[MS-OXCPERM]:

Exchange Access and Operation Permissions Protocol

---

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.

- **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 can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.

- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.

- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.

- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.

- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are 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 as specifically described above, whether by implication, estoppel, or otherwise.

**Tools.** The Open Specifications documentation does 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 documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

**Support.** For questions and support, please contact dochelp@microsoft.com.
### Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/4/2008</td>
<td>0.1</td>
<td>New</td>
<td>Initial Availability.</td>
</tr>
<tr>
<td>4/25/2008</td>
<td>0.2</td>
<td>Minor</td>
<td>Revised and updated property names and other technical content.</td>
</tr>
<tr>
<td>6/27/2008</td>
<td>1.0</td>
<td>Major</td>
<td>Initial Release.</td>
</tr>
<tr>
<td>8/6/2008</td>
<td>1.01</td>
<td>Minor</td>
<td>Revised and edited technical content.</td>
</tr>
<tr>
<td>9/3/2008</td>
<td>1.02</td>
<td>Minor</td>
<td>Updated references.</td>
</tr>
<tr>
<td>12/3/2008</td>
<td>1.03</td>
<td>Minor</td>
<td>Minor editorial fixes.</td>
</tr>
<tr>
<td>4/10/2009</td>
<td>2.0</td>
<td>Major</td>
<td>Updated applicable product releases.</td>
</tr>
<tr>
<td>7/15/2009</td>
<td>3.0</td>
<td>Major</td>
<td>Revised and edited for technical content.</td>
</tr>
<tr>
<td>2/10/2010</td>
<td>3.1.0</td>
<td>Minor</td>
<td>Updated the technical content.</td>
</tr>
<tr>
<td>5/5/2010</td>
<td>3.2.0</td>
<td>Minor</td>
<td>Updated the technical content.</td>
</tr>
<tr>
<td>8/4/2010</td>
<td>3.3</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>11/3/2010</td>
<td>3.3</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>3/18/2011</td>
<td>3.4</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>8/5/2011</td>
<td>3.5</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>10/7/2011</td>
<td>3.5</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>1/20/2012</td>
<td>4.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>4/27/2012</td>
<td>5.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>7/16/2012</td>
<td>5.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>10/8/2012</td>
<td>5.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>2/11/2013</td>
<td>5.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/26/2013</td>
<td>6.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>11/18/2013</td>
<td>7.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>2/10/2014</td>
<td>7.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>4/30/2014</td>
<td>7.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>7/31/2014</td>
<td>7.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>Date</td>
<td>Revision History</td>
<td>Revision Class</td>
<td>Comments</td>
</tr>
<tr>
<td>------------</td>
<td>------------------</td>
<td>----------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>10/30/2014</td>
<td>7.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>3/16/2015</td>
<td>8.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>5/26/2015</td>
<td>9.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>9/14/2015</td>
<td>10.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>6/13/2016</td>
<td>11.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>9/14/2016</td>
<td>11.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/24/2018</td>
<td>12.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>10/1/2018</td>
<td>13.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>4/22/2021</td>
<td>14.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>8/17/2021</td>
<td>15.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>4/16/2024</td>
<td>16.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
</tbody>
</table>
Table of Contents

1 Introduction ................................................................................. 6
  1.1 Glossary ............................................................................. 6
  1.2 References .......................................................................... 7
    1.2.1 Normative References ..................................................... 7
    1.2.2 Informative References ..................................................... 8
  1.3 Overview ............................................................................... 8
  1.4 Relationship to Other Protocols .............................................. 8
  1.5 Prerequisites/Preconditions .................................................... 8
  1.6 Applicability Statement ........................................................ 9
  1.7 Versioning and Capability Negotiation .................................... 9
  1.8 Vendor-Extensible Fields ....................................................... 9
  1.9 Standards Assignments ......................................................... 9

2 Messages ................................................................................... 10
  2.1 Transport ............................................................................. 10
  2.2 Message Syntax .................................................................... 10
    2.2.1 RopGetPermissionsTable ROP .......................................... 10
      2.2.1.1 RopGetPermissionsTable ROP Request Buffer ............ 10
      2.2.1.2 RopGetPermissionsTable ROP Response Buffer ......... 10
    2.2.2 RopModifyPermissions ROP ............................................. 11
      2.2.2.1 RopModifyPermissions ROP Request Buffer ............. 11
      2.2.2.1.1 PermissionData Structure ................................ 11
      2.2.2.2 RopModifyPermissions ROP Response Buffer ........ 12
    2.2.3 PidTagAccessControlListData Property ......................... 12
    2.2.4 PidTagEntryId Property ................................................ 12
    2.2.5 PidTagMemberId Property .............................................. 12
    2.2.6 PidTagMemberName Property .......................................... 13
    2.2.7 PidTagMemberRights Property ......................................... 13

3 Protocol Details ....................................................................... 15
  3.1 Client Details ...................................................................... 15
    3.1.1 Abstract Data Model ...................................................... 15
    3.1.2 Timers ........................................................................... 15
    3.1.3 Initialization .................................................................. 15
    3.1.4 Higher-Layer Triggered Events ........................................ 15
      3.1.4.1 Retrieving Folder Permissions ................................. 15
      3.1.4.2 Adding Folder Permissions ....................................... 16
      3.1.4.3 Updating Folder Permissions ................................... 16
      3.1.4.4 Removing Folder Permissions .................................. 17
    3.1.5 Message Processing Events and Sequencing Rules ............ 17
    3.1.6 Timer Events ................................................................. 17
    3.1.7 Other Local Events ......................................................... 17
  3.2 Server Details ...................................................................... 17
    3.2.1 Abstract Data Model ...................................................... 17
    3.2.2 Timers ........................................................................... 17
    3.2.3 Initialization .................................................................. 17
    3.2.4 Higher-Layer Triggered Events ........................................ 17
      3.2.4.1 Accessing a Folder .................................................. 17
    3.2.5 Message Processing Events and Sequencing Rules .......... 18
      3.2.5.1 Processing a RopGetPermissionsTable ROP Request .... 18
      3.2.5.2 Processing a RopModifyPermissions ROP Request .... 18
      3.2.5.3 Processing a Request for PidTagSecurityDescriptorAsXml Property ...................................................... 18
    3.2.6 Timer Events ................................................................. 18
    3.2.7 Other Local Events ......................................................... 19
1 Introduction

The Exchange Access and Operation Permissions Protocol is used by clients to retrieve and manage the permissions on a folder. This protocol extends the Folder Object Protocol, described in [MS-OXCFOLD]. This protocol also extends the Availability Web Service Protocol, described in [MS-OXWAVLS], if both the client and the server support the Availability Web Service Protocol.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

