[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.

Contents

1 Introduction

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

1.1 Glossary

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

Active Directory
active replica
address type
ASCII
attachment
binary large object (BLOB)
change number (CN)
change number set (CNSET)
code page (1)
Coordinated Universal Time (UTC)
delegate
distinguished name (DN)
double-byte character set (DBCS)
enterprise/site/server distinguished name (ESSDN)
EntryID
folder
folder associated information (FAI)
folder ID (FID)
free/busy
Gateway Address Routing Table (GWART)
Global Address List (GAL)
GUID
handle
IDSET
Inbox folder
IPM subtree
little-endian
local replica
Logon object
LogonID
LongTermID
mailbox
message
message class
message database (MDB)
message ID (MID)
named property
non-IPM subtree
notification
offline address book (OAB)
Out of Office (OOF)
property (1)
property ID

property tag
public folder
Receive folder
remote operation (ROP)
remote procedure call (RPC)
replica (1)
replica GUID (REPLGUID)
replica ID (REPLID)
Root folder
ROP request
ROP request buffer
ROP response
ROP response buffer
Server object
ShortTermID
special folder
Store object
store
table
Unicode

The following terms are specific to this document:

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

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

1.2 References

1.2.1 Normative References

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

[MS-LCID] Microsoft Corporation, "Windows Language Code Identifier (LCID) Reference", March 2007, http://msdn.microsoft.com/en-us/library/cc233965.aspx

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

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

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

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

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

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

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

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

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

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

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

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

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

[MS-UCODEREF] Microsoft Corporation, "Windows Protocols Unicode Reference", July 2007, http://msdn.microsoft.com/en-us/library/cc248954.aspx

[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-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 this document.

1.5 Prerequisites/Preconditions

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

1.6 Applicability Statement

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

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The **ROP request buffers** and **ROP response buffers** specified by this protocol are sent to and received from the server by using the underlying **remote procedure call (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 fields passed in ROP request buffers that are specific to the Store Object protocol.

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

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

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

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

2.2.1.1.1.1 LogonFlags

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

Name	Value	Description
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, then the server uses the public folder database that is present on the server for the logon. If there is no public folder database on the server, then the server responds with the **ReturnValue** field set to ecLoginFailure. If the Ghosted bit is not set, then server uses the default public folder database for the logon. If the server does not host that database, then the server responds with the **ReturnValue** field set to ecWrongServer. For more details about the **ReturnValue** field, see section 2.2.1.1.5.

2.2.1.1.1.2 OpenFlags

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

Name	Value	Description
USE_ADMIN_PRIVILEGE	0x00000001	When set, this bit indicates that the user is requesting administrative access to the mailbox. To grant administrative access to the mailbox, the server MUST confirm that the user has the right to such access. Confirmation is implementation-dependent.
PUBLIC	0x00000002	If set, RopLogon opens the public folders store. Otherwise, RopLogon opens a private user mailbox.
HOME_LOGON	0x00000004	This bit is ignored.<2>
TAKE_OWNERSHIP	0x00000008	This bit is ignored.<3>
ALTERNATE_SERVER	0x00000100	Requests a private server to provide an alternate public server.
IGNORE_HOME_MDB	0x00000200	This bit is used only for public logons. When set, this bit allows the client to log on to a public **MDB** that is not the user's default public MDB; otherwise, attempts to log on to a public MDB that is not the user's default results in the client being redirected back to the user's default public MDB.
NO_MAIL	0x00000400	Requests a non-messaging logon session. Non-messaging sessions allow clients to access the store, but do not allow messages to be sent or received.
USE_PER_MDB_REPLID_MAPPING	0x01000000	For a private–mailbox logon, the client uses this bit to control server behavior as specified in section 2.2.1.1.1.2.1. For logons to a public folder store, this bit is ignored.

2.2.1.1.1.2.1 USE_PER_MDB_REPLID_MAPPING Details

The client uses the USE_PER_MDB_REPLID_MAPPING bit of the **OpenFlags** field to control how the server maintains the **replica ID (REPLID)**-to-**replica GUID (REPLGUID)** mapping and **named property**-to-**property ID** mapping on an RPC session.<4>

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

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

2.2.1.1.1.3 StoreState

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

2.2.1.1.3 Success Response for Private Mailbox

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

2.2.1.1.3.1 LogonFlags

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

2.2.1.1.3.2 FolderIds

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

- Mailbox **Root folder**. All other folders listed here are direct or indirect children of this folder.
- Deferred Action
- 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	MUST be set.
OwnerRight	0x02	If set, the user has full-owner rights for the mailbox.
SendAsRight	0x04	If set, the user has the right to send mail from this mailbox.
OOF	0x10	Indicates whether **Out of Office (OOF)** is set for the mailbox. For more 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 made to the server on the same RPC session. If the client did set theUSE_PER_MDB_REPLID MAPPING bit in the **OpenFlags** field, the server can return values as deemed appropriate by the implementation.

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

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

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)**. The server generates a new numeric value with 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. The client compares the most recent value of **GwartTime** with the one that was returned on the previous logon to the mailbox. Matching values of **GwartTime** indicate that 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.

