

[MS-OXCMAPIHTTP]: Messaging Application Programming Interface (MAPI) Extensions for HTTP

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft [Open Specification Promise](#) or the [Community Promise](#). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email 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
11/18/2013	1.0	New	Released new document.
02/10/2014	2.0	Major	Significantly changed the technical content.
04/30/2014	2.1	Minor	Clarified the meaning of the technical content.
07/31/2014	3.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction	7
1.1 Glossary	7
1.2 References	8
1.2.1 Normative References	8
1.2.2 Informative References	8
1.3 Overview	9
1.4 Relationship to Other Protocols	10
1.5 Prerequisites/Preconditions	11
1.6 Applicability Statement	11
1.7 Versioning and Capability Negotiation	11
1.8 Vendor-Extensible Fields	11
1.9 Standards Assignments	11
2 Messages	12
2.1 Transport	12
2.2 Message Syntax	12
2.2.1 Common Data Types	12
2.2.1.1 AddressBookPropertyValueList Structure	12
2.2.1.2 AddressBookPropertyRow Structure	12
2.2.1.3 LargePropertyTagArray Structure	13
2.2.2 POST Method	13
2.2.2.1 Common Request Format	14
2.2.2.2 Common Response Format	14
2.2.3 Header Fields	15
2.2.3.1 Host Header Field	15
2.2.3.2 Entity Header Fields	15
2.2.3.2.1 Content-Length Header Field	15
2.2.3.2.2 Content-Type Header Field	16
2.2.3.2.3 Set-Cookie Header Field	16
2.2.3.2.4 Cookie Header Field	16
2.2.3.2.5 Transfer-Encoding Header Field	16
2.2.3.3 X-Header Fields	16
2.2.3.3.1 X-RequestType Header Field	16
2.2.3.3.2 X-RequestId Header Field	18
2.2.3.3.3 X-ResponseCode Header Field	18
2.2.3.3.4 X-ClientInfo Header Field	19
2.2.3.3.5 X-PendingPeriod Header Field	19
2.2.3.3.6 X-ClientApplication Header Field	19
2.2.3.3.7 X-ServerApplication Header Field	19
2.2.3.3.8 X-ExpirationInfo Header Field	19
2.2.3.3.9 X-ElapsedTime Header	19
2.2.3.3.10 X-StartTime Header	19
2.2.4 Request Types for Mailbox Server Endpoint	20
2.2.4.1 Connect Request Type	20
2.2.4.1.1 Connect Request Type Request Body	20
2.2.4.1.2 Connect Request Type Success Response Body	21
2.2.4.1.3 Connect Request Type Failure Response Body	22
2.2.4.2 Execute Request Type	22
2.2.4.2.1 Execute Request Type Request Body	23
2.2.4.2.2 Execute Request Type Success Response Body	23

2.2.4.2.3	Execute Request Type Failure Response Body	24
2.2.4.3	Disconnect Request Type	25
2.2.4.3.1	Disconnect Request Type Request Body	25
2.2.4.3.2	Disconnect Request Type Success Response Body	25
2.2.4.3.3	Disconnect Request Type Failure Response Body	26
2.2.4.4	NotificationWait Request Type	27
2.2.4.4.1	NotificationWait Request Type Request Body	27
2.2.4.4.2	NotificationWait Request Type Success Response Body	27
2.2.4.4.3	NotificationWait Request Type Failure Response Body	28
2.2.5	Request Types for Address Book Server Endpoint	28
2.2.5.1	Bind Request Type	29
2.2.5.1.1	Bind Request Type Request Body	29
2.2.5.1.2	Bind Request Type Success Response Body	29
2.2.5.1.3	Bind Request Type Failure Response Body	30
2.2.5.2	Unbind Request Type	31
2.2.5.2.1	Unbind Request Type Request Body	31
2.2.5.2.2	Unbind Request Type Success Response Body	31
2.2.5.2.3	Unbind Request Type Failure Response Body	32
2.2.5.3	CompareMinIds Request Type	32
2.2.5.3.1	CompareMinIds Request Type Request Body	33
2.2.5.3.2	CompareMinIds Request Type Success Response Body	33
2.2.5.3.3	CompareMinIds Request Type Failure Response Body	34
2.2.5.4	DnToMinId Request Type	35
2.2.5.4.1	DnToMinId Request Type Request Body	35
2.2.5.4.2	DnToMinId Request Type Success Response Body	36
2.2.5.4.3	DnToMinId Request Type Failure Response Body	37
2.2.5.5	GetMatches Request Type	37
2.2.5.5.1	GetMatches Request Type Request Body	37
2.2.5.5.2	GetMatches Request Type Success Response Body	39
2.2.5.5.3	GetMatches Request Type Failure Response Body	41
2.2.5.6	GetPropList Request Type	41
2.2.5.6.1	GetPropList Request Type Request Body	41
2.2.5.6.2	GetPropList Request Type Success Response Body	42
2.2.5.6.3	GetPropList Request Type Failure Response Body	43
2.2.5.7	GetProps Request Type	43
2.2.5.7.1	GetProps Request Type Request Body	44
2.2.5.7.2	GetProps Request Type Success Response Body	44
2.2.5.7.3	GetProps Request Type Failure Response Body	45
2.2.5.8	GetSpecialTable Request Type	46
2.2.5.8.1	GetSpecialTable Request Type Request Body	46
2.2.5.8.2	GetSpecialTable Request Type Success Response Body	47
2.2.5.8.3	GetSpecialTable Request Type Failure Response Body	48
2.2.5.9	GetTemplateInfo Request Type	49
2.2.5.9.1	GetTemplateInfo Request Type Request Body	49
2.2.5.9.2	GetTemplateInfo Request Type Success Response Body	50
2.2.5.9.3	GetTemplateInfo Request Type Failure Response Body	51
2.2.5.10	ModLinkAtt Request Type	51
2.2.5.10.1	ModLinkAtt Request Type Request Body	51
2.2.5.10.2	ModLinkAtt Request Type Success Response Body	52
2.2.5.10.3	ModLinkAtt Request Type Failure Response Body	53
2.2.5.11	ModProps Request Type	53
2.2.5.11.1	ModProps Request Type Request Body	54
2.2.5.11.2	ModProps Request Type Success Response Body	55