- access control list (ACL): A list of access control entries (ACEs) that collectively describe the security rules for authorizing access to some resource; for example, an object or set of objects.
- Address Book object: An entity in an address book that contains a set of attributes, each attribute with a set of associated values.
- anonymous user: A user who presents no credentials when identifying himself or herself. The process for determining an anonymous user can differ based on the authentication protocol, and the documentation for the relevant authentication protocol should be consulted.
- binary large object (BLOB): A discrete packet of data that is stored in a database and is treated as a sequence of uninterpreted bytes.
- Calendar folder: A Folder object that contains Calendar objects.
- flags: A set of values used to configure or report options or settings.
- Folder object: A messaging construct that is typically used to organize data into a hierarchy of objects containing Message objects and folder associated information (FAI) Message objects.
- handle: Any token that can be used to identify and access an object such as a device, file, or a window.
- hierarchy table: A Table object whose rows represent the Folder objects that are contained in another Folder object.
- little-endian: Multiple-byte values that are byte-ordered with the least significant byte stored in the memory location with the lowest address.
- Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.
- permission: A rule that is associated with an object and that regulates which users can gain access to the object and in what manner. See also rights.
- permissions list: A list of users and the permissions for each of those users.
- property tag: A 32-bit value that contains a property type and a property ID. The low-order 16 bits represent the property type. The high-order 16 bits represent the property ID.
**remote operation (ROP):** An operation that is invoked against a server. Each ROP represents an action, such as delete, send, or query. A ROP is contained in a **ROP buffer** for transmission over the wire.

**remote procedure call (RPC):** A communication protocol used primarily between client and server. The term has three definitions that are often used interchangeably: a runtime environment providing for communication facilities between computers (the RPC runtime); a set of request-and-response message exchanges between computers (the RPC exchange); and the single message from an RPC exchange (the RPC message). For more information, see [C706].

**ROP buffer:** A structure containing an array of bytes that encode a **remote operation (ROP).** The first byte in the buffer identifies the ROP. This byte is followed by ROP-specific fields. Multiple ROP buffers can be packed into a single **remote procedure call (RPC) request or response.**

**ROP request:** See **ROP request buffer.**

**ROP request buffer:** A **ROP buffer** that a client sends to a server to be processed.

**ROP response buffer:** A **ROP buffer** that a server sends to a client to be processed.

**Server object handle:** A 32-bit value that identifies a Server object.

**Stream object:** A Server object that is used to read and write large string and binary properties.

**table object:** A group of shapes that are arranged in rows and columns to form a table.

**Unicode:** A character encoding standard developed by the Unicode Consortium that represents almost all of the written languages of the world. The **Unicode** standard [UNICODE5.0.0/2007] provides three forms (UTF-8, UTF-16, and UTF-32) and seven schemes (UTF-8, UTF-16, UTF-16 BE, UTF-16 LE, UTF-32, UTF-32 LE, and UTF-32 BE).

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

### 1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the **Errata.**

#### 1.2.1 Normative References

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

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

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


[MS-OXCTABL] Microsoft Corporation, "Table Object Protocol".

---

*Exchange Access and Operation Permissions Protocol*

_Copyright © 2024 Microsoft Corporation*

_Release: April 16, 2024*
1.3 Overview

The Exchange Access and Operation Permissions Protocol is used by a client to retrieve and manage the permissions list on a folder by using remote operations (ROPs). Each entry in this list specifies the permissions granted to a single user. The user's permissions determine what actions the user is allowed on the folder. For example, a user can be allowed to view a folder but not allowed to modify the folder's properties.

The permissions list initially contains two reserved entries: an entry that specifies folder permissions for an anonymous user and an entry that specifies the default permissions for a user who is not currently included in the permissions list. For information about how these reserved entries are used, see section 3.2.4.1. Additional entries are added by an owner of the folder. Existing entries can be modified or deleted.

This protocol extends the Folder Object Protocol, described in [MS-OXCFOLD]. This protocol also extends the Availability Web Service Protocol, described in [MS-OXWAVLS], if both the client and server support the Availability Web Service Protocol.

1.4 Relationship to Other Protocols

This protocol extends the Folder Object Protocol, described in [MS-OXCFOLD], by adding the ability to retrieve and manage the permissions list on a folder and, therefore, has the same dependencies as those described in [MS-OXCFOLD] section 1.4.

If the client and the server both implement the Availability Web Service Protocol, described in [MS-OXWAVLS], this protocol also extends that protocol by adding the ability to retrieve and manage the permissions list on the Calendar folder.

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

1.5 Prerequisites/Preconditions

In addition to the prerequisites of the Folder Object Protocol that are specified in [MS-OXCFOLD] section 1.5, the Exchange Access and Operation Permissions Protocol requires that the client be connected to the server by using credentials that belong to a user who has permissions to read and modify the folder's permissions list.
The client is required to obtain a handle to the Folder object on the server by using the RopOpenFolder ROP ([MS-OXCROPS] section 2.2.4.1). This handle will be included in the ROP buffers that are used in this protocol.

1.6 Applicability Statement

A client can use the Exchange Access and Operation Permissions Protocol to read or update the permissions list on a folder. For example, if the owner of a folder grants read permission on that folder to another user, the folder owner's client updates the permissions list on the folder accordingly.

1.7 Versioning and Capability Negotiation

The client checks the server's version number that is returned by the server in either the EcDoConnectEx method, as described in [MS-OXRPC], or the X-ServerApplication header of the Connect request type response, as described in [MS-OXCMAPIHTTP]. If the server version is greater than or equal to 8.0.360.0, the server supports the Availability Web Service Protocol, described in [MS-OXAVWS].

The client indicates to the server whether it supports the Availability Web Service Protocol by setting the IncludeFreeBusy flag in the ROP request buffer for both the RopGetPermissionsTable ROP ([MS-OXCROPS] section 2.2.10.2) and the RopModifyPermissions ROP ([MS-OXCROPS] section 2.2.10.1), as described in sections 2.2.1.1 and 2.2.2.1.

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 in this protocol are sent to and
received from the server respectively by using the underlying protocol specified by [MS-OXCROPS]  
section 2.1.

2.2 Message Syntax

Unless otherwise noted, sizes in this section are expressed in bytes.

Unless otherwise noted, the fields specified in this section are packed in buffers in the order that they 
appear in this document, without any padding.

Unless otherwise noted, the fields specified in this section, which are larger than a single byte, MUST 
be converted to little-endian order when packed in buffers and converted from little-endian order 
when unpacked.

2.2.1 RopGetPermissionsTable ROP

The RopGetPermissionsTable ROP ([MS-OXCROPS] section 2.2.10.2) retrieves a Server object 
handle to a Table object, which is then used in other ROP requests to retrieve the current 
permissions list on a folder.

The complete syntax of the ROP request buffer and the ROP response buffer is specified in [MS-
OXCROPS]. This section specifies the syntax and semantics of various fields that are not fully specified 
in [MS-OXCROPS].

2.2.1.1 RopGetPermissionsTable ROP Request Buffer

The following descriptions define valid fields for the RopGetPermissionsTable ROP request buffer 
([MS-OXCROPS] section 2.2.10.2.1).

TableFlags (1 byte): A set of flags that control how the server uses the values of the 
PidTagMemberRights property (section 2.2.7). The valid flags for this field are specified in the 
following table. The client MUST NOT set any other flags.