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. 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 Bookk 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**.<6>

2.2.1.1.4.5 PerUserGuid

This field is not used and is ignored by the client. The server SHOULD set this field to an empty GUID (all zeros). <7>

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.<8>
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) 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 delivery to the Receive folder. The string includes the terminating NULL character. The string MUST meet the following requirements:

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

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

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

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

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

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

2.2.1.2.3 ReturnValue

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

Name	Value	Meaning
ecNone	0x00000000	Success.
ecInvalidParam	0x80070057	The **MessageClass** value does not conform to the format requirements specified in section 2.2.1.2.1.1.
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 that have a message class string that itself has a prefix of a given string. The request includes a message class string and an FID. As a result, all messages with a message class that matches this given class will be delivered to the folder identified by the FID. Message class matches are determined using a case-insensitive prefix string match. For example, a configured class of "xx.yy" will match a message with a class of "Xx.YY.ZZ".

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

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

2.2.1.3.1 Request

This operation MUST be issued against a private mailbox logon.

2.2.1.3.1.1 FolderId

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

2.2.1.3.1.2 MessageClass

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

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

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

2.2.1.3.2 Response

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

2.2.1.3.3 ReturnValue

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

Name	Value	Meaning
ecNone	0x00000000	Success.
ecAccessDenied	0x80070005	The client has attempted to change the Receive folder for the "IPM" or "Report.IPM" classes.
ecInvalidParam	0x80070057	The message class string does not conform to the requirements specified in section 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 to delivery folder entries. The return data consists of logical "rows" of data, each row consisting of three "columns" of values. There is one row for each configured entry, and within each row are the message class, FID and last modification time for the entry.

2.2.1.4.1 Request

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

2.2.1.4.2 Response

2.2.1.4.2.1 RowCount

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

2.2.1.4.2.2 Rows

An array that contains the rows of the Receive folder table. Each row is returned in either a **StandardPropertyRow** structure or a **FlaggedPropertyRow** structure, both of which are specified in [MS-OXCDATA] sections 2.8.1.1 and 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 — A **PtypInteger64** value that specifies the folder ID (FID) of the Receive folder, which is the folder to which messages of the specified message class will be delivered. The Receive folder MUST be a folder that is within the user's mailbox.

2. PidTagMessageClass property — A **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 — 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.<9>

2.2.1.5.1 Request

There are no explicit fields for this operation. The data to be retrieved is limited to the mailbox that is linked to the **LogonID** that is passed as part of the ROP request, as specified in [MS-OXCROPS] section 2.2.3.5.1. 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.

When the client issues a ROP that reads content from a public folder on a specific server (RopGetContentsTable, RopOpenMessage, and RopCreateMessage are such ROPs), the operation can fail with ecNoReplicaHere (0x00000468). This happens if the server that receives the request does not contain a replica copy of the data. In that event, the client issues RopGetOwningServers to obtain the set of servers that do contain the data. The returned data is an ordered set of server

names, sorted by the configured network costs that connect this server to each of the other servers. The cost data can come from a specific configuration for this server, or can be inferred from other network configuration settings (specific site router costs, for example).

2.2.1.6.1 Request

This operation SHOULD be issued against a public folders logon. <10>

2.2.1.6.1.1 FolderId

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

2.2.1.6.2 Response

2.2.1.6.2.1 OwningServersCount

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

2.2.1.6.2.2 CheapServersCount

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

2.2.1.6.2.3 OwningServers

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

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

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

2.2.1.6.3 ReturnValue

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

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

2.2.1.7 RopPublicFolderIsGhosted Semantics

The syntax of the RopPublicFolderIsGhosted request and response is specified in [MS-OXCROPS] section 2.2.3.7.

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. The ROP returns TRUE in the **IsGhosted** field when the server is not an active replica for the folder and returns FALSE when the server is an active replica.

2.2.1.7.1 Request

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

2.2.1.7.1.1 FolderId

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

2.2.1.7.2 Response

2.2.1.7.2.1 IsGhosted

Contains a Boolean value. TRUE if the server is not an active replica of the folder; otherwise, 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.

2.2.1.7.2.3 CheapServersCount

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

2.2.1.7.2.4 Servers

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

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

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

2.2.1.7.3 ReturnValue

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

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

2.2.1.8 RopLongTermIdFromId Semantics

The syntax of the RopLongTermIdFromId request and response is specified in [MS-OXCROPS] section 2.2.3.8.

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

2.2.1.8.1 Request

2.2.1.8.1.1 ObjectId

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

2.2.1.8.2 Response

2.2.1.8.2.1 LongTermId

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

2.2.1.8.3 ReturnValue

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

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

2.2.1.9 RopIdFromLongTermId Semantics

The syntax of the RopIdFromLongTermId request and response is specified in [MS-OXCROPS] section 2.2.3.9.

RopIdFromLongTermId is used to obtain the ShortTermID, given the LongTermID.

2.2.1.9.1 Request

2.2.1.9.1.1 LongTermId

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

2.2.1.9.2 Response

2.2.1.9.2.1 ObjectId

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

2.2.1.9.3 ReturnValue

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

Name	Value	Meaning
ecNone	0x00000000	Success.