2.2.5.11.3	ModProps Request Type Failure Response Body	55
2.2.5.12	QueryRows Request Type	56
2.2.5.12.1	QueryRows Request Type Request Body	56
2.2.5.12.2	QueryRows Request Type Success Response Body	57
2.2.5.12.3	QueryRows Request Type Failure Response Body	58
2.2.5.13	QueryColumns Request Type	59
2.2.5.13.1	QueryColumns Request Type Request Body	59
2.2.5.13.2	QueryColumns Request Type Success Response Body	60
2.2.5.13.3	QueryColumns Request Type Failure Response Body	60
2.2.5.14	ResolveNames Request Type	61
2.2.5.14.1	ResolveNames Request Type Request Body	61
2.2.5.14.2	ResolveNames Request Type Success Response Body	62
2.2.5.14.3	ResolveNames Request Type Failure Response Body	64
2.2.5.15	ResortRestriction Request Type	64
2.2.5.15.1	ResortRestriction Request Type Request Body	64
2.2.5.15.2	ResortRestriction Request Type Success Response Body	65
2.2.5.15.3	ResortRestriction Request Type Failure Response Body	67
2.2.5.16	SeekEntries Request Type	67
2.2.5.16.1	SeekEntries Request Type Request Body	67
2.2.5.16.2	SeekEntries Request Type Success Response Body	69
2.2.5.16.3	SeekEntries Request Type Failure Response Body	70
2.2.5.17	UpdateStat Request Type	70
2.2.5.17.1	UpdateStat Request Type Request Body	70
2.2.5.17.2	UpdateStat Request Type Success Response Body	71
2.2.5.17.3	UpdateStat Request Type Failure Response Body	72
2.2.5.18	GetMailboxUrl Request Type	73
2.2.5.18.1	GetMailboxUrl Request Type Request Body	73
2.2.5.18.2	GetMailboxUrl Request Type Success Response Body	73
2.2.5.18.3	GetMailboxUrl Request Type Failure Response Body	74
2.2.5.19	GetAddressBookUrl Request Type	75
2.2.5.19.1	GetAddressBookUrl Request Type Request Body	75
2.2.5.19.2	GetAddressBookUrl Request Type Success Response Body	75
2.2.5.19.3	GetAddressBookUrl Request Type Failure Response Body	76
2.2.6	PING Request Type	77
2.2.7	Response Meta-Tags	77

3	Protocol Details	78
3.1	Client Details	78
3.1.1	Abstract Data Model	78
3.1.2	Timers	78
3.1.3	Initialization	78
3.1.4	Higher-Layer Triggered Events	78
3.1.5	Message Processing Events and Sequencing Rules	78
3.1.5.1	Creating a Session Context by Using the Connect or Bind Request Type	78
3.1.5.2	Sending Remote Operations by Using the Execute Request Type	78
3.1.5.3	Keeping a Session Context Alive by Using the PING Request Type	79
3.1.5.4	Destroying a Session Context by Using the Disconnect or Unbind Request Type	79
3.1.5.5	Requesting Notification by Using the NotificationWait Request Type	79
3.1.5.6	Handling a Chunked Response	79
3.1.5.7	Reconnecting and Establishing a New Session Context	80
3.1.6	Timer Events	81
3.1.7	Other Local Events	81

3.1.7.1	Handling Communication Failure Sending a Request	81
3.1.7.2	Handling Communication Failure Reading a Response	81
3.2	Server Details	81
3.2.1	Abstract Data Model	81
3.2.2	Timers	81
3.2.3	Initialization	81
3.2.4	Higher-Layer Triggered Events.....	82
3.2.5	Message Processing Events and Sequencing Rules.....	82
3.2.5.1	Responding to a Connect or Bind Request Type Request.....	82
3.2.5.2	Responding to All Request Type Requests	82
3.2.5.3	Responding to a PING Request Type	83
3.2.5.4	Responding to a Disconnect or Unbind Request Type Request	83
3.2.5.5	Responding to a NotificationWait Request Type Request	83
3.2.5.6	Reconnecting and Establishing a New Session Context Server	84
3.2.6	Timer Events	84
3.2.7	Other Local Events	84
4	Protocol Examples.....	85
4.1	Establish a New Session Context.....	85
4.2	Issue ROP Commands to the Server.....	85
4.3	Refresh an Idle Session Context	87
4.4	Re-Establish a Timed-Out Connection to the Server	87
4.5	Disconnect from the Server.....	88
5	Security.....	90
5.1	Security Considerations for Implementers.....	90
5.2	Index of Security Parameters	90
6	Appendix A: Product Behavior	91
7	Change Tracking.....	92
8	Index	98

1 Introduction

The Messaging Application Programming Interface (MAPI) Extensions for HTTP enable a client to access personal messaging data and directory data on a server by sending HTTP requests and receiving responses returned on the same HTTP connection. This protocol extends HTTP/1.1 and HTTP Over TLS (**HTTPS**).

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [\[RFC2119\]](#). Sections 1.5 and 1.9 are also normative but does not contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

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

- ambiguous name resolution (ANR)**
- ASCII**
- code page**
- distinguished name (DN)**
- GUID**
- Hypertext Transfer Protocol (HTTP)**
- Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**
- language code identifier (LCID)**
- name service provider interface (NSPI)**
- NT LAN Manager (NTLM) Authentication Protocol**
- Unicode**

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

- address book**
- address book container**
- address book hierarchy table**
- Address Book object**
- address creation table**
- binary large object (BLOB)**
- cookie**
- endpoint**
- entry ID**
- mailbox**
- Minimal Entry ID**
- property tag**
- recipient**
- remote operation (ROP)**
- restriction**
- ROP request**
- ROP response**
- Session Context**
- Uniform Resource Identifier (URI)**
- Uniform Resource Locator (URL)**

The following terms are specific to this document:

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

References to Microsoft Open Specification documents do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

1.2.1 Normative References

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

[MS-LCID] Microsoft Corporation, "[Windows Language Code Identifier \(LCID\) Reference](#)".

[MS-OXABREF] Microsoft Corporation, "[Address Book Name Service Provider Interface \(NSPI\) Referral Protocol](#)".

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

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

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

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

[MS-OXDCLI] Microsoft Corporation, "[Autodiscover Publishing and Lookup Protocol](#)".

[MS-OXNSPI] Microsoft Corporation, "[Exchange Server Name Service Provider Interface \(NSPI\) Protocol](#)".

[MS-OXOABK] Microsoft Corporation, "[Address Book Object Protocol](#)".

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

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, <http://www.ietf.org/rfc/rfc2616.txt>

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, <http://www.ietf.org/rfc/rfc2818.txt>

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "[Windows Protocols Master Glossary](#)".

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

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

[RFC896] Nagle, J., "Congestion Control in IP/TCP Internetworks", RFC 896, January 1984, <http://www.ietf.org/rfc/rfc896.txt>

1.3 Overview

The MAPI Extensions for HTTP enable a client to communicate with a server to access personal messaging and directory data. When the client initiates communication with the server, the server creates a virtual **Session Context** and returns a session context **cookie** to the client. Once established, the Session Context allows the client to perform operations on the server. Even if network connectivity is interrupted, use of the virtual Session Context, in conjunction with the session context cookie, allows the server to keep the messaging object open and gives the client the ability to reconnect with the Session Context and to continue using the open objects. The client does not have to maintain a persistent, long-lived connection with the server.

Communications with the server are divided into three major function areas: (1) initiating and establishing connection with the server; (2) issuing **remote operations (ROPs)** to the **mailbox** server and sending commands to the **address book** server; (3) terminating communications with the server. If events on the server require more than a configured length of time, the client can either terminate the network connection and re-establish the connection at a later time or continue to keep the connection alive until the processing is completed and the server is ready to return the results to the client.

The following figure shows a simplified overview of client and server communications for a mailbox.