<table>
<thead>
<tr>
<th>Flag name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncludeFreeBusy</td>
<td>0x02</td>
<td>If this flag is set, the server MUST include the values of the FreeBusySimple and FreeBusyDetailed flags of the PidTagMemberRights property in the returned permissions list. If this flag is not set, the values of those flags in the returned permissions list are not valid and the client MUST ignore them. The client MUST NOT set this flag if the server version is less than 8.0.360.0, as specified in [MS-OXCRPC], or the folder is not the Calendar folder.</td>
</tr>
</tbody>
</table>

2.2.1.2 RopGetPermissionsTable ROP Response Buffer

The following descriptions define valid fields for the RopGetPermissionsTable ROP response 
buffer ([MS-OXCROPS] section 2.2.10.2.2).

ReturnValue (4 bytes): An integer that indicates the result of the operation. The value 0x00000000 
indicates success. For details about common error codes, see [MS-OXCDATA] section 2.4.
2.2.2 RopModifyPermissions ROP

The RopModifyPermissions ROP ([MS-OXCROPS] section 2.2.10.1) creates, updates, or deletes entries in the permissions list on a folder.

The complete syntax of the ROP request buffer and the ROP response buffer is specified in [MS-OXCROPS]. This section specifies the syntax and semantics of various fields that are not fully specified in [MS-OXCROPS].

2.2.2.1 RopModifyPermissions ROP Request Buffer

The following descriptions define valid fields for the RopModifyPermissions ROP request buffer ([MS-OXCROPS] section 2.2.10.1.1).

ModifyFlags (1 byte): A set of flags that control how the server uses the values of the PidTagMemberRights property (section 2.2.7) and the flags of the PermissionData structures. The valid flags for this field are specified in the following table. The client MUST NOT set any other flags.

<table>
<thead>
<tr>
<th>Flag name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReplaceRows</td>
<td>0x01</td>
<td>If this flag is set, the server MUST replace all existing entries except the default user entry in the current permissions list with the ones contained in the PermissionsData field. (In this case, each PermissionData structure in the PermissionsData field MUST have the AddRow flag set.) If this flag is not set, the server MUST add, update, or delete entries in the current permissions list according to the changes specified in the PermissionsData field. The client SHOULD set this flag when copying the access control list (ACL) from the Calendar folder to the Freebusy Data folder.</td>
</tr>
<tr>
<td>IncludeFreeBusy</td>
<td>0x02</td>
<td>If this flag is set, the server MUST apply the settings of the FreeBusySimple and FreeBusyDetailed flags of the PidTagMemberRights property when modifying the permissions of the Calendar folder. This flag MUST be set if the client sets either the FreeBusySimple flag or the FreeBusyDetailed flag. The client MUST NOT set this flag if the server version is less than 8.0.360.0, as specified in [MS-OXCRPC], or the folder is not the Calendar folder.</td>
</tr>
</tbody>
</table>

ModifyCount (2 bytes): An integer that specifies the number of structures contained in the PermissionsData field.

PermissionsData (variable): An array of PermissionData structures (section 2.2.2.1.1). Each PermissionData structure specifies details for adding a new entry to the permissions list, updating an existing entry in the permissions list, or deleting an entry from the permissions list. There is one PermissionData structure for each entry to be added, updated, or deleted.

If the ReplaceRows flag is set in the ModifyFlags field, entries can only be added. Therefore, each PermissionData structure contained in this field MUST have the AddRow flag set if the ReplaceRows flag is set.

2.2.2.1.1 PermissionData Structure

The PermissionData structure specifies the properties, including folder permissions, for a single user and the requested operation (add entry, update entry, delete entry) to be performed on the permissions list for that user.

PermissionDataFlags (1 byte): A set of flags that specify the type of change to be made to the folder permissions. The valid flags for this field are specified in the following table. The client MUST NOT set any other flags. If the ReplaceRows flag is set in the ModifyFlags field of the RopModifyPermissions ROP request buffer, only the AddRow flag is valid.
<table>
<thead>
<tr>
<th>Flag name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddRow</td>
<td>0x01</td>
<td>The user that is specified by the PidTagEntryId property (section 2.2.4) is added to the permissions list.</td>
</tr>
<tr>
<td>ModifyRow</td>
<td>0x02</td>
<td>The existing permissions for the user that is identified by the PidTagMemberId property (section 2.2.5) are modified.</td>
</tr>
<tr>
<td>RemoveRow</td>
<td>0x04</td>
<td>The user that is identified by the PidTagMemberId property is deleted from the permissions list.</td>
</tr>
</tbody>
</table>

**PropertyValueCount (2 bytes):** An integer that specifies the number of structures contained in the PropertyValues field.

**PropertyValues (variable):** An array of TaggedPropertyValue structures ([MS-OXCDATA] section 2.11.4). Each structure specifies one property. The properties included depend on the type of change that is being made. For details, see sections 3.1.4.2, 3.1.4.3, and 3.1.4.4.

### 2.2.2.2 RopModifyPermissions ROP Response Buffer

The following descriptions define valid fields for the RopModifyPermissions ROP response buffer ([MS-OXCROPS] section 2.2.10.1.2).

**ReturnValue (4 bytes):** An integer that indicates the result of the operation. The value 0x00000000 indicates success. For details about common error codes, see [MS-OXCDATA] section 2.4.

### 2.2.3 PidTagAccessControlListData Property

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

The PidTagAccessControlListData property ([MS-OXPROPS] section 2.506) contains a variable-length binary large object (BLOB) that constitutes a permissions list for a folder. This property is used when an ACL is copied from one folder to another. (<2>)

The client does not understand the data contained in the BLOB and does not try to use the data. The BLOB is only used to copy the ACL from one folder to another.

### 2.2.4 PidTagEntryId Property

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

The PidTagEntryId property ([MS-OXPROPS] section 2.684) identifies the Address Book object that specifies a user. The first two bytes of this property specify the number of bytes that follow. The remaining bytes constitute the PermanentEntryID structure ([MS-OXNSPI] section 2.2.9.3).

If the PidTagMemberId property (section 2.2.5) is set to one of the two reserved values, the first two bytes of this property MUST be 0x0000, indicating that zero bytes follow (that is, no PermanentEntryID structure follows the first two bytes).

### 2.2.5 PidTagMemberId Property

Type: PtypInteger64 ([MS-OXCDATA] section 2.11.1)

The PidTagMemberId property ([MS-OXPROPS] section 2.783) specifies the unique identifier that the server generates for each user.

The two reserved values for the PidTagMemberId property are listed in the following table.
### 2.2.6 PidTagMemberName Property

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

The `PidTagMemberName` property ([MS-OXPROPS] section 2.784) specifies the user-readable name of the user. The server provides the user-readable name for all entries in the permissions list.

The user-readable names that the server provides for the reserved entries of the permissions list are listed in the following table.

<table>
<thead>
<tr>
<th>Reserved entry</th>
<th>Value of the PidTagMemberId property</th>
<th>User-readable name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anonymous user</td>
<td>0xFFFFFFFFFFFFFFFF</td>
<td>&quot;Anonymous&quot;</td>
</tr>
<tr>
<td>Default user</td>
<td>0x0000000000000000</td>
<td>&quot;&quot; (empty string)</td>
</tr>
</tbody>
</table>

### 2.2.7 PidTagMemberRights Property

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

The `PidTagMemberRights` property ([MS-OXPROPS] section 2.785) specifies the folder permissions that are granted to the specified user.

