 24
2.2.1.9.2.1 ObjectId 24
2.2.1.9.3 ReturnValue 24
2.2.1.10 RopGetPerUserLongTermIds Semantics 25
2.2.1.10.1 Request 25
2.2.1.10.1.1 DatabaseGuid 25

2.2.1.10.2 Response 25
2.2.1.10.2.1 LongTermIdCount 25
2.2.1.10.2.2 LongTermIds 25
2.2.1.10.3 ReturnValue 25
2.2.1.11 RopGetPerUserGuid Semantics 25
2.2.1.11.1 Request 25
2.2.1.11.1.1 LongTermId 26
2.2.1.11.2 Response 26
2.2.1.11.2.1 DatabaseGuid 26
2.2.1.11.3 ReturnValue 26
2.2.1.12 RopReadPerUserInformation Semantics 26
2.2.1.12.1 Request 26
2.2.1.12.1.1 FolderId 26
2.2.1.12.1.2 Reserved 26
2.2.1.12.1.3 DataOffset 26
2.2.1.12.1.4 MaxDataSize 27
2.2.1.12.2 Response 27
2.2.1.12.2.1 HasFinished 27
2.2.1.12.2.2 DataSize 27
2.2.1.12.2.3 Data 27
2.2.1.12.3 ReturnValue 27
2.2.1.13 RopWritePerUserInformation Semantics 28
2.2.1.13.1 Request 28
2.2.1.13.1.1 FolderId 28
2.2.1.13.1.2 HasFinished 28
2.2.1.13.1.3 DataOffset 28
2.2.1.13.1.4 DataSize 28
2.2.1.13.1.5 Data 28
2.2.1.13.1.6 ReplGuid 28
2.2.1.13.2 Response 29
2.2.1.13.3 ReturnValue 29
2.2.2 Logon-Specific Properties 29
2.2.2.1 Private Mailbox Logon 29
2.2.2.2 Public Folders Logon 31

3 Protocol Details 33
3.1 Client Details 33
3.1.1 Abstract Data Model 33
3.1.2 Timers 33
3.1.3 Initialization 33
3.1.4 Higher-Layer Triggered Events 33
3.1.4.1 Logging on to a Store 33
3.1.4.2 Converting Between Long-Term IDs and Short-Term IDs 33
3.1.4.3 Syncing Per-User Read/Unread Data for Public Folders 34
3.1.4.4 Registering for Notifications 34
3.1.5 Message Processing Events and Sequencing Rules 34
3.1.5.1 Logon Failure or Connection Failure 34
3.1.5.2 Streaming of Per-User Read/Unread Data 35
3.1.6 Timer Events 36
3.1.7 Other Local Events 36
3.2 Server Details 36
3.2.1 Abstract Data Model 36
3.2.2 Timers 37

3.2.3 Initialization 37
3.2.4 Higher-Layer Triggered Events 37
3.2.5 Message Processing Events and Sequencing Rules 37
3.2.5.1 Processing RopLogon 38
3.2.5.1.1 Private Mailbox Logon 38
3.2.5.1.2 Public Folders Logon 39
3.2.5.2 Processing RopGetReceiveFolder 40
3.2.5.3 Processing RopSetReceiveFolder 41
3.2.5.4 Processing RopGetReceiveFolderTable 41
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 46
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 48
3.2.6 Timer Events 48
3.2.7 Other Local Events 48

4 Protocol Examples 49
4.1 RopLogon for a Private Mailbox 49
4.2 RopLogon for Public Folders 50
4.3 RopGetReceiveFolder 51
4.4 RopSetReceiveFolder 52
4.5 RopGetReceiveFolderTable 52
4.6 RopIdFromLongTermId 52
4.7 RopGetPerUserLongTermIds 53
4.8 RopReadPerUserInformation 53
4.9 RopWritePerUserInformation 54

5 Security 55
5.1 Security Considerations for Implementers 55
5.2 Index of Security Parameters 55

6 Appendix A: Product Behavior 56

7 Change Tracking 59

8 Index 60

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-GLOS]:

Active Directory
ASCII
code page
Coordinated Universal Time (UTC)
distinguished name (DN)
GUID
handle
little-endian
remote procedure call (RPC)
Unicode

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

active replica
address type
binary large object (BLOB)
change number set (CNSET)
double-byte character set (DBCS)
enterprise/site/server distinguished name (ESSDN)
EntryID
folder associated information (FAI)
Gateway Address Routing Table (GWART)
Global Address List (GAL)
Inbox folder
local replica
Logon object
LogonID
LongTermID
mailbox
message class
Out of Office (OOF)
public folder
Receive folder
remote operation (ROP)
replica
replica GUID (REPLGUID)
replica ID (REPLID)
Root folder
ROP request
ROP request buffer
ROP response
ROP response buffer
Server object
special folder

store
Store object

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. 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", July 2007.

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

[MS-OXCFOLD] Microsoft Corporation, "Folder Object Protocol Specification", June 2008.

[MS-OXCFXICS] Microsoft Corporation, "Bulk Data Transfer Protocol Specification", June 2008.

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

[MS-OXCNOTIF] Microsoft Corporation, "Core Notifications Protocol Specification", June 2008.

[MS-OXCPRPT] Microsoft Corporation, "Property and Stream Object Protocol Specification", June 2008.

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

[MS-OXCRPC] Microsoft Corporation, "Wire Format Protocol Specification", June 2008.

[MS-OXDISCO] Microsoft Corporation, "Autodiscover HTTP Service Protocol Specification", June 2008.

[MS-OXDSCLI] Microsoft Corporation, "Autodiscover Publishing and Lookup Protocol Specification", June 2008.

[MS-OXODLGT] Microsoft Corporation, "Delegate Access Configuration Protocol Specification", June 2008.

[MS-OXORULE] Microsoft Corporation, "E-Mail Rules Protocol Specification", June 2008.

[MS-OXOSFLD] Microsoft Corporation, "Special Folders Protocol Specification", June 2008.

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

[MS-OXWOOF] Microsoft Corporation, "Out of Office (OOF) Web Service Protocol Specification", June 2008.

[MS-UCODEREF] Microsoft Corporation, "Windows Protocols Unicode Reference", July 2007.

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

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

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

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 section 2.2.1.1.

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** (section 2.2.1.1), 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 (RPC)** 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 ROP request buffers and ROP response buffers that are specific to the Store Object protocol.

2.2.1.1 RopLogon Semantics

The syntax of the **RopLogon** request and response buffer is specified in [MS-OXCROPS] section 2.2.3.1.

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

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. The client MUST NOT set any unspecified flags.

Name	Value	Description
Private	0x01	This bit is set for logon to a private mailbox and is not set for logon to public folders.
Undercover	0x02	This bit is ignored by the server.
Ghosted	0x04	If the Private bit is set, then the Ghosted bit MUST NOT be set by the client and MUST be ignored by the server. If the **OpenFlags** field has either the ALTERNATE_SERVER bit or the IGNORE_HOME_MDB bit set, then the Ghosted bit MUST be ignored by the server. If the Ghosted bit is set, the client is requesting a logon to the public folder database that is present on the server. If the Ghosted bit is not set, the client is requesting a logon to the default public folder database.

2.2.1.1.1.2 OpenFlags

Note Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

Contains additional flags that control the behavior of the logon. Individual flag values and their meanings are specified in the following table. The client MUST NOT set any unspecified flags.