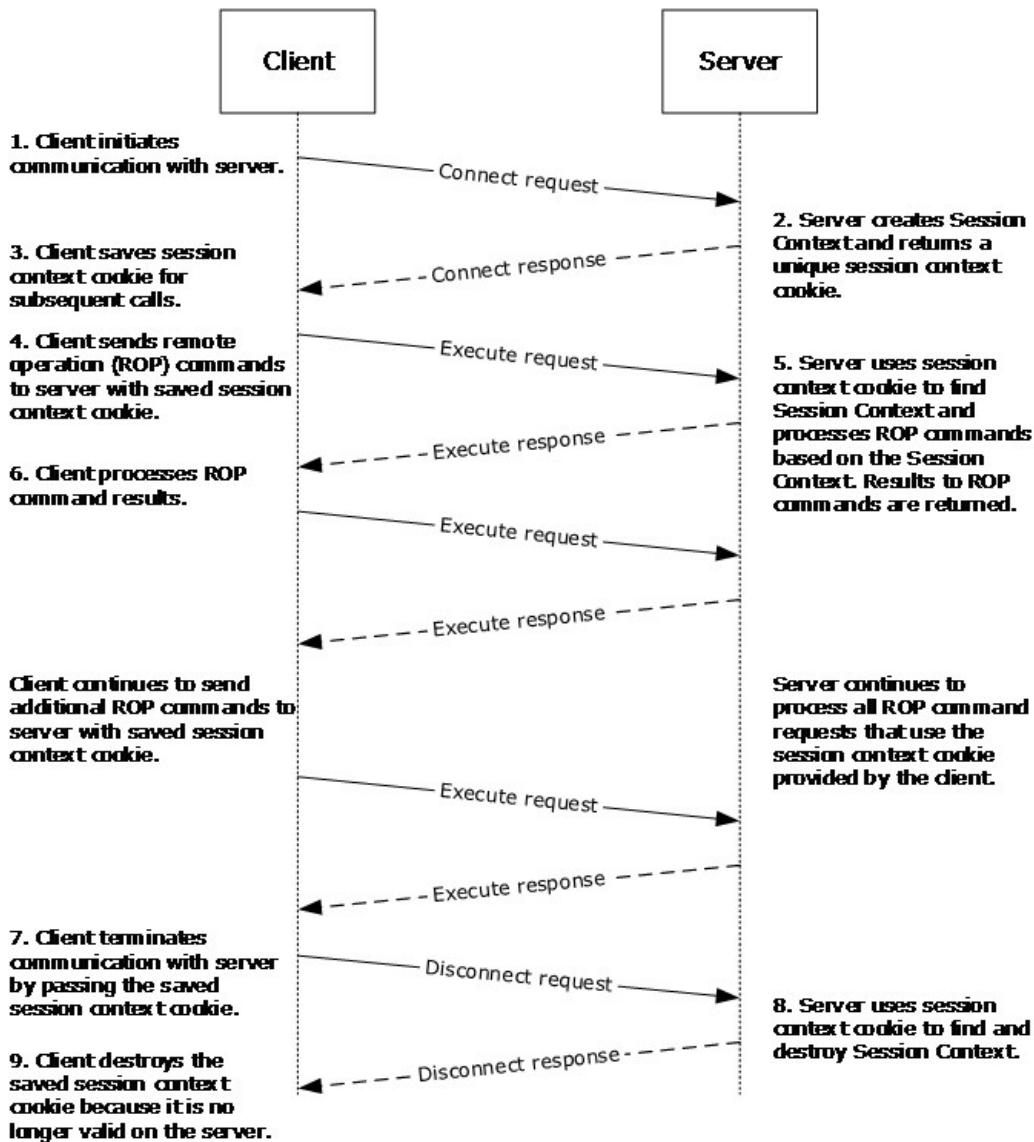


Figure 1: Client/server communications for a mailbox

1.4 Relationship to Other Protocols

A client that implements this protocol can use the Autodiscover Publishing and Lookup Protocol, as described in [\[MS-OXDCLI\]](#), to determine support for this protocol and to identify the target **endpoint (4)** to use for the requests.

This protocol uses **HTTP**, as described in [\[RFC2616\]](#), and HTTPS, as described in [\[RFC2818\]](#), as shown in the following layering diagram.

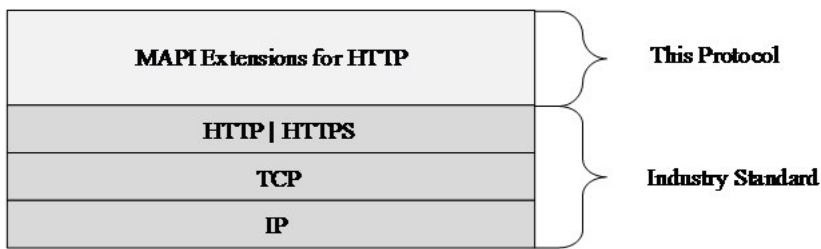


Figure 2: This protocol in relation to other protocols

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

1.5 Prerequisites/Preconditions

It is assumed that a client has discovered support for this protocol via information returned by an Autodiscover response that contains the **Uniform Resource Identifier (URI)** the client will use to access either the mailbox server endpoint (4) or the address book server endpoint (4).

1.6 Applicability Statement

This protocol is applicable to environments that require access to private mailbox messaging end-user data and **NSPI**.

1.7 Versioning and Capability Negotiation

This specification covers versioning and capability negotiation issues in the following areas:

Supported Transports: This protocol uses the transport described in section [2.1](#).

Protocol Versions: The **EMS MDB** protocol interface for this protocol has a single version number of 1.

Security and Authentication Methods: This protocol supports the following authentication methods: basic authentication scheme, **NT LAN Manager (NTLM) Authentication Protocol**, and Negotiate.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The protocol MUST use HTTPS secure requests using version 1.1 of HTTP, as specified in [\[RFC2616\]](#) and [\[RFC2818\]](#). All requests MUST use the **POST** method. **POST** supports uploading a request block and returning a response block.

The Autodiscover response, as specified in [\[MS-OXDSCLI\]](#), contains a URI that the client will use to access the two endpoints (4) used by this protocol: the mailbox sever endpoint (same as that used for the **EMSMDB** interface) and the address book server endpoint (same as that used for the **NSPI** interface).

2.2 Message Syntax

2.2.1 Common Data Types

In addition to the structures defined in the following subsections, this protocol uses data types and structures that are defined in [\[MS-OXCRPC\]](#) section 2.2, [\[MS-OXNSPI\]](#) section 2.2, and [\[MS-OXCADATA\]](#) section 2, as needed, in the request and response bodies defined in section [2.2.4](#), section [2.2.5](#), and section [2.2.6](#).

2.2.1.1 AddressBookPropertyValueList Structure

The **AddressBookPropertyValueList** structure contains a list of properties and their values.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
PropertyValueCount																															
PropertyValues (variable)																															
...																															

PropertyValueCount (4 bytes): An unsigned integer that specifies the number of structures contained in the **PropertyValues** field.