The flags that are used to specify permissions are listed in the following table. The client and server MUST NOT set any other flags.

<table>
<thead>
<tr>
<th>Flag name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadAny</td>
<td>0x000000010</td>
<td>If this flag is set, the server MUST allow the specified user’s client to read any Message object in the folder. If this flag is not set, the server MUST NOT allow the user’s client to read Message objects that are owned by other users.</td>
</tr>
<tr>
<td>Create</td>
<td>0x000000020</td>
<td>If this flag is set, the server MUST allow the specified user’s client to create new Message objects in the folder. If this flag is not set, the server MUST NOT allow the user’s client to create new Message objects in the folder.</td>
</tr>
<tr>
<td>EditOwned</td>
<td>0x000000080</td>
<td>If this flag is set, the server MUST allow the specified user’s client to modify a Message object that was created by that user in the folder. If this flag is not set, the server MUST NOT allow the user’s client to modify Message objects that were created by that user. If the client sets the EditAny flag, the client MUST set this flag as well.</td>
</tr>
<tr>
<td>DeleteOwned</td>
<td>0x000000100</td>
<td>If this flag is set, the server MUST allow the specified user’s client to delete any Message object that was created by that user in the folder. If this flag is not set, the server MUST NOT allow the user’s client to delete Message objects that were created by that user. If the client sets the DeleteAny flag, the client MUST set this flag as well.</td>
</tr>
<tr>
<td>Flag name</td>
<td>Value</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>EditAny</td>
<td>0x00000020</td>
<td>If this flag is set, the server MUST allow the specified user’s client to modify any Message object in the folder. If this flag is not set, the server MUST NOT allow the user’s client to modify Message objects that are owned by other users.</td>
</tr>
<tr>
<td>DeleteAny</td>
<td>0x00000040</td>
<td>If this flag is set, the server MUST allow the specified user’s client to delete any Message object in the folder. If this flag is not set, the server MUST NOT allow the user’s client to delete Message objects that are owned by other users.</td>
</tr>
<tr>
<td>CreateSubFolder</td>
<td>0x00000080</td>
<td>If this flag is set, the server MUST allow the specified user’s client to create new folders within the folder. If this flag is not set, the server MUST NOT allow the user’s client to create new folders within the folder.</td>
</tr>
<tr>
<td>FolderOwner</td>
<td>0x00000100</td>
<td>If this flag is set, the server SHOULD allow the specified user’s client to modify properties, including the folder permissions, that are set on the folder itself. If this flag is not set, the server SHOULD NOT allow the specified user’s client to make those modifications.</td>
</tr>
<tr>
<td>FolderContact</td>
<td>0x00000200</td>
<td>This flag has no meaning to the server regarding folder permissions. The client uses this flag when displaying permissions. If neither this flag nor the FolderOwner flag is set, the specified user’s client does not display the permissions list for the folder. Instead, the specified user’s client displays the folder permissions specified in the PidTagRights property ([MS-OXCFOLD] section 2.2.2.2.2.8), which contains the folder permissions only for that user.</td>
</tr>
<tr>
<td>FolderVisible</td>
<td>0x00000400</td>
<td>If this flag is set, the server MUST allow the specified user’s client to retrieve the folder's permissions list, as specified in section 3.1.4.1, to see the folder in the folder hierarchy table, or to open the folder. If the client sets the ReadAny flag or the FolderOwner flag, the client MUST set this flag as well.</td>
</tr>
<tr>
<td>FreeBusySimple</td>
<td>0x00000800</td>
<td>If this flag is set, the server MUST allow the specified user’s client to retrieve brief information about the appointments on the calendar through the Availability Web Service Protocol, as specified in [MS-OXWAVLS]. If this flag is not set, the server MUST NOT allow the specified user’s client to retrieve information through the Availability Web Service Protocol. If the client sets the FreeBusyDetailed flag, it MUST set this flag as well.</td>
</tr>
<tr>
<td>FreeBusyDetailed</td>
<td>0x00001000</td>
<td>If this flag is set, the server MUST allow the specified user’s client to retrieve detailed information about the appointments on the calendar through the Availability Web Service Protocol, as specified in [MS-OXWAVLS]. If this flag is not set, the server MUST NOT allow the specified user’s client to see these details.</td>
</tr>
</tbody>
</table>
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 abstract data model for the client is the same as that specified in [MS-OXCFOLD].

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Retrieving Folder Permissions

Before retrieving the permissions list of a folder, the client attempts to retrieve the folder permissions by reading the folder's PidTagSecurityDescriptorAsXml property ([MS-XWDVSEC] section 2.2.2). To read this property, the client opens the property as a Stream object by sending a RopOpenStream ROP request ([MS-OXCROPS] section 2.2.9.1). The server MUST return an error code of ecNotImplemented instead of satisfying the RopOpenStream ROP request.

To retrieve the current permissions list of a folder, the client SHOULD send the following three ROP requests to the server:

1. **RopGetPermissionsTable** ([MS-OXCROPS] section 2.2.10.2)

2. **RopSetColumns** ([MS-OXCROPS] section 2.2.5.1), with a column set that includes some or all of the following properties:
   - **PidTagEntryId** (section 2.2.4) — If the client is not required to match entries in the permissions list to users, as it would with a search for a particular user, the client SHOULD NOT include this property.
   - **PidTagMemberId** (section 2.2.5) — The client MUST include this property.
   - **PidTagMemberName** (section 2.2.6) — If the client is not displaying the contents of the permissions list, the client SHOULD NOT include this property.
   - **PidTagMemberRights** (section 2.2.7) — The client MUST include this property.

3. **RopQueryRows** ([MS-OXCROPS] section 2.2.5.4)

For more details about how the client uses the RopSetColumns and RopQueryRows ROP requests, see [MS-OXCTABL] section 3.1.4. If all three of the ROP requests succeed, the permissions list is returned in the RowData field of the RopQueryRows ROP response buffer. The RowData field
contains one PropertyRow structure ([MS-OXCDATA] section 2.8.1) for each entry in the permissions list. When the client is finished with table operations, the client MUST release the Table object by sending a RopRelease ROP request ([MS-OXCROPS] section 2.2.15.3).

The ROP sequence that is used to retrieve the current permissions list of a folder is shown in the following diagram.

![Diagram](image)

Figure 1: Sequence for retrieving folder permissions

### 3.1.4.2 Adding Folder Permissions

To add a new entry to the folder's permissions list, the client MUST send a RopModifyPermissions ROP request ([MS-OXCROPS] section 2.2.10.1) with the AddRow flag set in each PermissionData structure that specifies a user to be added. The PropertyValues field of the PermissionData structure MUST include the following properties:

- PidTagEntryId (section 2.2.4)
- PidTagMemberRights (section 2.2.7)

The PropertyValues field MUST NOT include the PidTagMemberId property (section 2.2.5).