Name	Value	Description
USE_ADMIN_PRIVILEGE	0x00000001	A request for administrative access to the mailbox.
HOME_LOGON	0x00000004	This bit is ignored.<1>
TAKE_OWNERSHIP	0x00000008	This bit is ignored.<2>
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 message database 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	A request for a nonmessaging logon session. Nonmessaging sessions allow clients to access the store, but do not allow messages to be sent or received.
USE_PER_MDB_REPLID_MAPPING	0x01000000	For a private–mailbox logon. This bit SHOULD<3> be set. For logons to a public folder store, this bit is ignored.
SUPPORT_PROGRESS	0x20000000	Indicates that the client supports asynchronous processing of **RopSetReadFlags**, as specified in [MS-OXCMSG] section 2.2.3.10.1.1.<4>

2.2.1.1.1.3 StoreState

Unused. This field MUST be set to 0x00000000 by the client and MUST be ignored by the server.

2.2.1.1.1.4 EssdnSize

The size, in bytes, of the **Essdn** field. 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 fields in the following sections are included in the **RopLogon** response when the value of the **ReturnValue** field is 0x00000478 (ecWrongServer).

2.2.1.1.2.1 LogonFlags

Composed of the Private, Undercover, and Ghosted flags. The server returns these flags unchanged from the **LogonFlags** field of the **RopLogon** request (section 2.2.1.1.1). The client MUST ignore all other flags.

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 connect 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 (section 2.2.1.1.1).

2.2.1.1.3.1 LogonFlags

Composed of the Private, Undercover, and Ghosted flags. The server returns these flags unchanged from the **LogonFlags** field of the **RopLogon** request (section 2.2.1.1.1). The client MUST ignore all other flags.

2.2.1.1.3.2 FolderIds

Identifies the folder ID (FID) ([MS-OXCDATA] section 2.2.1.1) of all of the following folders:

- Mailbox **Root folder**. All other folders listed here are direct or indirect children of this folder.
- Deferred Action
- Spooler Queue
- Interpersonal messages subtree (Root folder 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	This bit MUST be set and MUST be ignored by the client.
OwnerRight	0x02	The user has owner permission on the mailbox.
SendAsRight	0x04	The user has the right to send mail from the mailbox.
OOF	0x10	The **Out of Office (OOF)** state is set on the mailbox. For details about the OOF state, see [MS-OXWOOF].

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 on the same RPC session.

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 2.2.3.1.2.1.

2.2.1.1.3.8 GwartTime

Contains a numeric value that tracks the currency of the **Gateway Address Routing Table (GWART)**. This value is unique for each update of the GWART.

The client can use the value of the **GwartTime** field to determine whether the client's **address type** configuration data is current. If the most recent value of **GwartTime** matches the one that was returned on the previous logon to the mailbox, the client's address-type configuration data is up-to-date.

The client only uses the value of **GwartTime** in a comparison to detect a change; it does not interpret the value of **GwartTime** in any way.

2.2.1.1.3.9 StoreState

Unused. This field MUST be set to 0x00000000 by the server and MUST be ignored by the client. <5>

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 (section 2.2.1.1.1).

2.2.1.1.4.1 LogonFlags

Composed of the Private, Undercover, and Ghosted flags. The server returns these flags unchanged from the **LogonFlags** field of the **RopLogon** request (section 2.2.1.1.1). The client MUST ignore all other flags.

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
- Free/Busy Data
- Offline address book Data
- EForms Registry for the user's locale
- Local Site's Free/Busy Data
- Local Site's Offline Address Book Data
- NNTP Article Index
- 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.

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 zeros).<6>

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 **Essdn** field.<7>
ecUnknownUser	0x000003EB	The user that is specified by the **Essdn** field is unknown to the system.
ecUnknownCodePage	0x000003EF	The **code page** 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 **OpenFlags** 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 made more than five attempts within a 10-second period to log on to a mailbox that is not hosted on the server.

2.2.1.2 RopGetReceiveFolder Semantics

The syntax of the **RopGetReceiveFolder** request and response is specified in [MS-OXCROPS] section 2.2.3.2.

RopGetReceiveFolder is used to determine the **Receive folder** for messages of a specific **message class**. This ROP examines the message class string and returns the folder ID (FID) ([MS-OXCDATA] section 2.2.1.1) of the Receive folder to which messages of that class and all subclasses are 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

A string that specifies the message class. The string includes the terminating NULL character. Examination of the string is 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 to which messages are being delivered. The folder MUST be a folder within the user's mailbox.

2.2.1.2.2.2 ExplicitMessageClass

A string specifying the message class that is actually configured for the Receive folder. The string includes the terminating NULL character, and the case of the characters in the string is insignificant. 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.

2.2.1.2.3 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 **MessageClass** value does not conform to the format requirements specified in section 2.2.1.2.1.1.

Name	Value	Meaning
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 2.2.3.3.

RopSetReceiveFolder is used to establish the Receive folder for messages of a given message class. As a result, all messages of the specified message class will be delivered to the folder identified by the specified FID.

Multiple message classes are permitted to be registered to the same Receive 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.

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 Receive folder for the message class and all nonspecifically configured subclasses of that 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.

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 2.2.1.3.1.2.
ecError	0x80004005	The FID (specified in the **FolderId** field) is all zeros AND the message class string (specified in the **MessageClass** 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 2.2.3.4

RopGetReceiveFolderTable is used to obtain a comprehensive list of all configured message classes and their associated Receive folders.

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] section 2.8.1.1 and [MS-OXCDATA] section 2.8.1.2, 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 ([MS-OXPROPS] section 2.776) — A **PtypInteger64** value that specifies the folder ID (FID) [MS-OXCDATA] section 2.2.1.1) 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 ([MS-OXPROPS] section 2.884)— A **PtypString8** 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 ([MS-OXPROPS] section 2.861) — 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 2.2.3.5.

RopGetStoreState is used to obtain state information about the current mailbox.<8>

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 2.2.3.5.1. 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
Success	0x00000000	The operation succeeded.
NotSupported	0x80040102	The ROP was not performed against a private mailbox logon.
NotImplemented	0x80040FFF	The server does not implement this ROP.

2.2.1.6 RopGetOwningServers Semantics

The syntax of the **RopGetOwningServers** request and response is specified in [MS-OXCROPS] section 2.2.3.6.

RopGetOwningServers is used to obtain the set of servers that host content for a replicated public folder.

This ROP is useful for a situation in which the client issues a ROP that reads content from a public folder on a specific server (**RopGetContentsTable** ([MS-OXCROPS] section 2.2.4.14), **RopOpenMessage** ([MS-OXCROPS] section 2.2.6.1), and **RopCreateMessage** ([MS-OXCROPS] section 2.2.6.2) are such ROPs), and the server that receives the request does not contain a replica, resulting in a failure of the ROP with ecNoReplicaHere (0x00000468). The client can issue **RopGetOwningServers** to obtain the set of servers that do contain a replica.

2.2.1.6.1 Request

This operation SHOULD be issued against a public folders logon.<9>

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 null-terminated ASCII strings. Each string is the ESSDN of a public folder database that hosts an **active replica** of the content of the folder. The number of strings MUST be equal to the value specified in the **OwningServersCount** field. The entries are sorted by the server's interpretation of the network cost for each entry in the list.

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 2.2.3.7.

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.

2.2.1.7.1 Request

This operation SHOULD only be issued against a public folders logon.

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 value that is TRUE if the server is not an active replica of the public folder; otherwise, FALSE. If the client issues the operation against a private mailbox logon for any folder, the value of this field MUST be FALSE. If the client issues this operation against the IPM subtree or the non-IPM subtree public folders, the value of this field MUST be FALSE. Other fields are included in the response only when the **IsGhosted** field is set to TRUE.