2.2.1.10 RopGetPerUserLongTermIds Semantics

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

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

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

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

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

The client can use this ROP in conjunction with RopWritePerUserInformation, which is specified in section 2.2.1.13, to move read/unread information from one public folder replica to another. For example, the client could, periodically or on a specific user action, query the public logon for read/unread information for a specific public folder by issuing a RopReadPerUserInformation request. It would then issue a RopWritePerUserInformation request against the private mailbox logon, sending the same data back to the server. This effectively saves the read/unread data in the user's mailbox. Later, when the user re-visits the public folder, the client would issue a RopReadPerUserInformation request against the private mailbox logon to retrieve the cached information for the folder. It would then issue a RopWritePerUserInformation request to the public folders logon to save back to the public database. This sequence of operations allows the user to see the same set of unread messages as the last time they visited the folder, even in the event that the client is referred to a different public folder server each time they log on.

2.2.1.12.1 Request

2.2.1.12.1.1 FolderId

Contains the LongTermID of the folder to query.

2.2.1.12.1.2 Reserved

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

2.2.1.12.1.3 DataOffset

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

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

```
DataOffset = DataOffset + DataSize
```

2.2.1.12.1.4 MaxDataSize

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

2.2.1.12.2 Response

2.2.1.12.2.1 HasFinished

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

2.2.1.12.2.2 DataSize

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

2.2.1.12.2.3 Data

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

2.2.1.12.3 ReturnValue

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

Name	Value	Meaning
ecNone	0x00000000	Success.
ecRpcFormat	0x000004B6	The **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 folder. Note, this is not a set of MIDs, but rather the change numbers of the messages at the time they were read. Messages that are modified receive a new change number, and hence, fall out of the set of read messages. The user will see these modified messages marked as unread.

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

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

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

Used in conjunction with RopReadPerUserInformation, the client is able to move read/unread information from one public folder replica to another. For example, the client could, periodically or on a specific user action, query the public logon for read/unread information for a specific public folder by issuing a RopReadPerUserInformation request. It would then issue a RopWritePerUserInformation request against the private mailbox logon, sending the same data to the mailbox server. This effectively saves the read/unread data in the user's mailbox. Later, when the user re-visits the public folder, the client would issue a RopReadPerUserInformation request against the private mailbox logon to retrieve the cached information for the folder. It would then issue a RopWritePerUserInformation request to the public folders logon to save back onto the public database. This sequence of operations allows the user to see the same set of unread messages as the last time they visited the folder, even in the event that the client is referred to a different public folder server each time they log on.

2.2.1.13.1 Request

2.2.1.13.1.1 FolderId

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

2.2.1.13.1.2 HasFinished

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

2.2.1.13.1.3 DataOffset

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

2.2.1.13.1.4 DataSize

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

2.2.1.13.1.5 Data

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

2.2.1.13.1.6 ReplGuid

MUST NOT be present for operations against public folders logons. This field MUST be present when the value of the **DataOffset** field is zero. This field MUST NOT be present when **DataOffset** is not zero. Identifies which public database was the source of this data. The value is the REPLGUID of the last database for which relevant read/unread information was obtained. This GUID is obtained from the result of a RopLogon issued against a public store. For more details, see section 2.2.1.1.4.4.

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 2.2.2.3.2.

2.2.2 Logon-Specific Properties

The following properties are available on Logon objects. A Logon object is obtained by issuing a RopLogon request, 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. 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	X		
PidTagMaximumSubmitMessageSize	X		
PidTagProhibitReceiveQuota	X		
PidTagProhibitSendQuota	X		
PidTagStoreState	X		
PidTagComment	X	X	
PidTagContentCount	X		
PidTagDeleteAfterSubmit	X	X	X
PidTagDisplayName	X	X	
PidTagMailboxOwnerEntryId	X		
PidTagMailboxOwnerName	X		
PidTagMessageSize	X		
PidTagMessageSizeExtended	X		
PidTagOutOfOfficeState	X	X	
PidTagUserEntryId	X		
PidTagSentMailSvrEID	X	X	X
PidTagCodePageId	X		
PidTagLocaleId	X	X	

Name	Get	Set	Delete
PidTagSortLocaleId		X	

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

Name	Description
PidTagExtendedRuleSizeLimit	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	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	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	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	For a full description of the **StoreState** field, see section 2.2.1.1.3.9.
PidTagComment	A mailbox comment.
PidTagContentCount	Cumulative count of non-**FAI** messages in the mailbox.
PidTagDeleteAfterSubmit	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	The mailbox display name.
PidTagMailboxOwnerEntryId	The **EntryID** in the **Global Address List (GAL)** of the owner of the mailbox.
PidTagMailboxOwnerName	Display name of the owner of the mailbox.
PidTagMessageSize	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	Cumulative size, in bytes, of all content in the mailbox.
PidTagOutOfOfficeState	Indicates whether the user is OOF. Setting to FALSE causes accumulated OOF history maintained by the rules engine to be cleared for all folders and OOF rules. Setting to TRUE allows rules set to run only when the user is out of the office to run. For more details, see [MS-OXORULE].
PidTagUserEntryId	Address book EntryID of the user logged on to the mailbox.
PidTagSentMailSvrEID	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.