When copying the ACL from the Calendar folder to the Freebusy Data folder, the client SHOULD set the ReplaceRows flag in the RopModifyPermissions ROP request buffer. In this case, all of the PermissionData structures in the ROP request buffer MUST have the AddRow flag set.

### 3.1.4.3 Updating Folder Permissions

To update an existing entry in the folder's permissions list, the client MUST retrieve the existing permissions list as specified in section 3.1.4.1 to get the values of the PidTagMemberId properties that are assigned to the users in the permissions list.

The client MUST send a RopModifyPermissions ROP request ([MS-OXCROPS] section 2.2.10.1) with the ModifyRow flag set in each PermissionData structure that specifies a user to be modified. The PropertyValues field of the PermissionData structure MUST include the following properties:

- PidTagMemberId (section 2.2.5)
- PidTagMemberRights (section 2.2.7)
The PropertyValues field MUST NOT include the PidTagEntryId property (section 2.2.4).

3.1.4.4 Removing Folder Permissions

To remove an entry from the folder's permissions list, the client MUST retrieve the existing permissions list as specified in section 3.1.4.1 to get the values of the PidTagMemberId properties that are assigned to the users in the permissions list.

The client MUST send a RopModifyPermissions ROP request ([MS-OXCROPS] section 2.2.10.1) with the RemoveRow flag set in each PermissionData structure that specifies a user to be deleted. The PropertyValues field of the PermissionData structure MUST include only the PidTagMemberId property (section 2.2.5).

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

3.2.1 Abstract Data Model

The abstract data model for the client and server roles is the same.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

3.2.4.1 Accessing a Folder

When a client sends a request to the server to access a folder, as specified in [MS-OXCFOLD], the server MUST either allow or deny the request based on the permissions list for the folder and the user credentials that the client provided when making the request.

The server determines whether the user identified by the user credentials is included in the permissions list and then applies the folder permissions for that user as follows:

- Specific user permissions: If the user is included in the permissions list, either explicitly or through membership in a group that is included in the permissions list, the server MUST apply the permissions that have been set for that user.
- Default user permissions: If the user is not included in the permissions list, the server MUST apply the permissions that have been set in the default user entry of the permissions list.

- Anonymous user permissions: Permissions for an anonymous user are not supported and the anonymous user entry is not used. Consequently, the server requires that the client provide user credentials.

### 3.2.5 Message Processing Events and Sequencing Rules

#### 3.2.5.1 Processing a RopGetPermissionsTable ROP Request

When the server receives a RopGetPermissionsTable ROP request buffer ([MS-OXCROPS] section 2.2.10.2) from the client, the server parses the buffer. The server responds with a RopGetPermissionsTable ROP response buffer. For details about how the server parses buffers and processes ROPs, see [MS-OXCROPS] section 3.2.5.1. For details about how the server formats buffers for the response, see [MS-OXCROPS] section 3.2.5.2.

The server MUST return a Server object handle to a Table object, which the client uses to retrieve the permissions list of the folder, as specified in section 3.1.4.1. The server MUST determine whether the user has permission to view the permissions list of the folder by examining the FolderVisible flag of the PidTagMemberRights property (section 2.2.7). If the user has permission to view the permissions list of the folder, the server returns the permissions list in a RopQueryRows ROP response buffer ([MS-OXCROPS] section 2.2.5.4). If the user does not have permission to view the permissions list of the folder, the server returns 0x80070005 (AccessDenied) in the ReturnValue field of the RopQueryRows ROP response buffer.

#### 3.2.5.2 Processing a RopModifyPermissions ROP Request

When the server receives a RopModifyPermissions ROP request buffer ([MS-OXCROPS] section 2.2.10.1) from the client, the server parses the buffer. The server responds with a RopModifyPermissions ROP response buffer. For details about how the server parses buffers and processes ROPs, see [MS-OXCROPS] section 3.2.5.1. For details about how the server formats buffers for the response, see [MS-OXCROPS] section 3.2.5.2.

The server MUST determine whether the user has permission to modify the properties of the folder by examining the FolderOwner flag of the PidTagMemberRights property (section 2.2.7). If the user does not have permission to modify the folder’s properties, the server MUST return the AccessDenied (0x80070005) error code in the ReturnValue field of the ROP response buffer. If the user does have permission to modify the folder’s properties, the server MUST update the permissions list for the folder according to the PermissionData structures listed in the PermissionsData field of the ROP request buffer, as specified in section 2.2.2.1.

If any PermissionData structure specifies deletion or modification of a user that is not currently in the permissions list, the server MUST ignore that PermissionData structure.

#### 3.2.5.3 Processing a Request for PidTagSecurityDescriptorAsXml Property

When the server receives a RopOpenStream ROP request ([MS-OXCROPS] section 2.2.9.1) on the PidTagSecurityDescriptorAsXml property ([MS-XWDVSEC] section 2.2.2) of the folder, the server MUST return an error code of ecNotImplemented rather than satisfying the RopOpenStream ROP request.

#### 3.2.6 Timer Events

None.
3.2.7 Other Local Events

None.
4 Protocol Examples

4.1 Adding an Entry to the Permissions List

In this example, the client is adding an entry for "user8" to the permissions list on the Calendar folder. To retrieve the current permissions on the folder, the client starts by trying to read the deprecated PidTagSecurityDescriptorAsXml property ([MS-XWDVSEC] section 2.2.2) of the folder, as described in section 3.1.4.1. To read this property, the client sends the following RopOpenStream ROP request ([MS-OXCROPS] section 2.2.9.1).

The RopOpenStream ROP request buffer contains the following data (9 bytes).

```
0000: 2B 00 01 02 1F 00 6A 0E 00
```

- **RopId**: 0x2B
- **LogonId**: 0
- **InputHandleIndex**: 1 (HSOT=0x000001DA)
- **OutputHandleIndex**: 2 (HSOT=0xFFFFFFFF)
- **PropertyTag**: 0x0E6A001F (PidTagSecurityDescriptorAsXml property)
- **OpenModeFlags**: 0x00 (ReadOnly flag is set)

The server returns the following ROP response buffer, which indicates that it does not support the PidTagSecurityDescriptorAsXml property on this folder.

The RopOpenStream ROP response buffer contains the following data (6 bytes):

```
0000: 2B 02 02 01 04 80
```

- **RopId**: 0x2B
- **OutputHandleIndex**: 2 (HSOT=0xFFFFFFFF)
- **ReturnValue**: 0x80040102 (ecNotImplemented)

Because the server does not support the PidTagSecurityDescriptorAsXml property, the client falls back to using the RopGetPermissionsTable ROP ([MS-OXCROPS] section 2.2.10.2) as described in section 3.1.4.1. To retrieve a table that contains the current permissions list of the folder, the client sends the following three ROP requests, batched together into a single remote procedure call (RPC).