2.2.1.7.2.2 ServersCount

Identifies the number of strings contained in the **Servers** field. This field is present if the **IsGhosted** field is set to TRUE and is not present otherwise.

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. This field is present if the **IsGhosted** field is set to TRUE and is not present otherwise.

2.2.1.7.2.4 Servers

Contains an array of null-terminated ASCII strings. Each string is the ESSDN of a public folder database that itself hosts an active replica of the content of the folder. The number of strings MUST be equal to the value specified in the **ServersCount** field. The entries are sorted by the server's interpretation of the network cost for each entry in the list.

This field is present if the **IsGhosted** field is set to TRUE and is not present otherwise.

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.
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 2.2.3.8.

RopLongTermIdFromId is used to obtain the long-term ID, given the ShortTermID FID, as specified in [MS-OXCDATA] section 2.2.1.1 or MID, as specified in [MS-OXCDATA] section 2.2.1.2). For more details about long-term IDs, see [MS-OXCDATA] section 2.2.1.3.1. For more details about short-term IDs, see [MS-OXCDATA] section 2.2.1.1 and section 2.2.1.2.

2.2.1.8.1 Request

2.2.1.8.1.1 ObjectId

Contains the ShortTermID to be mapped to a **LongTermID**. The short-term ID is a 64-bit value composed of a 16-bit replica ID (REPLID) followed by a 48-bit global counter. The 16-bit REPLID portion of the short-term 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

Contains the long-term ID that is mapped from the given short-term ID. The long-term ID is formatted as a **LongTermID** structure, which is specified in [MS-OXCDATA] section 2.2.1.3.1.

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 2.2.3.9.

RopIdFromLongTermId is used to obtain the short-term ID, given the long-term ID.

2.2.1.9.1 Request

2.2.1.9.1.1 LongTermId

Contains the long-term ID to be mapped to the short-term ID. The long-term ID is formatted as a **LongTermID** structure, which is specified in [MS-OXCDATA] section 2.2.1.3.1.

2.2.1.9.2 Response

2.2.1.9.2.1 ObjectId

Contains the short-term ID that is mapped from the given long-term ID. The short-term ID is a 64-bit value composed of the 16-bit replica ID (REPLID) followed by the 48-bit global counter portion of the given long-term ID.

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-OXCDATA] section 2.4.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecInvalidParam	0x80070057	The **LongTermId** field of the request contained zeros for either the replica GUID component or the counter component.
ecParameterOverflow	0x00000450	The number of replica IDs registered is at the maximum. (The maximum is 32,768, but the implementation can impose a lower limit.)

2.2.1.10 RopGetPerUserLongTermIds Semantics

The syntax of the **RopGetPerUserLongTermIds** request and response is specified in the [MS-OXCROPS] section 2.2.3.10.

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** (section 2.2.1.1) 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 long-term IDs 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 2.2.3.11.

RopGetPerUserGuid obtains the REPLGUID of the public store that previously provided the now cached per-user read/unread data for a specific public folder. For more details about how the client uses **RopGetPerUserGuid**, see section 3.1.4.3.

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 public folder database for which relevant read/unread information was cached.

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 2.2.3.12.

RopReadPerUserInformation is used to obtain a set of change numbers, each of which is associated with a message that the user has read in a specific public folder.

When this ROP is issued against a private mailbox logon, cached data for the public folder is retrieved. When this ROP is issued against a public folders logon, the current per-user read/unread data for the public folder is retrieved. The client can use this ROP in conjunction with **RopWritePerUserInformation**, which is specified in section 2.2.1.13, to synchronize per-user read/unread data for a public folder.

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 ([MS-OXCROPS] section 2.2.3.12.1). 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 ([MS-OXCROPS] section 2.2.3.12.2). In other words, if **HasFinished** equals FALSE, then **DataOffset** MUST be updated, as follows, after each **RopReadPerUserInformation** response.

```
DataOffset = DataOffset + DataSize
```

2.2.1.12.1.4 MaxDataSize

Specifies the maximum amount of data to be returned to the client in a single **RopReadPerUserInformation** response ([MS-OXCROPS] section 2.2.3.12.2). 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 ([MS-OXCROPS] section 2.2.3.12.1) 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 the last block of data is being returned. The value of this field is TRUE if the last block of data is being returned and FALSE otherwise. This field MUST be set to TRUE if the underlying data has not changed since the last successful retrieval of per-user read/unread data, as specified in section 3.1.5.2.

2.2.1.12.2.2 DataSize

Contains the size, in bytes, of the data being returned. The value of this field MUST be less than or equal to the value of the **MaxDataSize** field of the request. This value MUST be zero if the underlying data has not changed since the last successful retrieval of per-user read/unread data, as specified in section 3.1.5.2.

2.2.1.12.2.3 Data

Contains the **change number set (CNSET)**, which is serialized into a **binary large object (BLOB)**. The format of the BLOB is the same as that of a serialized identifier set, which is specified in [MS-OXCFXICS] section 2.2.2.4.2. The size of the BLOB MUST be equal to the value specified in the **DataSize** field. 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** ([MS-OXCROPS] section 2.2.3.13).

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.

Name	Value	Meaning
ecRpcFormat	0x000004B6	The **DataOffset** value was less than zero.
ecError	0x80004005	The **DataOffset** value was greater than the data size.

2.2.1.13 RopWritePerUserInformation Semantics

The syntax of the **RopWritePerUserInformation** request and response is specified in [MS-OXCROPS] section 2.2.3.13.

RopWritePerUserInformation is used to establish the set of change numbers of messages the user has read in a specific public folder.

When this ROP is issued against a private mailbox logon, data for the public folder is saved. When this ROP is issued against a public folders logon, the current per-user read/unread data for the public folder is established. The client can use this ROP in conjunction with **RopReadPerUserInformation**, which is specified in section 2.2.1.12, to synchronize per-user read/unread data for a public folder.

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 the **Data** field contains the last block of data to be written. If the **Data** field contains the last block of data to be written, this field is set to TRUE; otherwise, this field is set to FALSE.

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 CNSET to be written. The CNSET is the one received in the **Data** field of the **RopReadPerUserInformation** response. The size of the data MUST be equal to the value specified in the **DataSize** field.

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** (section 2.2.1.1) issued against a public store. For more details, see section 2.2.1.1.4.4.

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 **DataOffset** didn't match the size of the data written so far OR the **FolderId** didn't match the value on the previous call, AND THEN **DataOffset** wasn't zero.
ecFmtError	0x000004ED	The data cumulatively written could not be parsed as a proper serialized IDSET with REPLGUIDs, as specified in [MS-OXCFXICS] section 2.2.2.4.2.

2.2.2 Logon-Specific Properties

The following properties are available on Logon objects. A Logon object is obtained by issuing a **RopLogon** request (section 2.2.1.1.1), 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.