PropertyValues (variable): An array of **TaggedPropertyValue** structures ([\[MS-OXCADATA\]](#) section 2.11.4), each of which specifies a property and its value.

2.2.1.2 AddressBookPropertyRow Structure

The **AddressBookPropertyRow** structure a list of property values without including the **property tags** that correspond to the property values. This structure is used when the list of property tags is known in advance.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags										ValueArray (variable)																					

...

Flags (1 byte): An unsigned integer that indicates whether all property values are present and without error in the **ValueArray** field.

ValueArray (variable): An array of variable-sized structures as follows:

- If the value of the **Flags** field is set to 0x00: The array contains either a **PropertyValue** structure ([\[MS-OXCDATA\]](#) section 2.11.2.1), if the type of the property is specified, or a **TypedPropertyValue** structure ([\[MS-OXCDATA\]](#) section 2.11.3), if the type of the property is **PtypUnspecified** ([\[MS-OXCDATA\]](#) section 2.11.1).
- If the value of the **Flags** field is set to 0x01: The array contains either a **FlaggedPropertyValue** structure ([\[MS-OXCDATA\]](#) section 2.11.5), if the type of the property is specified, or a **FlaggedPropertyValueWithType** structure ([\[MS-OXCDATA\]](#) section 2.11.6), if the type of the property is **PtypUnspecified**.

2.2.1.3 LargePropertyTagArray Structure

The **LargePropertyTagArray** structure contains a list of property tags.

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
PropertyTagCount																															
PropertyTags (variable)																															
...																															

PropertyTagCount (4 bytes): An unsigned integer that specifies the number of structures contained in the **PropertyTags** field. The number is limited to 100,000.

PropertyTags (variable): An array of **PropertyTag** structures ([\[MS-OXCDATA\]](#) section 2.9), each of which contains a property tag that specifies a property.

2.2.2 POST Method

All requests in this protocol are made using the **POST** method, as specified in [\[RFC2616\]](#).

The destination server, request path, and optional parameters for accessing a given mailbox are returned in URIs from Autodiscover. A separate URI is returned in Autodiscover for each endpoint (4). The URI is used by the server to route a request to the appropriate mailbox server.

This protocol uses a common request and common response format for all endpoints (4). Body content that depends on the request type, as specified in section [2.2.4](#), follows the common request or common response to complete the request or response.

All requests **MUST** be authenticated prior to being processed by server. No anonymous requests are allowed.

2.2.2.1 Common Request Format

The common client request across all endpoints (4) used in this protocol has the following format.

```
POST <Autodiscover-provided endpoint> HTTP/1.1
Host: <URL of the host server>
Content-Length: <length of REQUEST BODY>
Content-Type: application/mapi-http
X-RequestType: <?>
X-RequestId: <id>
X-ClientApplication:<client version>

<REQUEST BODY>
```

The client request **MUST** contain the following headers:

- **Host**, as specified in section [2.2.3.1](#)
- **X-RequestType**, as specified in section [2.2.3.3.1](#)
- **X-RequestId**, as specified in section [2.2.3.3.2](#)
- **Content-Type**, as specified in section [2.2.3.2.2](#)
- **Content-Length**, as specified in section [2.2.3.2.1](#)

Other headers are allowed based on the request type used. For details about all headers allowed, see section [2.2.3](#).

2.2.2.2 Common Response Format

The common server response across all endpoints (4) used in this protocol has one of the following formats, depending on whether the response is chunked.

Non-chunked response:

```
HTTP/1.1 200 OK
Content-Length: <length of META-TAGS, ADDITIONAL HEADERS and RESPONSE BODY>
Content-Type: application/mapi-http
X-RequestType: <?>
X-ResponseCode: <?>
X-RequestId: <?>
X-ServerApplication:<server version>

<META-TAGS>
<ADDITIONAL HEADERS>
<RESPONSE BODY>
```

Chunked response:

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
Content-Type: application/mapi-http
X-RequestType: <?>
X-ResponseCode: <?>
X-RequestId: <?>
```