Name	Description
PidTagCodePageId	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	Establishes the language locale for translating system-generated messages, such as delivery reports. For more details, see [MS-LCID].
PidTagSortLocaleId	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	X		
PidTagAddressBookMessageId	X		

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

Name	Description
PidTagUserEntryId	Address book EntryID of the user logged on to the public folders.
PidTagAddressBookMessageId	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.

The client SHOULD maintain a cached copy of the mapping between REPLIDs and REPLGUIDs used for short-term IDs and long-term IDs.

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

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Logging on to a Store

The client logs on to a store by using RopLogon before attempting any additional ROPs on the store. Prior to logging on, the client calls 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 can send a RopLogon request. 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.

3.1.4.2 Converting Between LongTermIDs and ShortTermIDs

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

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

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

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

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

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

When a folder is subsequently reopened in a later logon session, the client MUST check to see if the replica server has changed. This is done by issuing a RopGetPerUserGuid against the private store and comparing the REPLGUID returned (in the **DatabaseGuid** field) to the public store's REPLGUID, which is returned by RopLogon (in the **ReplGuid** field of the public folders logon response). 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. Because the server will cache interim data across client calls, the client MUST complete the entire streaming operation for the data of one folder before commencing streaming operations for another folder on the same server logon.

3.1.4.4 Registering for Notifications

The client can register to receive **notifications** for a store by using the Core Notifications protocol, as specified in [MS-OXCNOTIF]. 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, 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, then the client SHOULD use the Autodiscover HTTP Service protocol [MS-OXDISCO] in order to attempt to retrieve updated user and server information.<13> If successful, the client can attempt to 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 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.

3.1.6 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.

The REPLID and REPLGUID to-and-from mapping table contains rows of 16-bit REPLID values coupled with 128-bit REPLGUID values. When a client invokes RopIdFromLongTermId, this table is searched for the REPLGUID portion of the ID passed by the client. If a row with that REPLGUID is found, then the associated REPLID<14> 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 ID passed by the client. If a row is found, then the associated REPLGUID is used to formulate the return long-term ID. If a row is not found, then the ROP fails. A REPLID MUST NOT have a value of zero.

The mailbox table is used for logging on to a private mailbox. The table contains one row for each mailbox in the database. It contains columns to specify the root 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, 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 section 1.3 of [MS-OXOSFLD].

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, RopGetPerUserGuid, RopReadPerUserInformation, RopWritePerUserInformation ROPs each make use of the data in this table.

The Receive folder table contains rows of message class strings and associated FIDs.<15>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 and RopSetReceiveFolder ROPs each make use of the data in this table.

3.2.2 Timers

None.

3.2.3 Initialization

When a database is created, the database MUST be assigned a new randomly generated REPLGUID. When the REPLID and REPLGUID to-and-from mapping table is created, a single new entry MUST be added, consisting of the database REPLGUID and a newly assigned REPLID.

When a database is restored from backup, the server MUST take steps to ensure that it does not re-issue a REPLGUID that was issued prior to the restoration. These steps are implementation-specific. <16>

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

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

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

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

3.2.5.1 Processing RopLogon

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

3.2.5.1.1 Private Mailbox Logon

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

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

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

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

For the USE_PER_MDB_REPLID_MAPPING bit of the **OpenFlags** field, servers SHOULD NOT implement the behavior that is described in section 2.2.1.1.1.2.1 and SHOULD implement the following behavior:<17><18>

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

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

The server MUST then determine the appropriate FIDs to return to the user. For details, see section 2.2.1.1.3.2.

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

3.2.5.1.2 Public Folders Logon

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

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

If the **OpenFlags** field has the ALTERNATE_SERVER bit set, the server MUST search for another public folder database server in the organization. The process by which another public folder database is chosen is up to the implementation, however, the server SHOULD choose a public folder database server that is not the configured preferred server. If a suitable server cannot be found, the operation MUST fail with a **ReturnValue** of 0x80040111 (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 server being queried does not host the preferred public folder database, and the **LogonFlags** field does not have the ghosted bit set, and the **OpenFlags** field does not have the IGNORE_HOME_MDB bit set, the operation MUST fail with a **ReturnValue** of 0x00000478. For details about properly forming the return values when a **ReturnValue** of 0x00000478 is sent, see section 2.2.1.1.2.

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

The server MUST then determine the appropriate FIDs to return to the user. For more details, see section 2.2.1.1.4.2.

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

3.2.5.2 Processing RopGetReceiveFolder

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

The server MUST validate 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 MUST scan 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 MUST be scoped to the relevant mailbox. The server then MUST retrieve the actual message class string from the Receive folder table, and the associated FID. The Receive folder table is primed for the mailbox at creation time, as specified in section 3.2.3.

If no entry in the table can be matched, the server returns an empty string in the **ExplicitMessageClass** field.

If a match is found, then the server returns the actual configured message class string and the FID. The server can case-fold the string to all uppercase, all lowercase, or leave the string as stored.