To read any of the readable properties, the client issues a **RopGetPropertiesSpecific** ROP with the Logon object obtained from a successful invocation of **RopLogon** (section 2.2.1.1). To write any of the writable properties, the client issues a **RopSetProperties** ROP with the Logon object obtained from a successful invocation of **RopLogon**. To delete any of the deletable properties, the client issues **RopDeleteProperties** with the Logon object obtained from a successful invocation of **RopLogon**. For more details about **RopSetProperties**, **RopGetPropertiesSpecific**, or **RopDeleteProperties** 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
PidTagExtendedRuleSizeLimit [MS-OXPROPS] section 2.770()	X		
PidTagMaximumSubmitMessageSize ([MS-OXPROPS] section 2.878)	X		
PidTagProhibitReceiveQuota ([MS-OXPROPS] section 2.968)	X		
PidTagProhibitSendQuota ([MS-OXPROPS] section 2.969)	X		

Name	Get	Set	Delete
PidTagStoreState ([MS-OXPROPS] section 2.1140)	X		
PidTagComment ([MS-OXPROPS] section 2.706)	X	X	
PidTagContentCount ([MS-OXPROPS] section 2.716)	X		
PidTagDeleteAfterSubmit ([MS-OXPROPS] section 2.740)	X	X	X
PidTagDisplayName ([MS-OXPROPS] section 2.746)	X	X	
PidTagMailboxOwnerEntryId ([MS-OXPROPS] section 2.874)	X		
PidTagMailboxOwnerName ([MS-OXPROPS] section 2.875)	X		
PidTagMessageSize ([MS-OXPROPS] section 2.893)	X		
PidTagMessageSizeExtended ([MS-OXPROPS] section 2.894)	X		
PidTagOutOfOfficeState ([MS-OXPROPS] section 2.948)	X	X	
PidTagUserEntryId ([MS-OXPROPS] section 2.1170)	X		
PidTagSentMailSvrEID ([MS-OXPROPS] section 2.1116)	X	X	X
PidTagCodePageId ([MS-OXPROPS] section 2.705)	X		
PidTagLocaleId ([MS-OXPROPS] section 2.871)	X	X	
PidTagSortLocaleId ([MS-OXPROPS] section 2.1126)		X	

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

Name	Description
PidTagExtendedRuleSizeLimit ([MS-OXPROPS] section 2.770)	Maximum size, in bytes, the user is allowed to accumulate for a single "extended" rule. For details of extended rules, see [MS-OXORULE] section 2.2.4.
PidTagMaximumSubmitMessageSize ([MS-OXPROPS] section 2.878)	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.
PidTagProhibitReceiveQuota ([MS-OXPROPS] section 2.968)	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.
PidTagProhibitSendQuota ([MS-OXPROPS] section 2.969)	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.
PidTagStoreState ([MS-OXPROPS] section 2.1140)	Indicates whether the mailbox has any active search folders. The value FALSE indicates that the mailbox does not have active search folders.
PidTagComment ([MS-OXPROPS] section 2.706)	A mailbox comment.

Name	Description
PidTagContentCount ([MS-OXPROPS] section 2.716)	Cumulative count of non-**FAI** messages in the mailbox.
PidTagDeleteAfterSubmit ([MS-OXPROPS] section 2.740)	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.
PidTagDisplayName ([MS-OXPROPS] section 2.746)	The mailbox display name.
PidTagMailboxOwnerEntryId ([MS-OXPROPS] section 2.874)	The **EntryID** in the **Global Address List (GAL)** of the owner of the mailbox.
PidTagMailboxOwnerName ([MS-OXPROPS] section 2.875)	Display name of the owner of the mailbox.
PidTagMessageSize ([MS-OXPROPS] section 2.893)	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.
PidTagMessageSizeExtended ([MS-OXPROPS] section 2.894)	Cumulative size, in bytes, of all content in the mailbox.
PidTagOutOfOfficeState ([MS-OXPROPS] section 2.948)	Indicates whether the user is Out of Office (OOF). The value TRUE indicates that the user is OOF, in which case the out of office rules are evaluated and executed. When the value of this property is reset, regardless of the value, the accumulated OOF history is cleared for all OOF rules. For more details about rules, see [MS-OXORULE].
PidTagUserEntryId ([MS-OXPROPS] section 2.1170)	Address book EntryID of the user logged on to the mailbox.
PidTagSentMailSvrEID ([MS-OXPROPS] section 2.1116)	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.
PidTagCodePageId ([MS-OXPROPS] section 2.705)	Establishes the client code page for **Unicode** to **double-byte character set (DBCS)** string conversion. For details, see [MS-UCODEREF] section 2.2.1.
PidTagLocaleId ([MS-OXPROPS] section 2.871)	Establishes the language locale for translating system-generated messages, such as delivery reports. For more details, see [MS-LCID].
PidTagSortLocaleId ([MS-OXPROPS] section 2.1126)	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
PidTagUserEntryId ([MS-OXPROPS] section 2.1170)	X		

Name	Get	Set	Delete
PidTagAddressBookMessageId ([MS-OXPROPS] section 2.615)	X		

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

Name	Description
PidTagUserEntryId ([MS-OXPROPS] section 2.1170)	Address book EntryID of the user logged on to the public folders.
PidTagAddressBookMessageId ([MS-OXPROPS] section 2.615)	Short-term MID of the first message in the local site's offline address book public folder, if it exists and has a **local replica**. The property MUST 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

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.

cache of REPLID / REPLGUID mapping: The client maintains a cache of the mapping between REPLIDs and REPLGUIDs used for short-term IDs and long-term IDs.

cache of per-user data: The client maintains a cache of per-user data currently stored in the private store. This enables the client to sync per-user data only 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

When the user opens the client application, the client connects to the server by calling the **EcDoConnectEx** method, which is specified in [MS-OXCRPC] section 3.1.4.11. Once the client has successfully connected to the server, the client logs on to the store by sending a **RopLogon** request (section 3.1.4.2). When the client sends **RopLogon**, the client MUST specify a **LogonID** to be used in the ROP request buffer. For more details about logging on and the **LogonID**, see [MS-OXCROPS] section 3.1.4.2

After successfully logging on, the client SHOULD cache the REPLGUID. In some cases, the client will have to re-attempt the logon. For more details, see section 3.1.5.1. The client cannot attempt any additional ROPs until it successfully logs on to the store.

3.1.4.2 Converting Between Long-Term IDs and Short-Term IDs

When the client needs to persist an ID across logon sessions, the client MUST first convert the short-term ID to a long-term ID by using **RopLongTermIdFromId** (section 2.2.1.8). Short-term IDs use a 16-bit REPLID in place of a REPLGUID. Short-term IDs MUST NOT be persisted in any storage that could be accessed on a different logon session for the same mailbox. Any short-term IDs cached in non-persistent storage MUST be forgotten (deleted from non-persistent storage) if the client reconnects to the server, issues a **RopLogon** (section 2.2.1.1), and the return value of REPLGUID is different from the value obtained from a previous **RopLogon**. To persist IDs in any long-term storage, the client MUST first convert the ID to a long-term ID.

When the client needs to specify a short-term ID in a ROP request, the client uses **RopIdFromLongTermId** (section 2.2.1.9) to convert a long-term ID into a short-term ID. Most ROPs that take IDs require the short-term ID.

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

When the user marks an item as read/unread, switches to a different public folder, or logs off from the store, the client synchronizes the per-user read/unread data for the public folder. Other high-level events, as determined by the implementer, can trigger a synchronization.

Public folder data is replicated across multiple servers, with each server maintaining per-user read/unread data for each public 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 public folder, the client issues a **RopReadPerUserInformation** (section 2.2.1.12) against the public store (this is not necessary if the public folder has not been modified) to retrieve the per-user read/unread data. This data is then stored in the private store by using **RopWritePerUserInformation** (section 2.2.1.13).

When a public 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** (section 2.2.1.11) 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 **ReplGuid** field of the public folders logon response) (section 2.2.1.1). 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 synching 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. For details, see section 3.1.5.2.