The RopGetPermissionsTable ROP request buffer contains the following data (5 bytes):

```
0000: 3E 00 00 01 02
```

- **RopId**: 0x3E
- **LogonId**: 0
- **InputHandleIndex**: 0 (HSOT=0x000001DA)
- **OutputHandleIndex**: 1 (HSOT=0xFFFFFFFF)
TableFlags: 0x02 (IncludeFreeBusy flag is set)

The RopSetColumns ROP request buffer ([MS-OXCROPS] section 2.2.5.1) contains the following data (22 bytes):

```
0000: 12 00 01 00 04 00 14 00-71 66 1F 00 72 66 03 00 ........qf..rf..
0010: 73 66 02 01 FF 0F ........sf...
```

RopId: 0x12
LogonId: 0
InputHandleIndex: 1 (HSOT=0xFFFFFFFF)
SetColumnsFlags: 0x00 (Wait flag is set)

PropertyTagCount: 0x0004 (four property tags in the PropertyTags field)

PropertyTags:
* 0x66710014 (PidTagMemberId property (section 2.2.5))
* 0x6672001F (PidTagMemberName property (section 2.2.6))
* 0x66730003 (PidTagMemberRights property (section 2.2.7))
* 0xFFF0102 (PidTagEntryId property (section 2.2.4))

The RopQueryRows ROP request buffer ([MS-OXCROPS] section 2.2.5.4) contains the following data (7 bytes):

```
0000: 15 00 01 00 01 00 10
```

RopId: 0x15
LogonId: 0
InputHandleIndex: 1 (HSOT=0xFFFFFFFF)
QueryRowsFlags: 0x00 (Advance)
ForwardRead: 0x01 (True)
RowCount: 0x1000 (4096)

The server returns the following three ROP response buffers. The folder's current permissions list is in the RowData field of the RopQueryRows ROP response buffer.

The RopGetPermissionsTable ROP response buffer contains the following data (6 bytes):

```
0000: 3E 01 00 00 00 00
```

RopId: 0x3E
OutputHandleIndex: 1 (HSOT=0x000000CA)
ReturnValue: 0x00000000 (success)
The **RopSetColumns** ROP response buffer contains the following data (7 bytes):

```plaintext
0000: 12 01 00 00 00 00 00
```

**RopId**: 0x12  
**InputHandleIndex**: 1 (HSOT=0x000000CA)  
**ReturnValue**: 0x00000000 (success)  
**TableStatus**: 0x00 (**TBLSTAT_COMPLETE** flag is set)

The **RopQueryRows** ROP response buffer contains the following data (61 bytes):

```plaintext
0000: 15 01 00 00 00 00 02 02-00 00 00 00 00 00 00 00 00 ............... 0010: 00 00 00 00 00 FF FF FF FF FF FF FF FF FF FF FF FF FF FF ............... 0020: FF FF FF FF 41 00 6E 00 6F-00 6E 00 79 00 6D 00 6F ..A.n.o.n.y.m.o 0030: 00 75 00 73 00 00 00 00 ...................  .u.s.........
```

**RopId**: 0x15  
**InputHandleIndex**: 1 (HSOT=0x000000CA)  
**ReturnValue**: 0x00000000 (success)  
**Origin**: 0x02 (**BOOKMARK_END** flag is set)  
**RowCount**: 0x0002 (two **PropertyRow** structures in the **RowData** field)

**RowData**:  
**PropertyRow** structure #1 (beginning at address 0x0009 in the **RopQueryRows** ROP response buffer):

**Flag**: 0x00 (no errors)  
**ValueArray**: This field contains the values of the properties that were specified in the **PropertyTags** field of the previous **RopSetColumns** request and are in the same order as those properties.

0x0000000000000000 (default user)  
0x00000 (Unicode null)  
0x00000800 (**FreeBusySimple** flag is set)  
0x0000 (byte count is zero bytes, indicating that no **PermanentEntryID** structure ([MS-OXNSPI] section 2.2.9.3) is present)

**PropertyRow** structure #2 (beginning at address 0x001A in the **RopQueryRows** ROP response buffer):

**Flag**: 0x00 (no errors)  
**ValueArray**:  
0xFFFFFFFFFFFFFFFF (anonymous user)  
"Anonymous"  
0x00000000 (no permissions flags are set)
0x0000 (byte count is zero bytes, indicating that no PermanentEntryID structure is present)

Note that the current permissions list on this folder has two entries. The default user entry, contained
in PropertyRow structure #1, has the FreeBusySimple permissions (0x00000800) on this folder. The anonymous user entry, contained in PropertyRow structure #2, has no permissions (0x00000000) on this folder.

Finally, the client sends the following RopModifyPermissions ROP request ([MS-OXCROPS] section 2.2.10.1) to add "user8" to the permissions list with the FreeBusyDetailed, FreeBusySimple, FolderVisible, FolderContact, FolderOwner, CreateSubFolder, DeleteAny, EditAny, DeleteOwned, EditOwned, Create, and ReadAny permissions flags set (0x00001FFB) for "user8" on this folder:

The RopModifyPermissions ROP request buffer contains the following data (147 bytes):

```
0000: 40 00 02 02 01 00 01 02-00 02 01 FF 0F 7C 00 00 @............|..
0010: 00 00 00 DC A7 40 C8 C0-42 10 1A B4 B9 08 00 2B ......@..B......+
0020: 2F E1 82 01 00 00 00 00-00 00 02 6F 3D 46 69 /........../o-Fi
0030: 72 73 74 20 4F 72 67 61-6E 69 7A 61 74 69 6F 6E rst Organization
0040: 2F 6F 75 3D 45 78 63 68-61 6E 67 65 20 41 64 6D /ou=Exchange Adm
0050: 69 6E 69 73 72 61 74 69-76 65 20 47 72 6F 75 inistrative Group
0060: 70 20 28 46 59 44 49 42-p (FYDIBOHF23SPD
0070: 4C 54 29 2F 63 6C 3D 52 LT)/cn=Recipient
0080: 73 2F 63 6C 3D 75 73 65-s/cn=user8...sf.
0090: 1F 00 00                                        ...
```

RopId: 0x40
LogonId: 0
InputHandleIndex: 2 (HSOT=0x000001DA)
ModifyFlags: 0x02 (IncludeFreeBusy flag is set)
ModifyCount: 0x0001 (one PermissionData structure in the PermissionsData field)

PermissionsData:

PermissionData structure (beginning at address 0x0006 in the RopModifyPermissions ROP request buffer):

PermissionDataFlags: 0x01 (AddRow flag is set)

PropertyValueCount: 0x0002 (two TaggedPropertyValue structures in the PropertyValues field)

PropertyValues:

TaggedPropertyValue structure #1 (beginning at address 0x0009 in the RopModifyPermissions ROP request buffer):

PropertyTag: 0x0FFFF0102 (PidTagEntryId property)

PropertyValue: Beginning at address 0x000D in the RopModifyPermissions ROP request buffer, this field contains the byte-count, 0x007C (124), followed by the 124-byte PermanentEntryID structure.

TaggedPropertyValue structure #2 (beginning at address 0x008B in the RopModifyPermissions ROP request buffer):

PropertyTag: 0x66730003 (PidTagMemberRights property)
4.2 Modifying an Entry in the Permissions List

In this example, the client is modifying the entry for "user8" in the permissions list on the Calendar folder. First, the client retrieves the permissions list by sending the same RopGetPermissionsTable ([MS-OXCROPS] section 2.2.10.2), RopSetColumns ([MS-OXCROPS] section 2.2.5.1), and RopQueryRows ([MS-OXCROPS] section 2.2.5.4) ROP requests as in the example in section 4.1.