3.2.5.3 Processing RopSetReceiveFolder

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

The server MUST validate 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 MUST search the Receive folder table using a case-insensitive string comparison for an exact match to the value of the **MessageClass** field. The search MUST be scoped to the relevant mailbox. If a match is found, the value of the **FolderId** field MUST replace the FID stored in the table on that row. 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 MUST be 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) — Contains the folder ID (FID) of the Receive folder, which is the folder to which messages of the specified message class will be delivered. The Receive folder MUST be a folder that is within the user's mailbox.

2. "Message Class" column (PidTagMessageClass property) — Contains a string that specifies the message class that is configured for the Receive folder.

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

3.2.5.4 Processing RopGetReceiveFolderTable

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

The server MUST return all rows of the Receive folder table for the relevant mailbox. The **Rows** field of the RopGetReceiveFolderTable response contains either a **StandardPropertyRow** structure or a **FlaggedPropertyRow** structure for each row of the Receive folder table. If there is an error retrieving any data of a row from the Receive folder table, the server returns the row formatted as a **FlaggedPropertyRow** structure; otherwise, the server returns the row formatted as a **StandardPropertyRow** structure. For more details about these structures, see [MS-OXCDATA] section 2.8.1 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.<19>

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

- The server MUST verify that the operation is being performed against a private mailbox logon, and not a public folders logon. If the operation is performed against a public folders logon, 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.

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

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

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

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

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

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

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

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

3.2.5.7 Processing RopPublicFolderIsGhosted

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

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

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

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

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

in the list. The network cost values are expressed relative to the server, not the client making the request.

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

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

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

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

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

3.2.5.8 Processing RopLongTermIdFromId

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

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

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

3.2.5.9 Processing RopIdFromLongTermId

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

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

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

3.2.5.10 Processing RopGetPerUserLongTermIds

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

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

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

3.2.5.11 Processing RopGetPerUserGuid

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

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

3.2.5.12 Processing RopReadPerUserInformation

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

3.2.5.12.1 Private Mailbox Specific Behavior

The server MUST search the per-user data table for the mailbox for the only row with an FID equal to the value of the **FolderId** field. If the row exists, then the server MUST retrieve 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 MUST first determine that the persisted read/unread information for the user is up to date. If the server is maintaining any in-memory caches of the per-user read/unread information, the data for the current user MUST now be flushed to disk.

The server MUST search the per-user data table for the only row with an FID equal to the value of the **FolderId** field and the user ID equal to the logged on user. If the row exists, the server MUST retrieve 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

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

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

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

Summary:

1. The **MaxDataSize** field of the request specifies the maximum number of bytes that can be returned in a single RopReadPerUserInformation response. The server MUST adjust the **MaxDataSize** value in certain cases, as specified in item 2 of this summary.

2. When the client retrieves a BLOB in segments, the client can set **MaxDataSize** to a different value in each RopReadPerUserInformation request that is used to retrieve the BLOB. Therefore, the server MUST examine the value of **MaxDataSize** on each invocation of RopReadPerUserInformation as follows.

    1. The server MUST compare the value of **MaxDataSize** to zero. If **MaxDataSize** equals 0, then the server MUST adjust the value of **MaxDataSize** to a suitable default value, which is determined by the implementation.<20>

    2. The server SHOULD compare the value of **MaxDataSize** to some suitable maximum value, as determined by the implementation. If **MaxDataSize** > [server's suitable maximum], then the server SHOULD adjust the value of **MaxDataSize** to the suitable maximum value.<21>

    3. The server MUST compare the adjusted value of **MaxDataSize** to the size of the remaining BLOB segment. If [size of remaining BLOB segment] > [adjusted **MaxDataSize**], then the server MUST set **HasFinished** to FALSE to indicate to the client that additional requests are necessary to retrieve all of the remaining portions of the BLOB. The size of the remaining BLOB segment is equal to the size of the entire BLOB minus the value of **DataOffset**.

3. The **DataSize** field specifies the actual number of bytes that are returned in the response. The value of **DataSize** MUST NOT exceed the adjusted value of the **MaxDataSize** field. For details about adjusting **MaxDataSize**, see item number 2 of this summary. The server MUST set **DataSize** to the lesser of the following two values:

    1. The adjusted value of **MaxDataSize**.

    2. The size of the remaining BLOB segment. This is the size of the portion of the BLOB that remains to be sent to the client and is equal to the size of the entire BLOB minus the value of **DataOffset**.