3.1.4.4 Registering for Notifications

When the user opens the client application, the client registers to receive notifications for a store by using the Core Notifications protocol, as specified in [MS-OXCNOTIF]. The notifications for which the client registers are determined by the implementer. The various events for which the server sends a notification are listed in [MS-OXCNOTIF] section 2.2.1.1.

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 Logon Failure or Connection Failure

If the server returns the value ecWrongServer in the **ReturnValue** field of the **RopLogon** response (section 3.1.4.11), then the client SHOULD create a new RPC connection to the server that is specified by the **ServerName** field of the response. Using that connection, the client then re-attempts the logon. For more details about creating the RPC connection, see [MS-OXCRPC] section 3.1.4.11. For more details about logging on to a store, see section 3.1.4.1.

If the server returns either ecUnknownUser or ecLoginFailure in the **ReturnValue** field of the **RopLogon** response (section 2.2.1.1), then the client SHOULD use the Autodiscover HTTP Service

protocol [MS-OXDISCO] in order to attempt to retrieve updated user and server information.<10> If successful, the client can attempt to log on 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 if no changes are detected, then the client MUST fail the logon.

If the client is unable to establish an RPC connection to a public folder store, then 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 (section 2.2.1.1.1) to the private store. The client MUST set the ALTERNATE_SERVER flag in the **OpenFlags** field of the **RopLogon** request. The logon request returns ecWrongServer and redirects 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 the **OpenFlags** field.

If the RPC session to the server is lost and then reconnected, then the existing logon is invalid. The client MUST log on again by calling **RopLogon** (the client can reuse the existing LogonID). Additionally, all objects (folders, messages, and tables) that were opened on the original logon are now invalid and MUST be re-opened. The new REPLGUID returned by **RopLogon** MUST be compared to the cached value. If the GUIDs are different, then 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.

3.1.5.2 Streaming of Per-User Read/Unread Data

When synchronizing the per-user read/unread data for a public folder, the size of the returned data can exceed the maximum amount of data that can be sent in a single ROP. For this reason, **RopReadPerUserInformation** (section 2.2.1.12) and **RopWritePerUserInformation** (section 2.2.1.13) are designed to stream the data by means of multiple invocations until all data is received or sent. The client MUST complete the streaming of data for one public folder before commencing streaming operations for another public folder on the same server logon.

RopReadPerUserInformation Streaming

If the **HasFinished** field of the **RopReadPerUserInformation** response is set to FALSE, indicating that there is more data to be retrieved for the public folder, the client sends another **RopReadPerUserInformation** request. The client continues to send **RopReadPerUserInformation** requests until all data is retrieved from the server.

If multiple requests are necessary to retrieve all of the data, the client MUST specify an updated value in the **DataOffset** field of the next request so that **DataOffset** always points to the first byte of the next block of data to be returned. The new value is equal to the sum of the value used in the previous **RopReadPerUserInformation** request and the value of the **DataSize** field of the previous **RopReadPerUserInformation** response. In other words, if **HasFinished** is set to FALSE, **DataOffset** is set as follows in the next **RopReadPerUserInformation** request.

DataOffset = DataOffset + DataSize

If multiple requests are necessary to retrieve all of the data, the **MaxDataSize** field can be set to a different value in each invocation of the ROP. This is completely at the client's discretion.

The per-user read/unread data has been completely retrieved when the **HasFinished** field of the response is set to TRUE. The client MUST NOT interpret the value of the **Data** field of the **RopReadPerUserInformation** response. The client simply provides the value unaltered in a future sequence of invocations of **RopWritePerUserInformation**.

RopWritePerUserInformation Streaming

The client sends the data as it was received in the **RopReadPerUserInformation** response. The client continues to send **RopWritePerUserInformation** requests until all of the data is sent to the server. The client sets **HasFinished** to TRUE when the last block of data is sent.

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 maintains 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, named property-to-property ID mapping table, a mailbox table, a per-user data table, and a Receive folder table.

REPLID / REPLGUID mapping table: The REPLID and REPLGUID to-and-from mapping table contains rows of 16-bit REPLID values coupled with 128-bit REPLGUID values. This table is used to map a REPLID to a REPLGUID and vice versa. When a client invokes **RopIdFromLongTermId** (section 2.2.1.9)**,** this table is searched for the REPLGUID portion of the long-term ID passed by the client. If a row with that REPLGUID is found, then the associated REPLID is used to formulate the returned short-term ID. 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 short-term ID. 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 short-term ID passed by the client. If a row is found, then the associated REPLGUID is used to formulate the returned long-term ID. If a row is not found, then the ROP fails. A REPLID MUST NOT have a value of zero.

named property / property ID mapping table: The named property-to-property ID mapping table contains the mapping between registered named properties and their server-assigned property identifiers. The server uses this table to look up the named property for a given property ID and vice versa when processing **RopGetNamesFromPropertyIds** and **RopGetPropertyIdsFromNames**, as specified in [MS-OXCPRPT] section 3.2.5.9 and [MS-OXCPRPT] section 3.2.5.10

mailbox table: 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 folder and other **special folders** of the mailbox, the access permissions to the mailbox, and 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** (section 1.3), 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 indicates the proper server to log on to for this user's mailbox, and contains 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 the user's access is

checked. If the logon is allowed, then the FIDs of various special folders are obtained from the table and returned to the client. For more details about special folders, see [MS-OXOSFLD]. A list of the folders that are special folders is provided in [MS-OXOSFLD] section 1.3.

per-user data table: 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, and the change number set of read items. The **RopGetPerUserLongTermIds** ([MS-OXCROPS] section 2.2.3.10), **RopGetPerUserGuid** ([MS-OXCROPS] section 2.2.3.11), **RopReadPerUserInformation** ([MS-OXCROPS] section 2.2.3.12), **RopWritePerUserInformation** ([MS-OXCROPS] section 2.2.3.13) ROPs each make use of the data in this table.

Receive folder table: The Receive folder table contains rows of message class strings and associated FIDs. Each FID specifies a Receive folder. The server maintains a single Receive folder table per database. The data within the table are scoped to each mailbox. Conceptually, there is a single Receive folder table per mailbox. The delivery process uses the message class string on the incoming e-mail to look up the appropriate folder to which to deliver that message. The **RopGetReceiveFolder** ([MS-OXCROPS] section 2.2.3.2) and **RopSetReceiveFolder** ([MS-OXCROPS] section 2.2.3.3) ROPs are used to retrieve and set 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 take steps to ensure that it does not re-issue a REPLGUID that was issued prior to the restoration. These steps are implementation-specific. <11>

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** (section 3.1.4.11), 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 the **LogonFlags** field has the Private bit set, the logon is going to a private mailbox. Otherwise, the 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 client attempts to log on to a mailbox that is in transit, the server MUST fail the operation with 0x0000050C (ecMailboxInTransit) in the **ReturnValue** field. If the client attempts to logon to a mailbox that is disabled, the server MUST fail the operation with 0x0000096C (ecMailboxDisabled) in the **ReturnValue** field.

If the user's configured mailbox is not hosted by this server, the server determines 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 client attempts to log on to a nonhome message database (MDB) and the USE_ADMIN_PRIVILEGE bit in the **OpenFlags** field is not set, the server MUST fail the operation with 0x0000011C (ecProfileNotConfigured) in the **ReturnValue** field.

If the client specified an invalid code page for a string or a Server object, the server MUST fail the operation with 0x000003EF (ecUnknownCodePage) in the **ReturnValue** field.

If the client has made more than five attempts within a 10-second period to log on to a mailbox that is not hosted on the server, the server MUST fail the operation with 0x0000047F (ecServerPaused) in the **ReturnValue** field.