The server returns the following three ROP response buffers with the current permissions list contained in the RowData field of the RopQueryRows ROP response buffer.

The RopGetPermissionsTable ROP response buffer contains the following data (6 bytes):


RopId: 0x3E
OutputHandleIndex: 1 (HSOT=0x000000CA)
ReturnValue: 0x00000000 (success)

The RopSetColumns ROP response buffer contains the following data (7 bytes):


RopId: 0x12
InputHandleIndex: 1 (HSOT=0x000000CA)
ReturnValue: 0x00000000 (success)
TableStatus: 0x00 (TBLSTAT_COMPLETE flag is set)

The RopQueryRows ROP response buffer contains the following data (212 bytes):


RopId: 0x15
InputHandleIndex: 1 (HSOT=0x000000CA)
ReturnValue: 0x00000000 (success)
Origin: 0x02 (BOOKMARK_END flag is set)
RowCount: 0x0003 (three PropertyRow structures in the RowData field)

RowData:

PropertyRow structure #1 (beginning at address 0x0009 in the RopQueryRows ROP response buffer):

Flag: 0x00 (no errors)
ValueArray:
0x0000000000000000 (default user)
0x0000000000000000 (Unicode null)
0x000000800 (FreeBusySimple flag is set)
0x0000000000000000 (byte count is zero bytes, indicating that no PermanentEntryID structure ([MS-OXNSPI] section 2.2.9.3) is present)

PropertyRow structure #2 (beginning at address 0x001A in the RopQueryRows ROP response buffer):

Flag: 0x00 (no errors)
ValueArray:
0x0000000000000002 "user8"
0x00000001FB (FreeBusyDetailed, FreeBusySimple, FolderVisible, FolderContact, FolderOwner, CreateSubFolder, DeleteAny, EditAny, DeleteOwned, EditOwned, Create, and ReadAny flags are set)

The last value in the ValueArray field is the value of the PidTagEntryId property (section 2.2.4). The value begins with the byte count, 0x007C (124), at address 0x0033 in the RopQueryRows ROP response buffer, followed by the 124-byte PermanentEntryID structure.

PropertyRow structure #3 (beginning at address 0x00B1 in the RopQueryRows ROP response buffer):

Flag: 0x00 (no errors)
ValueArray:
The permissions list on this folder now has an entry for "user8", which the client added as shown in the example in section 4.1. The client changes the permissions for "user8" from 0x0000FFFB to 0x00001800 (FreeBusyDetailed and FreeBusySimple flags) by sending the following \texttt{RopModifyPermissions} request ([MS-OXCROPS] section 2.2.10.1) with the \texttt{PermissionsData} field containing the entry to be modified.

The \texttt{RopModifyPermissions ROP request buffer} contains the following data (29 bytes).

\begin{verbatim}
  0000: 40 00 00 02 01 00 02 02-00 14 00 71 66 02 00 00 0.........qf....
  0010: 00 15 00 00 00 03 00 73-66 00 18 00 00          .......sf....
\end{verbatim}

\textbf{RopId}: 0x40

\textbf{LogonId}: 0

\textbf{InputHandleIndex}: 0 (HSOT=0x000001DA)

\textbf{ModifyFlags}: 0x02 (IncludeFreeBusy flag is set)

\textbf{ModifyCount}: 0x0001 (one PermissionData structure in the PermissionsData field)

\textbf{PermissionsData}:

\texttt{PermissionData} structure (beginning at address 0x0006 in the \texttt{RopModifyPermissions} ROP request buffer):

\textbf{PermissionDataFlags}: 0x02 (ModifyRow flag is set)

\textbf{PropertyValueCount}: 0x0002 (two TaggedPropertyValue structures in the PropertyValues field)

\textbf{PropertyValues}:

\textbf{TaggedPropertyValue} structure #1 (beginning at address 0x0009 in the \texttt{RopModifyPermissions} ROP request buffer):

\textbf{PropertyTag}: 0x66710014 (PidTagMemberId property (section 2.2.5))

\textbf{PropertyValue}: 0x0000001500000002

\textbf{TaggedPropertyValue} structure #2 (beginning at address 0x0015 in the \texttt{RopModifyPermissions} ROP request buffer):

\textbf{PropertyTag}: 0x66730003 (PidTagMemberRights property (section 2.2.7))

\textbf{PropertyValue}: 0x00001800 (FreeBusyDetailed and FreeBusySimple flags are set)

The server returns the following ROP response buffer, which indicates that it successfully updated the permissions list for the folder.

The \texttt{RopModifyPermissions ROP response buffer} contains the following data (6 bytes).

\begin{verbatim}
  0000: 40 00 00 00 00 00
\end{verbatim}
4.3 Removing an Entry from the Permissions List

In this example, the client is removing the entry for "user8" from the permissions list on the Calendar folder. First, the client retrieves the permissions list by sending the same RopGetPermissionsTable ([MS-OXCROPS] section 2.2.10.2), RopSetColumns [MS-OXCROPS] section 2.2.5.1, and RopQueryRows ([MS-OXCROPS] section 2.2.5.4) ROP requests as in the example in section 4.1. The server returns the following three ROP response buffers with the current permissions list contained in the RowData field of the RopQueryRows ROP response buffer.

The RopGetPermissionsTable ROP response buffer contains the following data (6 bytes).

```
0000: 3E 01 00 00 00 00
```

RopId: 0x3E

OutputHandleIndex: 1 (HSOT=0x000000CA)

ReturnValue: 0x00000000 (success)

The RopSetColumns ROP response buffer contains the following data (7 bytes).

```
0000: 12 01 00 00 00 00 00
```

RopId: 0x12

InputHandleIndex: 1 (HSOT=0x000000CA)

ReturnValue: 0x00000000 (success)

TableStatus: 0x00 (TBLSTAT_COMPLETE flag is set)

The RopQueryRows ROP response buffer contains the following data (212 bytes).

```
0000: 15 01 00 00 00 00 02 03-00 00 00 00 00 00 00 00 ................
0010: 00 00 00 00 00 00 00 00 00 00 02 00 00 00 00 15 ................
0020: 00 00 00 75 00 73 00 65 00 65-00 72 00 38 00 00 00 00 ...u.s.e.r.8....
0030: 18 00 00 7C 00 00 00 00 00 DC A7 40 C8 C0 42 10 ...|.......@..B.
0040: 00 00 2F 4F 3D 46 49 52 53 54 20 4F 52 47 41 4E 49 5A 41 54 49 4F 4E 54 53 20 4F 55 3D 45 58 43 48 41 4E 20 47 52 4F 55 20 28 46 59 44 49 42 4F 48 46 32 33 53 50 44 4L 54 29 2F 43 4E 3D 55 53 45 52 38 41 6E 00 6F 00 6N 00 79 00 6D 00 6F 00 75 00 73 00 00 00 00 00 00 n.y.m.o.u.s.....
```