```
X-ServerApplication:<server version>

<META-TAGS>
<ADDITIONAL HEADERS>
<RESPONSE BODY>
```

The first line of all server responses begins with the **POST** method response specified in [\[RFC2616\]](#), "HTTP/1.1". It also contains an HTTP status code, as described in this section.

Both non-chunked and chunked responses MUST contain the following headers:

- **Content-Type**, specified in section [2.2.3.2.2](#)
- **X-RequestType**, specified in section [2.2.3.3.1](#)
- **X-ResponseCode**, specified in section [2.2.3.3.3](#)
- **X-RequestId**, specified in section [2.2.3.3.2](#)

In addition to these headers, a non-chunked response contains the **Content-Length** header, specified in section [2.2.3.2.1](#), and a chunked response contains the **Transfer-Encoding** header, specified in section [2.2.3.2.5](#).

The server returns a "200 OK" HTTP status code even if the request failed for all non-exceptional failures. The server deviates from a "200 OK" HTTP status code only for authentication ("401 Access Denied"), redirection ("302 Redirect"), or if there is some exceptional condition for which the server cannot continue. Exceptional failures will produce a "500 Internal Server Error" status code or other 5xx-level status codes. For more details about the status codes, see [\[RFC2616\]](#) section 10.

For success or non-exceptional conditions (most failures), the server will return "200 OK" and an **X-ResponseCode** header. This header contains a numerical value that indicates the specific failure that occurred on the server. An **X-ResponseCode** of 0 (zero) means success from the perspective of the protocol transport, and the client parses the RESPONSE BODY based on the request that was issued. If the **X-ResponseCode** is not zero, the server populates the RESPONSE BODY with diagnostic information.

2.2.3 Header Fields

The following sections specify the header fields that are used by this protocol.

2.2.3.1 Host Header Field

The **Host** header field specifies the server and the endpoint (4) that are being used for the connection. The value of this header field is in the format of a **URL** and port number, as specified in [\[RFC2616\]](#) section 14.23.

2.2.3.2 Entity Header Fields

The following sections specify the entity header fields that are used by this protocol.

2.2.3.2.1 Content-Length Header Field

The **Content-Length** header field contains the length, in bytes, of the request or response body, as specified in section [2.2.4](#). This header is not used in chunked responses.

2.2.3.2.2 Content-Type Header Field

The **Content-Type** header field specifies the type of content. The **Content-Type** header MUST contain the string "application/mapi-http" on requests. The **Content-Type** header MUST contain the string "application/mapi-http" on responses with **X-ResponseCode** header of 0. If **X-ResponseCode** is non-zero, the **Content-Type** header MUST contain the string "text/html".

2.2.3.2.3 Set-Cookie Header Field

The **Set-Cookie** header field contains an opaque value of the form <cookie name>=<opaque string>. For details on the use of this header field, see section [3.2.5.1](#). Any **Set-Cookie** header value sent by the server to the client in response to a request is saved by the client for future use, as specified in section [3.1.1](#).

2.2.3.2.4 Cookie Header Field

The **Cookie** header field contains the name and value of one or more cookies. The name and value of a cookie are formatted as <cookie name>=<opaque string>. Multiple cookies are separated by a semi-colon. For example: cookie1=value1; cookie2=value2. The client's use of cookies is specified in section [3.1.5.1](#).

2.2.3.2.5 Transfer-Encoding Header Field

The **Transfer-Encoding** header field contains the string "chunked" transfer coding, as specified in [\[RFC2616\]](#) section 3.6. The use of the **Transfer-Encoding** header field is specified in section [3.1.5.6](#) and section [3.2.5.2](#).

2.2.3.3 X-Header Fields

The following sections specify the X-header fields that are used by this protocol.

2.2.3.3.1 X-RequestType Header Field

The **X-RequestType** header identifies the request type for a mailbox server endpoint (4) or an address book server endpoint (4). For details about these endpoints (4), see section [2.2.4](#) and section [2.2.5](#), respectively.

The following table lists the possible values for the **X-RequestType** header.

Value	Description	Reference
Connect	Indicates the Connect request type for the mailbox server endpoint (4).	Section 2.2.4.1
Execute	Indicates the Execute request type for the mailbox server endpoint (4).	Section 2.2.4.2
Disconnect	Indicates the Disconnect request type for the mailbox server endpoint (4).	Section 2.2.4.3
NotificationWait	Indicates the NotificationWait request type for the mailbox server endpoint (4).	Section 2.2.4.4
PING	Indicates the PING request type for the mailbox server endpoint (4) and address book server endpoint (4).	Section 2.2.6

Value	Description	Reference
Bind	Indicates the Bind request type for the address book server endpoint (4).	Section 2.2.5.1
Unbind	Indicates the Unbind request type for the address book server endpoint (4).	Section 2.2.5.2
CompareMIDs	Indicates the CompareMinIds request type for the address book server endpoint (4).	Section 2.2.5.3
DNToMID	Indicates the DNToMinId request type for the address book server endpoint (4).	Section 2.2.5.4
GetMatches	Indicates the GetMatches request type for the address book server endpoint (4).	Section 2.2.5.5
GetPropList	Indicates the GetPropList request type for the address book server endpoint (4).	Section 2.2.5.6
GetProps	Indicates the GetProps request type for the address book server endpoint (4).	Section 2.2.5.7
GetSpecialTable	Indicates the GetSpecialTable request type for the address book server endpoint (4).	Section 2.2.5.8
GetTemplateInfo	Indicates the GetTemplateInfo request type for the address book server endpoint (4).	Section 2.2.5.9
ModLinkAtt	Indicates the ModLinkAtt request type for the address book server endpoint (4).	Section 2.2.5.10
ModProps	Indicates the ModProps request type for the address book server endpoint (4).	Section 2.2.5.11
QueryColumns	Indicates the QueryColumns request type for the address book server endpoint (4).	Section 2.2.5.13
QueryRows	Indicates the QueryRows request type for the address book server endpoint (4).	Section 2.2.5.12
ResolveNames	Indicates the ResolveNames request type for the address book server endpoint (4).	Section 2.2.5.14
ResortRestriction	Indicates the ResortRestriction request type for the address book server endpoint (4).	Section 2.2.5.15
SeekEntries	Indicates the SeekEntries request type for the address book server endpoint (4).	Section 2.2.5.16
UpdateStat	Indicates the UpdateStat request type for the address book server endpoint (4).	Section 2.2.5.17
GetMailboxUrl	Indicates the GetMailboxUrl request type for the specified mailbox server endpoint (4).	Section 2.2.5.18
GetAddressBookUrl	Indicates the GetAddressBookUrl request type for the address book server endpoint (4).	Section 2.2.5.19

2.2.3.3.2 X-RequestId Header Field

The **X-RequestId** header field MUST be a combination of a globally unique value in the format of a **GUID** followed by an increasing decimal counter which MUST increase with every new HTTP request (for example, "E2EA6C1C-E61B-49E9-9CFB-38184F907552:**123456**"). The GUID portion of the **X-RequestId** header MUST be unique across all client instances and MUST NOT change for the life of the client instance.

2.2.3.3.3 X-ResponseCode Header Field

The **X-ResponseCode** header contains a numerical value that represents the specific result that occurred on the server.

An **X-ResponseCode** of 0 (zero) means success from the perspective of the protocol transport, and the client SHOULD parse the response body based on the request that was issued. If the **X-ResponseCode** is not zero, the client parses the response body for diagnostic information.

The response codes that can be returned in the **X-ResponseCode** header are listed in the following table.

Code	Error	Description
0	Success	The request was properly formatted and accepted.
1	Unknown Failure	The request produced an unknown failure.
2	Invalid Verb	The request has an invalid verb.
3	Invalid Path	The request has an invalid path.
4	Invalid Header	The request has an invalid header.
5	Invalid Request Type	The request has an invalid X-RequestType header.
6	Invalid Context Cookie	The request has an invalid session context cookie.
7	Missing Header	The request has a missing required header.
8	Anonymous Not Allowed	The request is anonymous, but anonymous requests are not accepted.
9	Too Large	The request is too large.
10	Context Not Found	The Session Context is not found.
11	No Privilege	The client has no privileges to the Session Context.
12	Invalid Request Body	The request body is invalid.
13	Missing Cookie	The request is missing a required cookie.
14	Reserved	This value MUST be ignored by the client.
15	Invalid Sequence	The request has violated the sequencing requirement of one request at a time per Session Context.

Code	Error	Description
16	Endpoint Disabled	The endpoint (4) is disabled.
17	Invalid Response	The response is invalid.
18	Endpoint Shutting Down	The endpoint (4) is shutting down.

2.2.3.3.4 X-ClientInfo Header Field

The **X-ClientInfo** header contains opaque information. The server MUST return this header with the same opaque information in the response back to the client.

2.2.3.3.5 X-PendingPeriod Header Field

The **X-PendingPeriod** header field, returned by the server, specifies the number of milliseconds to be expected between keep-alive **PENDING** meta-tags in the response stream while the server is executing the request. The default value of this header is 15000 milliseconds (15 seconds).

For more details about client keep-alive functions, see section [3.1.5.3](#). For more details about the server keep-alive function, see section [3.2.5.3](#).

2.2.3.3.6 X-ClientApplication Header Field