If the client sets an undefined flag in either the **LogonFlags** field or the **OpenFlags** field, the server SHOULD<12> fail the operation with 0x80004005 (ecError) in the **ReturnValue** field of the response.

If the SUPPORT_PROGRESS flag is set, the server responds as specified in [MS-OXCMSG] section 3.2.5.10<13>

For the USE_PER_MDB_REPLID_MAPPING flag of the **OpenFlags** field, the server SHOULD<14> have the following behavior:

- 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 flag 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 flag of the **OpenFlags** field to see if it is set. If the USE_PER_MDB_REPLID_MAPPING flag is not set, then the server SHOULD<15> 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 flag is set, then the server takes no action.<16>

If the user does not match the owner of the mailbox, then the ROP MUST fail with a **ReturnValue** of 0x80070005; otherwise, the server checks the user's permissions to determine whether the user is an owner of the mailbox or a delegate. An owner is not required to have security settings checked before performing any non-administrative operations on the mailbox. For more details about delegates, see [MS-OXODLGT].

The server then finds the mailbox in the mailbox table. If the mailbox is not present in the table and the user has owner permission on the mailbox, the server creates 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 does not create the mailbox if the user does not have 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 then determines the appropriate FIDs to return to the client. For details, see section 2.2.1.1.3.2. The server returns the appropriate REPLGUID in the **ReplGuid** field. If the server returns the same REPLGUID for different logons, the server MUST use the same REPLID-to-REPLGUID mapping and named property-to-property ID mapping for those different logons.

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 confirms the user logging on to the public folders has a mailbox in the organization. The server performs 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 client sets an undefined flag in either the **LogonFlags** field or the **OpenFlags** field, the server SHOULD<17> fail the operation with 0x80004005 (ecError) in the **ReturnValue** field of the response.

If the **OpenFlags** field has the ALTERNATE_SERVER bit set, the server searches 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 (ecLoginFailure). Otherwise, the operation MUST fail with a **ReturnValue** of 0x00000478 (ecWrongServer). For more details about properly forming the return values when a **ReturnValue** of 0x00000478 is sent, see section 2.2.1.1.2.

If the **OpenFlags** field has either the ALTERNATE_SERVER bit or the IGNORE_HOME_MDB bit set, the server ignores the Ghosted bit of the **LogonFlags** field. Otherwise, the server has the following behavior: If the Ghosted bit of the **LogonFlags** field is set, then the server uses the public folder database that is present on the server. If there is no public folder database on the server, then the

server responds with the **ReturnValue** field set to 0x80040111 (ecLoginFailure). If the Ghosted bit is not set, the server uses the default public folder database for the logon. If the server does not host that database, the server MUST fail the operation with a **ReturnValue** of 0x00000478 (ecWrongServer). For details about properly forming the response 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 (ecLoginFailure).

The server then determines 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 verifies 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 the **ReturnValue** field set to 0x80040102.

The server validates the value of the **MessageClass** field, as specified in section 2.2.1.2.1.1. If the value does not conform to the requirements, then the server MUST fail the operation with the **ReturnValue** field set to 0x80070057.

The server then searches the Receive folder table to find the entry with the longest case-insensitive prefix string that matches the value of the **MessageClass** field. The search is scoped to the mailbox that is identified by the logon. The server then retrieves the actual message class string from the Receive folder table, and the associated folder ID (FID) ([MS-OXCDATA] section 2.2.1.1). 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 returns an empty string in the **ExplicitMessageClass** field and the FID for the user's **Inbox folder** in the **FolderId** field. If a match is found, the server returns a string specifying the actual configured message class and the FID of the associated Receive folder. The server can case-fold the string to all uppercase or all lowercase, or leave the string as stored.

For example, if the client sends "IPM.Schedule.Meeting.Request" in the **MessageClass** field, the string returned in the **ExplicitMessageClass** field might be "IPM.Schedule.Meeting", which implies that all messages that are message class "IPM.Schedule.Meeting" (or that are a subclass of "IPM.Schedule.Meeting") will be delivered to the folder that is associated with the "IPM.Schedule.Meeting" message class. In this same example, if a client sends "IPM.Schedule.Meeting" in the **MessageClass** field, 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 an empty string in the **ExplicitMessageClass** field and the FID for the user's Inbox folder in the **FolderId** field. If the client requests "IPM.MY.Class", the server will return "IPM" in the **ExplicitMessageClass** field and the FID for the user's Inbox folder in the **FolderId** field.

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's **RopGetReceiveFolder** request specifies the message class "MY.Class.SOMETHING", the server will return the string "MY.Class" and the FID registered for "MY.Class".

3.2.5.3 Processing RopSetReceiveFolder

The server verifies 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, then the server MUST fail the operation with the **ReturnValue** field set to 0x80040102.

The server validates the value of the **MessageClass** field, as specified in section 2.2.1.3.1.2. If the value does not conform to the requirements, then the server MUST fail the operation with the **ReturnValue** field set to 0x80070057.

If the value of the **MessageClass** field is a case-insensitive match to either "IPM" or "Report.IPM", then the server MUST fail the operation with the **ReturnValue** field set to 0x80070005. If the **MessageClass** field is set to a zero-length string and the **FolderId** field is set to zero, then the server MUST fail the operation with the **ReturnValue** field set to 0x80004005.

The server searches the Receive folder table using a case-insensitive string comparison for an exact match to the value of the **MessageClass** field. The search is scoped to the mailbox that is identified by the logon. If a match is found, the value of the **FolderId** field replaces the FID stored in the table on that row. If the **FolderId** field is set to zero, then the table row for the specified message class is deleted from the Receive folder table. (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 is set to the current system-time of the server, adjusted to Coordinated Universal Time (UTC).

The Receive folder table is initialized when the mailbox is created, as specified in section 3.2.3. 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 ([MS-OXPROPS] section 2.776)) — Contains the folder ID (FID) ([MS-OXCDATA] section 2.2.1.1) 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 ([MS-OXPROPS] section 2.884)) — Contains a string that specifies the message class that is configured for the Receive folder.

3. "Last-modification Time" column (**PidTagLastModificationTime** property ([MS-OXPROPS] section 2.861)) — Contains the current system-time, in UTC, when the entry was created or last modified.

3.2.5.4 Processing RopGetReceiveFolderTable

The server verifies 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 return all rows of the Receive folder table for the mailbox identified by the logon. If there are no entries in the Receive folder table, the server MUST fail the operation with 0x00000463 (ecNoReceiveFolder) in the **ReturnValue** field. The **Rows** field of the **RopGetReceiveFolderTable** response (section 2.8.1) 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 2.8.1 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 value of 0x80040FFF (NotImplemented) in the **ReturnValue** field of the response. Servers MAY implement this ROP as specified in this section.<18>

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

- The server verifies 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, then the server MUST fail the operation with a **ReturnValue** of 0x80040102 (NotSupported).
- If the mailbox has any persisted search folders, then the server MUST set the STORE_HAS_SEARCHES flag in the response. If the mailbox does not have any persisted search folders, then the server MUST NOT set the STORE_HAS_SEARCHES flag in the response.
- The server MUST NOT set any other flags in the response.

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. If the public folder specified by the **FolderId** field cannot be found in the public folder database, the server MUST fail the operation with 0x8004010F (ecNotFound) in the **ReturnValue** field.

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 retrieves 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 obtains the network cost for each server in the list, if that cost information isn't already in the list. The source used to determine these network costs can be whatever configuration source the server finds most appropriate.<19> The network cost values are expressed relative to the server servicing the request, not the client making the request.