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

3.2.5.13 Processing RopWritePerUserInformation

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

3.2.5.13.1 Common Behavior

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

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

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

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

3.2.5.13.2 Private Mailbox Specific Behavior

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

3.2.5.13.3 Public Folders Specific Behavior

The server MUST search the per-user data table for the only row with a user ID equal to 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)

[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 for a private mailbox:

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

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

[0000-0000] **LogonFlags** — Private

[0001-0004] **OpenFlags** — HOME_LOGON, TAKE_OWNERSHIP, NO_MAIL, USE_PER_MDB_REPLID_MAPPING

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

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

[000B-0072] **Essdn**

RopLogon success response for a private mailbox:

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

[0000-0000] **LogonFlags** — Private

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

[0001-0008] Mailbox Root Folder FID

[0009-0010] Deferred Action Folder FID

[0011-0068] <more FIDs>

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

[006A-0079] **MailboxGuid**

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

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

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

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

[009C-009F] **StoreState** — STORE_HAS_SEARCHES

4.2 RopLogon for Public Folders

RopLogon request for public folders:

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

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

[0001-0004] **OpenFlags** — PUBLIC, HOME_LOGON, NO_MAIL, USE_PER_MDB_REPLID_MAPPING

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

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

RopLogon success response for public folders:

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

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

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

[0001-0008] Public Folders FID

[0009-0010] IPM Subtree FID

[0011-0050] <other FIDs>

[0051-0068] Unused — Set to zero.

[0069-006A] **ReplId**

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

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

4.3 RopGetReceiveFolder

RopGetReceiveFolder request:

```
0000: 00
```

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

RopGetReceiveFolder response:

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

[0000-0007] **FolderId**

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

4.4 RopSetReceiveFolder

RopSetReceiveFolder request:

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

[0000-0007] **FolderId**

[0008-001C] **MessageClass**

RopSetReceiveFolder response:

No response.

4.5 RopGetReceiveFolderTable

RopGetReceiveFolderTable request:

No fields in the request.

RopGetReceiveFolderTable response:

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

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

[0004-005B] **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

[000D-000D], [001F-0022], [0034-0037], [0049-0053] PidTagMessageClass property

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

4.6 RopIdFromLongTermId

RopIdFromLongTermId request:

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

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

[0010-0015] **LongTermID** counter

[0016-0017] **LongTermID** padding

RopIdFromLongTermId response:

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

[0000-0001] **ObjectId REPLID**

[0002-0007] **ObjectId** counter

4.7 RopGetPerUserLongTermIds

RopGetPerUserLongTermIds request:

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

[0000-000F] **DatabaseGuid**

RopGetPerUserLongTermIds response:

```
0000: 00 00
```

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

4.8 RopReadPerUserInformation

RopReadPerUserInformation request:

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

[0000-0017] **FolderId**

[0018-0018] **Reserved**

[0019-001C] **DataOffset**

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

RopReadPerUserInformation response:

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

[0000-0000] **HasFinished**

[0001-0002] **DataSize**

[0003-001A] **Data**

4.9 RopWritePerUserInformation

RopWritePerUserInformation request:

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

[0000-0017] **FolderId**

[0018-0018] **HasFinished**

[0019-001C] **DataOffset**

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

[001F-0036] **Data**

[0037-0046] **ReplGuid**

RopWritePerUserInformation response:

No response.

5 Security

5.1 Security Considerations for Implementers

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

5.2 Index of Security Fields

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products:

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

Exceptions, if any, are noted below. If a service pack number appears with the product version, behavior changed in that service pack. The new behavior also applies to subsequent service packs of the product unless otherwise specified. 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 prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that product does not follow the prescription.

<1> Section 2.2.1.1.1: Exchange 2003, Exchange 2007, and Exchange 2010 do not reject the operation when the client uses an undefined flag value.

<2> 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.

<3> 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 a full owner of the mailbox. If not set, then the user is considered a delegate.

<4> Section 2.2.1.1.1.2.1: Outlook 2003 does not use this bit and always assumes that all logons on an RPC session use the same REPLID/REPLGUID mapping and named property/property ID mapping.

<5> Section 2.2.1.1.3.9: 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 2.2.1.1.4.4: Outlook 2003, Outlook 2007, and Outlook 2010 cache the REPLGUID returned by RopLogon. If a reconnect occurs and the REPLGUID changes, Outlook 2003, Outlook 2007, and Outlook 2010 fail the logon and prompt the user to restart the application.

<7> Section 2.2.1.1.4.5: Exchange 2007 does not set the **PerUserGuid** field to an empty GUID.

<8> 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.

<9> Section 2.2.1.5: Exchange 2010 does not implement RopGetStoreState, but it is implemented in Exchange 2003 and Exchange 2007.

<10> Section 2.2.1.6.1: Exchange 2003, Exchange 2007, and Exchange 2010 successfully complete a RopGetOwningServers operation when issued against a private mailbox logon, but the results are undefined.

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

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

<13> Section 3.1.5.1: The Autodiscover HTTP Service protocol, as specified in [MS-OXDISCO], is not supported by Outlook 2003.

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

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

<16> Section 3.2.3: 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.

<17> Section 3.2.5.1.1: Exchange 2003 always ignores the USE_PER_MDB_REPLID_MAPPING bit and behaves as if the bit is not set, as described in section 2.2.1.1.1.2.1.

<18> Section 3.2.5.1.1: Exchange 2007 implements the behavior that is described in section 2.2.1.1.1.2.1.

<19> Section 3.2.5.5: Exchange 2003 and Exchange 2007 implement **RopGetStoreState**.