On every request, the client includes the **X-ClientApplication** header to indicate to the server what version of the client is being used. The value of this header field has the following format: "Outlook/15.xx.xxxx.xxxx". For details about client versions, see [\[MS-OXCRPC\]](#) section 3.2.4.1.3.

2.2.3.3.7 X-ServerApplication Header Field

On every response, the server includes the **X-ServerApplication** header to indicate to the client what server version is being used. The value of this header field has the following format: "Exchange/15.xx.xxxx.xxx". For details about server versions, see [\[MS-OXCRPC\]](#) section 3.1.4.1.3.

2.2.3.3.8 X-ExpirationInfo Header Field

The **X-ExpirationInfo** header is returned by the server in every response to notify the client of the number of milliseconds before the server times-out the Session Context.

2.2.3.3.9 X-ElapsedTime Header

The **X-ElapsedTime** header specifies the amount of time, in milliseconds, that the server took to process the request. This header is returned by the server as an additional header in the final response.

2.2.3.3.10 X-StartTime Header

The **X-StartTime** header specifies the time that the server started processing the request. This header is returned by the server as an additional header in the final response. This header follows the date/time format, as specified in [\[RFC2616\]](#).

2.2.4 Request Types for Mailbox Server Endpoint

The mailbox server endpoint (4) supports the four request types that are specified in section [2.2.4.1](#) through section [2.2.4.4](#). The mailbox server endpoint (4) also supports the **PING** request type, which is specified in section [2.2.6](#). The **X-RequestType** header, specified in section [2.2.3.3.1](#), identifies which request type is being used.

The request and response bodies associated with the specific request types are each a raw binary data **binary large object (BLOB)** that follows the common request format, as specified in section [2.2.2.1](#), and the common response format, as specified in section [2.2.2.2](#). Request and response bodies are separated from the common request and response by a blank line, as specified in [\[RFC2616\]](#).

2.2.4.1 Connect Request Type

The **Connect** request type is used to establish a Session Context with the server, as specified in section [3.1.5.1](#) and section [3.1.5.7](#).

2.2.4.1.1 Connect Request Type Request Body

The **Connect** request type request body contains the following fields. For more details about the fields, see the *[in]* parameters in [\[MS-OXCRPC\]](#) section 3.1.4.1.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
UserDn (variable)																															
...																															
Flags																															
DefaultCodePage																															
LcidSort																															
LcidString																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

UserDn (variable): A null-terminated **ASCII** string that specifies the **DN (1)** of the user who is requesting the connection.

Flags (4 bytes): A set of flags that designate the type of connection being requested.

DefaultCodePage (4 bytes): An unsigned integer that specifies the **code page** that the server is being requested to use for string values of properties.

LcidSort (4 bytes): An unsigned integer that specifies the **language code identifier (LCID)**, as specified in [\[MS-LCID\]](#), to be used for sorting.

LcidString (4 bytes): An unsigned integer that specifies the language code identifier (LCID), as specified in [\[MS-LCID\]](#), to be used for everything other than sorting.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.1.2 Connect Request Type Success Response Body

The **Connect** request type success response body contains the following fields. For more details about the fields, see the *[out]* parameters in [\[MS-OXCRPC\]](#) section 3.1.4.1.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
PollsMax																															
RetryCount																															
RetryDelay																															
DnPrefix (variable)																															
...																															
DisplayName (variable)																															
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXCRPC\]](#) section 3.1.4.1 as the Return Values.

PollsMax (4 bytes): An unsigned integer that specifies the number of milliseconds for the maximum polling interval.

RetryCount (4 bytes): An unsigned integer that specifies the number of times to retry request types.

RetryDelay (4 bytes): An unsigned integer that specifies the number of milliseconds for the client to wait before retrying a failed request type.

DnPrefix (variable): A null-terminated ASCII string that specifies the DN (1) prefix to be used for building message **recipients (1)**.

DisplayName (variable): A null-terminated **Unicode** string that specifies the display name of the user who is specified in the **UserDn** field of the **Connect** request type request body.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.1.3 Connect Request Type Failure Response Body

The **Connect** request type failure response body contains the following fields. For more details about the fields, see the *[out]* parameters in [\[MS-OXCRPC\]](#) section 3.1.4.1.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.2 Execute Request Type

The **Execute** request type is used by the client to send remote operation requests to the server.

2.2.4.2.1 Execute Request Type Request Body

The **Execute** request type request body contains the following fields. For more details about the fields, see the *[in]* parameters in [\[MS-OXCRPC\]](#) section 3.1.4.2.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
RopBufferSize																															
RopBuffer (variable)																															
...																															
MaxRopOut																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Flags (4 bytes): A set of flags that specify to the server how to build the **ROP responses** in the **RopBuffer** field of the **Execute** request type success response body, as specified in section [2.2.4.2.2](#).

RopBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **RopBuffer** field.

RopBuffer (variable): An array of bytes that constitute the **ROP requests** payload. The size of this field, in bytes, is specified by the **RopBufferSize** field.

MaxRopOut (4 bytes): An unsigned integer that specifies the maximum size for the **RopBuffer** field of the **Execute** request type success response body.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.2.2 Execute Request Type Success Response Body

The **Execute** request type success response body contains the following fields. For more details about the fields, see the *[out]* parameters in [\[MS-OXCRPC\]](#) section 3.1.4.2.

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
StatusCode																															
ErrorCode																															
Flags																															
RopBufferSize																															
RopBuffer (variable)																															
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXCRPC\]](#) section 3.1.4.2 as the Return Values.

Flags (4 bytes): Reserved. The server MUST set this field to 0x00000000 and the client MUST ignore this field.

RopBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **RopBuffer** field.

RopBuffer (variable): An array of bytes that constitute the ROP responses payload. The size of this field, in bytes, is specified by the **RopBufferSize** field.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.2.3 Execute Request Type Failure Response Body

The **Execute** request type failure response body contains the following fields. For more details about the fields, see the *[out]* parameters in [\[MS-OXCRPC\]](#) section 3.1.4.2.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.3 Disconnect Request Type

The **Disconnect** request type is used by the client to delete a Session Context with the server, as specified in section [3.1.5.4](#).

2.2.4.3.1 Disconnect Request Type Request Body

The **Disconnect** request type request body contains the following fields. For more details about the fields, see the *[in]* parameters in [\[MS-OXCRPC\]](#) section 3.1.4.3.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.3.2 Disconnect Request Type Success Response Body

The **Disconnect** request type success response body contains the following fields. For more details about the fields, see the *[out]* parameters in [\[MS-OXCRPC\]](#) section 3.1.4.3.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXCRPC\]](#) section 3.1.4.3 as the Return Values.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.3.3 Disconnect Request Type Failure Response Body

The **Disconnect** request type failure response body contains the following fields. For more details about the fields, see the *[out]* parameters in [\[MS-OXCRPC\]](#) section 3.1.4.3.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.4 NotificationWait Request Type

The **NotificationWait** request type is used by the client to request that the server notify the client when a processing request that takes an extended amount of time completes.

2.2.4.4.1 NotificationWait Request Type Request Body

The **NotificationWait** request type request body contains the following fields. For more details about the fields, see the *[in]* parameters in [\[MS-OXCRPC\]](#) section 3.3.4.1.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Flags (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.4.2 NotificationWait Request Type Success Response Body

The **NotificationWait** request type success response body contains the following fields. For more details about the fields, see the *[out]* parameters in [\[MS-OXCRPC\]](#) section 3.3.4.1.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
EventPending																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXCRPC\]](#) section 3.3.4.1 as the Return Values.