The server removes 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

algorithm used to determine the servers that are too expensive is implementation-defined.<20> 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 sorts 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 counts 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 RopPublicFolderIsGhosted

If the operation is issued against a private mailbox store, the server MUST return FALSE in the **IsGhosted** field of the response. In this case, no replication state data is returned. If the public folder specified by the **FolderId** field cannot be found in the public folder database, the server MUST fail the operation with 0x8004010F (ecNotFound) in the **ReturnValue** field.

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 retrieves 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 obtains the network cost for each server in the list, if that cost information isn't already in the list. The source used to determine these network costs can be whatever configuration source the server finds most appropriate.<21> The network cost values are expressed relative to the server, not the client making the request.

The server MUST return TRUE in the **IsGhosted** field if the queried server is not listed as an active replica of the folder. The value of the **IsGhosted** field MUST be FALSE if the queried server is listed as an active replica of the folder. If the client issues this operation against the IPM subtree or the non-IPM subtree public folders, the value of the **IsGhosted** field MUST be FALSE.

The server removes 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 algorithm used to determine the servers that are too expensive is implementation-

defined.<22> 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 **IsGhosted** value of TRUE.

The server sorts 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 counts 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 searches the REPLID and REPLGUID to-and-from mapping table for the replica ID (REPLID) portion of the given short-term ID. The server does not attempt to confirm that the given short-term 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 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 map the same REPLID to the same replica GUID (REPLGUID) every time it is queried. Other servers can map a particular REPLID to a different REPLGUID, but each server MUST map any particular REPLID to the same REPLGUID every time it is queried.

After obtaining the REPLGUID from the REPLID and REPLGUID to-and-from mapping table, the server uses the REPLGUID to construct the 192-bit long-term ID, which is returned in the **LongTermId** field. The long-term ID consists of the 128-bit REPLGUID, followed by the 48-bit global counter portion of the given short-term ID, followed by 16 bits of padding set to 0x0000.

3.2.5.9 Processing RopIdFromLongTermId

If the **LongTermId** field of the request contains zeros for either the replica GUID (REPLGUID) component or the global counter component, the server MUST fail the operation with 0x80070057 (ecInvalidParam) in the **ReturnValue** field.

The server searches the REPLID and REPLGUID to-and-from mapping table for the REPLGUID portion of the long-term ID. The server does not attempt to confirm that the long-term 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 REPLGUID is not found, the server adds a new entry consisting of the REPLGUID portion of the long-term ID and a newly assigned replica ID (REPLID). The new REPLID MUST be unique in the REPLID and REPLGUID to-and-from mapping table. If the maximum number of REPLIDs have already been registered, the server cannot register a new REPLID. In this case, the server MUST fail the operation with 0x00000450 (ecParameterOverflow) in the **ReturnValue** field.

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, but each server MUST map any particular REPLGUID to the same REPLID every time it is queried.

The server ignores the content of the padding bytes in the long-term ID.

After obtaining the REPLID from the REPLID and REPLGUID to-and-from mapping table, the server uses the REPLID to construct the 64-bit short-term ID, which is returned in the **ObjectId** field. The short-term ID consists of the 16-bit REPLID followed by the 48-bit global counter portion of the given long-term ID.

3.2.5.10 Processing RopGetPerUserLongTermIds

The server verifies that the operation is being performed against a private mailbox logon, not against 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 searches the per-user data table of the mailbox for entries identified by the **DatabaseGuid** field in the request. For each entry in the table, the server collects 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 verifies 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 searches the per-user data table for the mailbox for the only row with an FID that is associated with the public folder specified by the **LongTermId** field in the request. The server returns the associated REPLGUID value in the **DatabaseGuid** field. If the public folder specified by the **LongTermId** field cannot be found, the server MUST fail the operation with 0x8004010F (ecNotFound) in the **ReturnValue** 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 searches the per-user data table for the mailbox for the only row with an FID equal to the value of the **FolderId** field. If the row exists, then the server retrieves from that row the stored change number set of read items. If the row does not exist, then the server returns an empty array in the **Data** field of the response. 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 first determines 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 searches 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. If the row exists, the server retrieves from that row the stored change number set of read items. If the row does not exist, then the server returns an empty array in the **Data** field of the response. 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

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 ([MS-OXPROPS] section 2.698) is added to the change number set (CNSET). If the user marks a message as unread, the current value of that message's **PidTagChangeNumber** property is removed from the CNSET.

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 2.2.2.4.2. 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 (section 2.2.3.12). For this reason, the **RopReadPerUserInformation** ([MS-OXCROPS] section 2.2.3.12) 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 (section 2.2.1.12.1) to retrieve the BLOB in segments.

On each invocation of **RopReadPerUserInformation**, the server inspects 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 compares 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** field 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. If the value of the **DataOffset** field is less than zero, the server MUST fail the operation with 0x000004B6 (ecRpcFormat) in the **ReturnValue** field. If the value of the **DataOffset** field is greater than the size of the next BLOB segment to be returned, the server MUST fail the operation with 0x80004005 (ecError) in the **ReturnValue** field.

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 examines the value of MaxDataSize on each invocation of **RopReadPerUserInformation** as follows.

    1. The server compares 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.<23>

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.<24>

3. The server compares 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 aggregates the data across multiple invocations and it validates the entire data set before persisting to permanent storage.

The server determines whether 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 validates the accumulated data and verifies that it is a properly formed serialized IDSET with REPLGUID as specified in [MS-OXCFXICS] section 2.2.2.4.2. 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 records, in UTC, the current system time on the appropriate row in the table.

3.2.5.13.2 Private Mailbox Specific Behavior

The server searches 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 searches the per-user data table for the only row with a user ID equal to the 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 **ROP 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's 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 **TaggedPropertyValue** structure format specified in [MS-OXCDATA] section 2.11.4). 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** ([MS-OXPROPS] section 2.1063))

[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 2.11.1.

4.1 RopLogon for a Private Mailbox

RopLogon request (section 2.2.1.1.1) 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 (section 2.2.1.1.3):

```
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 (section 2.2.1.1.1) for public folders:

```
0000: 00 04 04 00 01 00 00 00-00 00 00
```

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

[0001-0004] **OpenFlags** — 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 (section 2.2.1.1.4):

```
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 (section 2.2.1.2.1):

```
0000: 00
```

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

RopGetReceiveFolder response (section 2.2.1.2.2):

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

[0000-0007] **FolderId**

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

4.4 RopSetReceiveFolder

RopSetReceiveFolder request (section 2.2.1.3.1):

```
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 (section 2.2.1.3.2):

No response.

4.5 RopGetReceiveFolderTable

RopGetReceiveFolderTable request (section 2.2.1.4.1):

No fields in the request.

RopGetReceiveFolderTable response (section 2.2.1.4.2):

```
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)

[0004-005B] **Rows** — As follows:

[0004-0004], [0016-0016], [002B-002B], [0040-0040] Error Flag Indicator (no error)

[0005-000C], [0017-001E], [002C-0033], [0041-0048] **PidTagFolderId** property ([MS-OXPROPS] section 2.776)

[000D-000D], [001F-0022], [0034-0037], [0049-0053] **PidTagMessageClass** property ([MS-OXPROPS] section 2.884)

[000E-0015], [0023-002A], [0038-003F], [0054-005B] **PidTagLastModificationTime** property ([MS-OXPROPS] section 2.861)