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

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

7 Change Tracking

This section identifies changes that were made to the [MS-OXCSTOR] protocol document between the May 2010 and August 2010 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

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

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

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

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

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

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

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

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

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

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.1 Glossary	56247 Added the following to the list of terms that are defined in [MS-OXGLOS]: Inbox folder, offline address book (OAB), property ID, Root folder, ROP request, ROP response, and Store object. Removed the following from the list of terms that are defined in [MS-OXGLOS]: interpersonal messaging subtree, and non-interpersonal messaging subtree.	N	Content update.
1.1 Glossary	55901 Removed the term "Session Context Handle (CXH)" from the list of external glossary terms.	N	Content update.
1.1 Glossary	55613 Added the term "notification" to the list of external glossary terms.	N	Content update.
1.1 Glossary	56261 Added the term "address type" to the list of external glossary terms.	N	Content update.
1.1 Glossary	56296 Added the term "delegate" to the list of external glossary terms.	N	Content update.
1.2.1 Normative References	55751 Moved [MS-OXGLOS] from Normative	N	Content update.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
	References section to Informative References section.		
1.2.1 Normative References	56297 Added reference [MS-OXOSFLD].	N	Content update.
1.2.1 Normative References	56296 Added reference [MS-OXODLGT].	N	Content update.
1.2.1 Normative References	57633 Deleted reference [MS-OXCMAIL] from list of normative references.	N	Content update.
2.2.1 Remote Operations	55608 Removed the statement about flags fields.	N	Content update.
2.2.1.1.1 Request	55892 Revised the product behavior note to specify that Exchange 2010 does not reject the operation when the client uses an undefined flag value.	N	Product behavior note updated.
2.2.1.1.1.1 LogonFlags	55608 Specified handling of unspecified flags.	N	Content update.
2.2.1.1.1.1 LogonFlags	55775 Removed the SplProcess flag from the table. Removed statements about flags being returned to the client in the response.	N	Content update.
2.2.1.1.1.1 LogonFlags	55451 Clarified the server's behavior for the Ghosted flag.	N	Content update.
2.2.1.1.1.2 OpenFlags	55608 Specified handling of unspecified flags.	N	Content update.
2.2.1.1.1.2 OpenFlags	55767 Updated the product behavior note for the TAKE_OWNERSHIP flag.	N	Product behavior note updated.
2.2.1.1.1.3 StoreState	55890 Clarified the requirement for the setting of this field.	N	Content update.
2.2.1.1.2 Redirect Response	56272 Specified the error code associated with the ecWrongServer error.	N	Content update.
2.2.1.1.2.1 LogonFlags	55608 Clarified the flag values returned and specified how the client should handle unspecified flags.	N	Content update.
2.2.1.1.3.1 LogonFlags	55608 Clarified the flag values returned and specified	N	Content update.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
	how the client should handle unspecified flags.		
2.2.1.1.3.3 ResponseFlags	55973 Removed "view admin" from the description of the OwnerRight flag.	N	Content update.
2.2.1.1.3.6 ReplGuid	55738 Removed the product behavior note about Outlook's processing of a logon response.	N	Product behavior note removed.
2.2.1.1.3.8 GwartTime	55973 Deleted the statement that references the PidTagAddressType property.	N	Content update.
2.2.1.1.3.8 GwartTime	56261 Updated the description of this field.	N	Content update.
2.2.1.1.3.9 StoreState	55893 Updated the description of the field to specify the behavior of Exchange 2010. Added a product behavior note to specify the behavior of Exchange 2003 and Exchange 2007.	N	New product behavior note added.
2.2.1.1.4.1 LogonFlags	55608 Clarified the flag values returned and specified how the client should handle unspecified flags.	N	Content update.
2.2.1.1.4.5 PerUserGuid	56009 Added a product behavior note to specify that Exchange 2007 does not set the PerUserGuid field to an empty GUID.	N	New product behavior note added.
2.2.1.1.5 ReturnValue	56550 Incorporated product behavior note for ecServerPaused into the main body of the document.	N	Product behavior note removed.
2.2.1.2.3 ReturnValue	56006 Removed the ecNoReceiveFolder error from the table of possible values.	N	Content update.
2.2.1.4.2.2 Rows	55543 Changed the data type of the PidTagMessageClass property from PtypString to PtypString8.	N	Content update.
2.2.1.5 RopGetStoreState Semantics	55954 Changed "support" to "implement" in the product behavior note.	N	Product behavior note updated.
2.2.1.5.3 ReturnValue	55954 Added the value 0x80040FFF to the table of values. Updated the name and description of code 0x00000000.	N	Content update.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
2.2.1.6 RopGetOwningServers Semantics	55874 Clarified when RopGetOwningServers is used.	N	Content update.
2.2.1.6 RopGetOwningServers Semantics	55618 Removed the product behavior note about RopGetOwningServers behavior when issued against a private mailbox logon.	N	Product behavior note removed.