EventPending (4 bytes): An unsigned integer that indicates whether an event is pending on the Session Context. The value 0x00000001 indicates that an event is pending; the value 0x00000000 indicates that no event is pending.

The client MUST send the **Execute** request type, as specified in section [2.2.4.2](#), with additional data in the request body if there is additional data to send to the server, or with an empty request body if there is no additional data to send to the server. The server will return the event details in the **Execute** request type response body.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.4.4.3 NotificationWait Request Type Failure Response Body

The **NotificationWait** request type failure response body contains the following fields. For more details about the fields, see the *[out]* parameters in [\[MS-OXCRPC\]](#) section 3.3.4.1.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5 Request Types for Address Book Server Endpoint

The address book server endpoint (4) supports the 19 request types that are specified in section [2.2.5.1](#) through section [2.2.5.19](#). The address book server endpoint (4) also supports the **PING** request type, which is specified in section [2.2.6](#). The **X-RequestType** header, specified in section [2.2.3.3.1](#), identifies which request type is being used.

The request and response bodies associated with the specific request types are each a raw binary data binary large object (BLOB) that follows the common request format, as specified in section [2.2.2.1](#), and the common response format, as specified in section [2.2.2.2](#). Request and response bodies are separated from the common request and response by a blank line, as specified in [\[RFC2616\]](#).

2.2.5.1 Bind Request Type

The **Bind** request type is used by the client to establish a Session Context with the server, as specified in section [3.1.5.1](#) and section [3.1.5.7](#). For information regarding the server-side processing of the **Bind** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.1.

2.2.5.1.1 Bind Request Type Request Body

The **Bind** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
HasState										State																					
...																															
(State field continues for 7 rows)																															
...										AuxiliaryBufferSize																					
...										AuxiliaryBuffer (variable)																					
...																															

Flags (4 bytes): A set of bit flags that specify the authentication type for the connection. The server MUST ignore values other than the bit flag **fAnonymousLogin** (0x00000020).

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific **address book container**. This field is present when the **HasState** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.1.2 Bind Request Type Success Response Body

The **Bind** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
ServerGuid																															
...																															
...																															
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

ServerGuid (16 bytes): A GUID that is associated with a specific address book server.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.1.3 Bind Request Type Failure Response Body

The **Bind** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															

...

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.2 Unbind Request Type

The **Unbind** request type is used by the client to delete a Session Context with the server, as specified in section 3.1.5.4. For information regarding the server-side processing of the **Unbind** request type, see Server Processing Rules in [MS-OXNSPI] section 3.1.4.1.2.

2.2.5.2.1 Unbind Request Type Request Body

The **Unbind** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.2.2 Unbind Request Type Success Response Body

The **Unbind** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															

ErrorCode
AuxiliaryBufferSize
AuxiliaryBuffer (variable)
...

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.2.3 Unbind Request Type Failure Response Body

The **Unbind** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.3 CompareMinIds Request Type

The **CompareMinIds** request type is used by the client to compare the positions of two objects in an address book container. For information regarding the server-side processing of the **CompareMinIds** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.12.

2.2.5.3.1 CompareMinIds Request Type Request Body

The **CompareMinIds** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
HasState										State																					
...																															
(State field continues for 7 rows)																															
...										MinimalId1																					
...										MinimalId2																					
...										AuxiliaryBufferSize																					
...										AuxiliaryBuffer (variable)																					
...																															

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

MinimalId1 (4 bytes): A **MinimalEntryID** structure ([\[MS-OXNSPI\]](#) section 2.2.9.1) that specifies the **Minimal Entry ID** of the first object.

MinimalId2 (4 bytes): A **MinimalEntryID** structure that specifies the Minimal Entry ID of the second object.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.3.2 CompareMinIds Request Type Success Response Body

The **CompareMinIds** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
Result																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

Result (4 bytes): A signed integer that specifies the result of the comparison. The valid values for this field are specified in the following table.

Value	Meaning
0 (zero)	The position of the object specified by the MinimalId1 field of the request body is the same as the position of the object specified by the MinimalId2 field. That is, the two fields specify the same object.
A value less than 0 (zero)	The position of the object specified by the MinimalId1 field of the request body precedes the position of the object specified by the MinimalId2 field.
A value greater than 0 (zero)	The position of the object specified by the MinimalId1 field of the request body succeeds the position of the object specified by the MinimalId2 field.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.3.3 CompareMinIds Request Type Failure Response Body

The **CompareMinIds** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															

AuxiliaryBufferSize
AuxiliaryBuffer (variable)
...

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.4 DnToMinId Request Type

The **DnToMinId** request type is used by the client to map a set of DNs (1) to a set of Minimal Entry IDs. For information regarding the server-side processing of the **DnToMinId** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.13.

2.2.5.4.1 DnToMinId Request Type Request Body

The **DnToMinId** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
HasNames										Names (variable)																					
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

HasNames (1 byte): A Boolean value that specifies whether the **Names** field is present.

Names (variable): A **StringsArray_r** structure ([\[MS-OXNSPI\]](#) section 2.2.7.1) that contains the list of distinguished names (DNs) (1) to be mapped to Minimal Entry IDs. This field is present when the value of the **HasNames** field is nonzero and is not present otherwise. For details about Minimal Entry IDs, see [\[MS-OXNSPI\]](#) section 2.2.9.1.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.4.2 DnToMinId Request Type Success Response Body

The **DnToMinId** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
HasMinimalIds																MinimalIdCount															
...																MinimalIds (variable)															
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field **MUST** be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

HasMinimalIds (1 byte): A Boolean value that specifies whether the **MinimalIdCount** and **MinimalIds** fields are present.

MinimalIdCount (4 bytes): An unsigned integer that specifies the number of structures in the **MinimalIds** field. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

MinimalIds (variable): An array of **MinimalEntryID** structures ([\[MS-OXNSPI\]](#) section 2.2.9.1), each of which specifies a Minimal Entry ID that matches a requested distinguished name (DN) (1). This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.4.3 DnToMinId Request Type Failure Response Body

The **DnToMinId** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.5 GetMatches Request Type

The **GetMatches** request type is used by the client to get an Explicit Table, in which the rows are determined by the specified criteria. For details about Explicit Tables, see [\[MS-OXNSPI\]](#) section 3.1.4.4.2. For information regarding the server-side processing of the **GetMatches** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.10.

2.2.5.5.1 GetMatches Request Type Request Body

The **GetMatches** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
HasState																State															
...																															
(State field continues for 7 rows)																															

...	HasMinimalIds	MinimalIdCount
...		MinimalIds (variable)
...		
InterfaceOptionFlags		
HasFilter	Filter (variable)	
...		
HasPropertyName	PropertyNameGuid	
...		
...		
...		
...	PropertyNameId	
...	RowCount	
...	HasColumns	Columns (variable)
...		
AuxiliaryBufferSize		
AuxiliaryBuffer (variable)		
...		

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the value of the **HasState** field is nonzero and is not present otherwise.

HasMinimalIds (1 byte): A Boolean value that specifies whether the **MinimalIdCount** and **MinimalIds** fields are present.

MinimalIdCount (4 bytes): An unsigned integer that specifies the number of structures present in the **MinimalIds** field. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