4.6 RopIdFromLongTermId

RopIdFromLongTermId request (section 2.2.1.9.1):

```
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 (section 2.2.1.9.2):

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

[0000-0001] **ObjectId REPLID**

[0002-0007] **ObjectId** counter

4.7 RopGetPerUserLongTermIds

RopGetPerUserLongTermIds request (section 2.2.1.10.1):

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

[0000-000F] **DatabaseGuid**

RopGetPerUserLongTermIds response (section 2.2.1.10.2):

```
0000: 00 00
```

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

4.8 RopReadPerUserInformation

RopReadPerUserInformation request (section 2.2.1.12.1):

```
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 00
```

[0000-0017] **FolderId**

[0018-0018] **Reserved**

[0019-001C] **DataOffset**

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

RopReadPerUserInformation response (section 2.2.1.12.2):

```
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 (section 2.2.1.13.1) :

```
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 (section 2.2.1.13.2):

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 Parameters

None.

6 Appendix A: Product Behavior

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

- Microsoft® Exchange Server 2003
- Microsoft® Exchange Server 2007
- Microsoft® Exchange Server 2010
- Microsoft® Exchange Server 2010 Service Pack 1 (SP1)
- Microsoft® Office Outlook® 2003
- Microsoft® Office Outlook® 2007
- Microsoft® Outlook® 2010
- Microsoft® Outlook® 2010 Service Pack 1 (SP1) Beta

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

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

<1> Section 2.2.1.1.1.2: 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.

<2> Section 2.2.1.1.1.2: Exchange 2003 and Exchange 2007 use the TAKE_OWNERSHIP bit as follows: If set, then the server checks to determine whether the user can act as an owner of the mailbox. If not set, then the user is considered a delegate.

<3> Section 2.2.1.1.1.2: Office Outlook 2003 does not set this bit. Office Outlook 2007 uses this bit to control whether the server maintains one REPLID-to-REPLGUID mapping and one named property-to-property ID mapping for all logon sessions.

<4> Section 2.2.1.1.1.2: Exchange 2010 SP1 and Outlook 2010 SP1 Beta implement this flag. Exchange 2003, Exchange 2007, Exchange 2010, Office Outlook 2003, Office Outlook 2007, and Outlook 2010, do not implement this flag.

<5> Section 2.2.1.1.3.9: If the mailbox currently has any active search folders, then Exchange 2003 and Exchange 2007 set this field to 0x01000000; otherwise, this field is set to 0x00000000. For more details about search folders, see [MS-OXCFOLD].

<6> Section 2.2.1.1.4.5: Exchange 2007 does not set the **PerUserGuid** field to an empty GUID.

<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 2.2.1.5: Exchange 2010 does not implement the **RopGetStoreState** remote operation (ROP) ([MS-OXCROPS] section 2.2.3.5), but it is implemented in Exchange 2003 and Exchange 2007.

<9> Section 2.2.1.6.1: Exchange 2003, Exchange 2007, and Exchange 2010 successfully complete a **RopGetOwningServers** ROP ([MS-OXCROPS] section 2.2.3.6) when issued against a private mailbox logon, but the results are undefined.

<10> Section 3.1.5.1: The Autodiscover HTTP Service protocol, as specified in [MS-OXDISCO], is not supported by Office Outlook 2003.

<11> Section 3.2.3: When a database is restored from backup, Exchange 2003 and Exchange 2007 assign a new randomly-generated REPLGUID to the database and then add this new REPLGUID, along with a new REPLID, to the REPLID and REPLGUID to-and-from mapping table.

<12> Section 3.2.5.1.1: The behavior of Exchange 2003, Exchange 2007, and Exchange 2010 is undefined if the client sets an undefined flag in either the **LogonFlags** field or the **OpenFlags** field.

<13> Section 3.2.5.1.1: Exchange 2010 SP1 implements the SUPPORT_PROGRESS flag. Exchange 2003, Exchange 2007, and Exchange 2010 do not implement the flag.

<14> Section 3.2.5.1.1: Exchange 2003 ignores the USE_PER_MDB_REPLID_MAPPING flag, and therefore, the behavior of Exchange 2003 is not affected by this flag. Exchange 2003 always maintains one REPLID-to-REPLGUID mapping and one named property-to-property ID mapping per RPC session, and these mappings are shared by all logons on the RPC session.

<15> Section 3.2.5.1.1: If the USE_PER_MDB_REPLID_MAPPING flag is not set, Exchange 2007 does not fail the ROP and instead has the following behavior: Exchange 2007 maintains one REPLID-to-REPLGUID mapping and one named property-to-property ID mapping per RPC session, and these mappings are shared by all logons on the RPC session.

<16> Section 3.2.5.1.1: If the USE_PER_MDB_REPLID_MAPPING flag is set, Exchange 2007 maintains one REPLID-to-REPLGUID mapping and one named property-to-property ID mapping for each logon session.

<17> Section 3.2.5.1.2: The behavior of Exchange 2003, Exchange 2007, and Exchange 2010 is undefined if the client sets an undefined flag in either the **LogonFlags** field or the **OpenFlags** field.

<18> Section 3.2.5.5: Exchange 2003 and Exchange 2007 implement the **RopGetStoreState** ROP ([MS-OXCROPS] section 2.2.3.5).

<19> Section 3.2.5.6: Exchange 2003 queries the transport engine for cost information. Exchange 2007 and Exchange 2010 query Active Directory for cost information.

<20> Section 3.2.5.6: Exchange 2003 removes servers that have a connection cost of "infinite". Exchange 2007 and Exchange 2010 remove servers that have a connection cost greater than 500.

<21> Section 3.2.5.7: Exchange 2003 queries the transport engine for cost information. Exchange 2007 and Exchange 2010 query Active Directory for cost information.

<22> Section 3.2.5.7: Exchange 2003 removes servers that have a connection cost of "infinite". Exchange 2007 and Exchange 2010 remove servers that have a connection cost greater than 500.

<23> Section 3.2.5.12.3: Exchange 2003, Exchange 2007, and Exchange 2010 use 4096 for the default value.

<24> Section 3.2.5.12.3: Exchange 2003, Exchange 2007, and Exchange 2010 use 4096 for the maximum value.

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8 Index

A

Abstract data model
  client 33
  server 36
Applicability 10

C

Capability negotiation 10
Change tracking 59
Client
  abstract data model 33
  initialization 33
  other local events 36
  timer events 36
  timers 33

D

Data model - abstract
  client 33
  server 36

F

Fields - vendor-extensible 10

G

Glossary 7

H

Higher-layer triggered events
  server 37

I

Implementer - security considerations 55
Index of security parameters 55
Informative references 9
Initialization
  client 33
  server 37
Introduction 7

L

Logon-Specific Properties message 29

M

Message processing
  server 37
Messages
  Logon-Specific Properties 29
  Remote Operations 11
  transport 11

N

Normative references 8

O

Other local events
  client 36
  server 48

P

Parameters - security index 55
Preconditions 10
Prerequisites 10
Product behavior 56

R

References
  informative 9
  normative 8
Relationship to other protocols 10
Remote Operations message 11

S

Security
  implementer considerations 55
  parameter index 55
Sequencing rules
  server 37
Server
  abstract data model 36
  higher-layer triggered events 37
  initialization 37
  message processing 37
  other local events 48
  sequencing rules 37
  timer events 48
  timers 37
Standards assignments 10

T

Timer events
  client 36
  server 48
Timers
  client 33
  server 37
Tracking changes 59
Transport 11
Triggered events - higher-layer
  server 37

V

Vendor-extensible fields 10
Versioning 10