2.2.1.6.1 Request	55618 Added a product behavior note about RopGetOwningServers behavior when issued against a private mailbox logon.	N	New product behavior note added.
2.2.1.7 RopPublicFolderIsGhosted Semantics	56292 Specified the field that is set to TRUE or FALSE in the response.	N	Content update.
2.2.1.7.2.1 IsGhosted	56292 Clarified which fields are included in the response and the conditions under which those fields are present. Changed the word "property" to "value".	N	Content update.
2.2.1.11 RopGetPerUserGuid Semantics	55409 Clarified the description of this ROP. Removed details about how the client uses the ROP and referenced section 3.1.4.3 for the details.	N	Content update.
2.2.1.11.2.1 DatabaseGuid	55409 Clarified the description of this field.	N	Content update.
2.2.1.12.3 ReturnValue	55941 Deleted the ecNotFound error from the table of possible values.	N	Content update.
2.2.1.13.3 ReturnValue	55934 Removed the ecNotFound error from the table of possible values.	N	Content update.
2.2.2 Logon-Specific Properties	56441 Removed the statement specifying that the server must return the ecPropSecurityViolation error when the client attempts an impermissible operation on a property.	N	Content update.
2.2.2.1 Private Mailbox Logon	56901 Deleted the statement that references [MS-OXCDATA] in the description of the PidTagSentMailSvrEID property.	N	Content update.
2.2.2.1 Private Mailbox Logon	56444 Updated the table to specify that GET is allowed on the PidTagCodePageId property and that SET is not allowed on the	N	Content update.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
	PidTagCodePageId property.		
2.2.2.1 Private Mailbox Logon	56445 Updated the table to specify that GET is allowed on the PidTagLocaleId property.	N	Content update.
3.1 Client Details	55973 Removed "none".	N	Content update.
3.1.4.1 Logging on to a Store	55621 Moved content about processing logon and connection failures to section 3.1.5.1.)	Y	Content update.
3.1.4.1 Logging on to a Store	55901 Clarified the logon process and added a reference to [MS-OXCROPS] section 3.1.4.2.	N	New content added.
3.1.4.4 Registering for Notifications	55613 Clarified how a client registers for notifications.	N	Content update.
3.1.5 Message Processing Events and Sequencing Rules	55621 Removed statement.	N	Content update.
3.1.5.1 Logon Failure or Connection Failure	55621 New section.	Y	New content added.
3.1.5.1 Logon Failure or Connection Failure	56246 Clarified the client's behavior when ecWrongServer is returned in the RopLogon response.	N	Content update.
3.2.1 Abstract Data Model	55615 Updated statement to specify constraints on a REPLID instead of server behavior.	N	Content updated for template compliance.
3.2.1 Abstract Data Model	55669 Added named property-to-property ID mapping table to the list of tables maintained by the server.	N	Content update.
3.2.1 Abstract Data Model	56297 Changed wording from "various special FIDs are obtained" to "the FIDs of various special folders are obtained" to clarify meaning. Added reference link to [MS-OXOSFLD].	N	Content update.
3.2.3 Initialization	55796 Specified Exchange 2010 behavior for issuing REPLGUIDs when a database is restored from backup and added a product behavior note to specify the behavior of Exchange 2003 and	N	New product behavior note added.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
	Exchange 2007.		
3.2.5.1.1 Private Mailbox Logon	56293 Clarified what happens when the user does not match the owner.	N	Content update.
3.2.5.1.1 Private Mailbox Logon	56294 Clarified statement regarding the OpenFlags field.	N	Content update.
3.2.5.1.1 Private Mailbox Logon	55767 Removed statements about the TAKE_OWNERSHIP flag.	N	Content update.
3.2.5.1.1 Private Mailbox Logon	56296 Clarified server behavior when the user matches the owner of the mailbox.	N	Content update.
3.2.5.1.2 Public Folders Logon	56272 Specified the names associated with error codes 0x80040111 and 0x00000478 on first mention.	N	Content update.
3.2.5.2 Processing RopGetReceiveFolder	55972 Removed product behavior note.	N	Product behavior note removed.
3.2.5.2 Processing RopGetReceiveFolder	56006 Clarified that the server returns an empty string in the ExplicitMessageClass field when no entry in the table can be matched.	N	Content update.
3.2.5.3 Processing RopSetReceiveFolder	56273 Clarified that both conditions in the statement must be true in order for the server to fail the operation with a ReturnValue of 0x80004005.	N	Content update.
3.2.5.3 Processing RopSetReceiveFolder	56006 Added a reference for details about initialization of the Receive folder table. Specified how the Receive folder table is modified when the FolderId field is set to zero for a given message class.	N	Content update.
3.2.5.5 Processing RopGetStoreState	55954 Specified 0x80040FFF (NotImplemented) as the value to be returned in the ReturnValue field when the ROP is not implemented. Changed "return value" to "response" for clarity about the flags settings.	N	Content update.
3.2.5.12.1 Private Mailbox Specific Behavior	55941 Specified the server behavior for the case in which no table entry exists for the specified FID.	N	Content update.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
3.2.5.12.2 Public Folders Specific Behavior	55941 Specified the server behavior for the case in which no table entry exists for the specified FID.	N	Content update.
4.5 RopGetReceiveFolderTable	56264 Removed the reference to [MS-OXCDATA]. Added a line showing the buffer location of the Rows field and changed the names of the values to the property names.	N	Content update.

8 Index