RopId: 0x15

InputHandleIndex: 1 (HSOT=0x000000CA)
ReturnValue: 0x00000000 (success)
Origin: 0x02 (BOOKMARK_END flag is set)
RowCount: 0x0003 (three PropertyRow structures in the RowData field)
RowData:

PropertyRow structure #1 (beginning at address 0x0009 in the RopQueryRows ROP response buffer):
Flag: 0x00 (no errors)
ValueArray:
0x0000000000000000 (default user)
0x0000 (Unicode null)
0x00000800 (FreeBusySimple flag is set)
0x0000 (byte count is zero bytes, indicating that no PermanentEntryID structure ([MS-OXNSPI] section 2.2.9.3) is present)

PropertyRow structure #2 (beginning at address 0x001A in the RopQueryRows ROP response buffer):
Flag: 0x00 (no errors)
ValueArray:
0x00001500000002
"user8"
0x00001800 (FreeBusyDetailed and FreeBusySimple flags are set)
The last value in the ValueArray field is the value of the PidTagEntryId property (section 2.2.4). The value begins with the byte count, 0x007C (124), at address 0x0033 in the RopQueryRows ROP response buffer, followed by the 124-byte PermanentEntryID structure.

PropertyRow structure #3 (beginning at address 0x00B1 in the RopQueryRows ROP response buffer):
Flag: 0x00 (no errors)
ValueArray:
0xFFFFFFFFFFFFFFFF (anonymous user)
"Anonymous"
0x00000000 (no permissions flags are set)
0x0000 (byte count is zero bytes, indicating that no PermanentEntryID structure is present)

The permissions list on this folder now has an entry for "user8", which the client added as shown in the example in section 4.1 and modified as shown in the example in section 4.2. The client removes the permissions for "user8" from the permissions list by sending the following RopModifyPermissions ROP request ([MS-OXCROPS] section 2.2.10.1) with the PermissionsData field containing the entry to be removed.
The **RopModifyPermissions ROP request buffer** contains the following data (21 bytes).

```plaintext
0000: 40 00 00 02 01 00 04 01=00 14 00 71 66 02 00 00 @.........qf...
0010: 00 15 00 00 00
```

- **RopId**: 0x40
- **LogonId**: 0
- **InputHandleIndex**: 0 (HSOT=0x000001DA)
- **ModifyFlags**: 0x02 (**IncludeFreeBusy** flag is set)
- **ModifyCount**: 0x0001 (one **PermissionData** structure in the **PermissionsData** field)
- **PermissionsData**:
  - **PermissionData** structure (beginning at address 0x0006 in the RopModifyPermissions ROP request buffer):
    - **PermissionDataFlags**: 0x04 (**RemoveRow** flag is set)
  - **PropertyValueCount**: 0x0001 (one **TaggedPropertyValue** structure in the **PropertyValues** field)
- **PropertyValues**:
  - **TaggedPropertyValue** structure #1 (beginning at address 0x0009 in the RopModifyPermissions ROP request buffer):
    - **PropertyTag**: 0x66710014 (**PidTagMemberId** property (section 2.2.5))
    - **PropertyValue**: 0x0000001500000002

The server returns the following ROP response buffer, indicating that it has successfully updated the permissions list for the folder.

The **RopModifyPermissions** ROP response buffer contains the following data (6 bytes).

```plaintext
0000: 40 00 00 00 00 00
```

- **RopId**: 0x40
- **InputHandleIndex**: 0 (HSOT=0x000001DA)
- **ReturnValue**: 0x00000000 (success)
5 Security

5.1 Security Considerations for Implementers

Implementers of this protocol have to manage the folder permissions specified by the FolderVisible, FolderContact, and FolderOwner flags properly. General security considerations pertaining to the underlying ROP-based transport apply.

5.2 Index of Security Parameters

None.
Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

- Microsoft Exchange Server 2003
- Microsoft Exchange Server 2007
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Microsoft Exchange Server 2019
- Microsoft Office Outlook 2003
- Microsoft Office Outlook 2007
- Microsoft Outlook 2010
- Microsoft Outlook 2013
- Microsoft Outlook 2016
- Microsoft Outlook 2019
- Microsoft Outlook 2021
- Microsoft Outlook 2024 Preview

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

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

<1> Section 2.2.2.1: Office Outlook 2003 and Office Outlook 2007 do not set the ReplaceRows flag in the RopModifyPermissions ROP request buffer ([MS-OXCROPS] section 2.2.10.1.1).

<2> Section 2.2.3: Outlook 2010, Outlook 2013, Outlook 2016, Outlook 2019 do not use the PidTagAccessControlListData property (section 2.2.3).

<3> Section 2.2.7: Exchange 2007 does not allow the properties of a folder to be modified when the FolderOwner flag is set.

<4> Section 2.2.7: Exchange 2007 and Exchange 2010 allow the properties of a folder to be modified when the FolderOwner flag is not set.

<5> Section 2.2.7: Exchange 2007, Exchange 2010, Exchange 2013, Exchange 2016, and Exchange 2019 include the FreeBusySimple flag by default on the Calendar folder for any entry in the permissions list except the anonymous user and add the FreeBusyDetailed flag to any entries that have the ReadAny flag set. Exchange 2007, Exchange 2010, Exchange 2013, Exchange 2016, and Exchange 2019 use these defaults until the client modifies the permissions list with the
IncludeFreeBusy flag set in the RopModifyPermissions ROP request to override the default value.

<6> Section 3.1.4.2: Office Outlook 2003 and Office Outlook 2007 do not set the ReplaceRows flag in the RopModifyPermissions ROP request buffer ([MS-OXCROPS] section 2.2.10.1.1).
7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

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.
- A document revision that captures changes to protocol functionality.

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 None means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

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

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Revision class</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 Appendix A: Product Behavior</td>
<td>Updated list of supported products.</td>
<td>Major</td>
</tr>
</tbody>
</table>
Relationship to other protocols 8
Removing an entry in the permissions list example 27

RopGetPermissionsTable ROP
overview 10
request buffer 10
response buffer 10
RopGetPermissionsTable ROP message 10

RopModifyPermissions ROP
overview 11
request buffer 11
response buffer 12
RopModifyPermissions ROP message 11

S

Security
implementer considerations 30
parameter index 30
Sequencing rules
client 17
Sequencing rules - server
processing a request
PidTagSecurityDescriptorAsXml property 18
processing a RopGetPermissionsTable ROP request 18
processing a RopModifyPermissions ROP request 18
Server
abstract data model 17
initialization 17
other local events 19
timer events 18
timers 17
Server - higher-layer triggered events
accessing a folder 17
Server - message processing
processing a request for
PidTagSecurityDescriptorAsXml property 18
processing a RopGetPermissionsTable ROP request 18
processing a RopModifyPermissions ROP request 18
Server - sequencing rules
processing a request for
PidTagSecurityDescriptorAsXml property 18
processing a RopGetPermissionsTable ROP request 18
processing a RopModifyPermissions ROP request 18
Standards assignments 9
Syntax 10

T

Timer events
client 17
server 18
Timers
client 15
server 17
Tracking changes 33
Transport 10
Triggered events - client