MinimalIds (variable): An array of **MinimalEntryID** structures ([\[MS-OXNSPI\]](#) section 2.2.9.1) that constitute an Explicit Table. For details about Explicit Tables, see [\[MS-OXNSPI\]](#) section 3.1.4.4.2. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

InterfaceOptionFlags (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

HasFilter (1 byte): A Boolean value that specifies whether the **Filter** field is present.

Filter (variable): A **restriction (2)**, as specified in [\[MS-OXCDATA\]](#) section 2.12, that is to be applied to the rows in the address book container.

HasPropertyName (1 byte): A Boolean value that specifies whether the **PropertyNameGuid** and **PropertyNameId** fields are present.

PropertyNameGuid (16 bytes): The GUID of the property to be opened. This field is present when the value of the **HasPropertyName** field is nonzero and is not present otherwise.

PropertyNameId (4 bytes): A 4-byte value that specifies the ID of the property to be opened. This field is present when the value of the **HasPropertyName** field is nonzero and is not present otherwise.

RowCount (4 bytes): An unsigned integer that specifies the number of rows the client is requesting.

HasColumns (1 byte): A Boolean value that specifies whether the **Columns** field is present.

Columns (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the columns that the client is requesting. This field is present when the value of the **HasColumns** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.5.2 GetMatches Request Type Success Response Body

The **GetMatches** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
HasState																State															
...																															
(State field continues for 7 rows)																															

...	HasMinimalIds	MinimalIdCount
...		MinimalIds (variable)
...		
HasColsAndRows	Columns (variable)	
...		
RowCount		
RowData (variable)		
...		
AuxiliaryBufferSize		
AuxiliaryBuffer (variable)		
...		

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. . This field is present when the **HasState** field is nonzero and is not present otherwise.

HasMinimalIds (1 byte): A Boolean value that specifies whether the **MinimalIdCount** and **MinimalIds** fields are present.

MinimalIdCount (4 bytes): An unsigned integer that specifies the number of structures present in the **MinimalIds** field. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

MinimalIds (variable): An array of **MinimalEntryID** structures ([\[MS-OXNSPI\]](#) section 2.2.9.1), each of which is the ID of an object found. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

HasColsAndRows (1 byte): A Boolean value that specifies whether the **Columns**, **RowCount**, and **RowData** fields are present.

Columns (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the columns used for each row returned. This field is present when the value of the **HasColumnsAndRows** field is nonzero and is not present otherwise.

RowCount (4 bytes): An unsigned integer that specifies the number of structures in the **RowData** field. This field is present when the value of the **HasColumnsAndRows** field is nonzero and is not present otherwise.

RowData (variable): An array of **AddressBookPropertyRow** structures (section [2.2.1.2](#)), each of which specifies the row data for the entries requested. This field is present when the **HasColumnsAndRows** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.5.3 GetMatches Request Type Failure Response Body

The **GetMatches** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.6 GetPropList Request Type

The **GetPropList** request type is used by the client to get a list of all of the properties that have values on an object. For information regarding the server-side processing of the **GetPropList** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.6.

2.2.5.6.1 GetPropList Request Type Request Body

The **GetPropList** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
MinimalId																															
CodePage																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Flags (4 bytes): A set of bit flags that specify options to the server. The server MUST ignore values other than the bit flag **fSkipObjects** (0x00000001).

MinimalId (4 bytes): A **MinimalEntryID** structure ([\[MS-OXNSPI\]](#) section 2.2.9.1) that specifies the object for which to return properties.

CodePage (4 bytes): An unsigned integer that specifies the code page that the server is being requested to use for string values of properties. For more details about code pages, see [\[MS-OXNSPI\]](#) section 2.2.1.5.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.6.2 GetPropList Request Type Success Response Body

The **GetPropList** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
HasPropertyTags										PropertyTags (variable)																					
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															

...

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

HasPropertyTags (1 byte): A Boolean value that specifies whether the **PropertyTags** field is present.

PropertyTags (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that contains the property tags of properties that have values on the requested object. This field is present when the value of the **HasPropertyTags** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.6.3 GetPropList Request Type Failure Response Body

The **GetPropList** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.7 GetProps Request Type

The **GetProps** request type is used by the client to get specific properties on an object. For information regarding the server-side processing of the **GetProps** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.7.

2.2.5.7.1 GetProps Request Type Request Body

The **GetProps** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
HasState										State																					
...																															
(State field continues for 7 rows)																															
...										HasPropertyTags										PropertyTags (variable)											
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Flags (4 bytes): A set of bit flags that specify options to the server. The server MUST ignore values other than the bit flags **fEphID** (0x00000002) and **fSkipObjects** (0x00000001).

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI section 2.2.8\]](#)) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasPropertyTags (1 byte): A Boolean value that specifies whether the **PropertyTags** field is present.

PropertyTags (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that contains the property tags of the properties that the client is requesting. This field is present when the value of the **HasPropertyTags** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.7.2 GetProps Request Type Success Response Body

The **GetProps** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
CodePage																															
HasPropertyValues	PropertyValues (variable)																														
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

CodePage (4 bytes): An unsigned integer that specifies the code page that the server used to express string properties. For details about code pages, see [\[MS-OXNSPI\]](#) section 2.2.1.5.

HasPropertyValues (1 byte): A Boolean value that specifies whether the **PropertyValues** field is present.

PropertyValues (variable): An **AddressBookPropertyValueList** structure (section [2.2.1.1](#)) that contains the values of the properties requested. This field is present when the value of the **HasPropertyValues** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.7.3 GetProps Request Type Failure Response Body

The **GetProps** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															

AuxiliaryBufferSize
AuxiliaryBuffer (variable)
...

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.8 GetSpecialTable Request Type

The **GetSpecialTable** request type is used by the client to get a special table, which can be either an **address book hierarchy table** or an **address creation table**. For details about tables that are used in the NSPI data model, see [\[MS-OXNSPI\]](#) section 3.1.4.4. For information regarding the server-side processing of the **GetSpecialTable** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.3.

2.2.5.8.1 GetSpecialTable Request Type Request Body

The **GetSpecialTable** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
HasState																State															
...																															
(State field continues for 7 rows)																															
...																HasVersion								Version							
...																AuxiliaryBufferSize															
...																AuxiliaryBuffer (variable)															
...																															

Flags (4 bytes): A set of bit flags that specify options to the server. The server MUST ignore values other than the bit flags **NspiAddressCreationTemplates** (0x00000002) and **NspiUnicodeStrings** (0x00000004).

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasVersion (1 byte): A Boolean value that specifies whether the **Version** field is present.

Version (4 bytes): An unsigned integer that specifies the version number of the address book hierarchy table that the client has. This field is present when the value of the **HasVersion** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.8.2 GetSpecialTable Request Type Success Response Body

The **GetSpecialTable** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
CodePage																															
HasVersion										Version																					
...										HasRows										RowCount											
...										Rows (variable)																					
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

CodePage (4 bytes): An unsigned integer that specifies the code page the server used to express string properties. The required code pages are specified in For details about code pages, see .

HasVersion (1 byte): A Boolean value that specifies whether the **Version** field is present.

Version (4 bytes): An unsigned integer that specifies the version number of the address book hierarchy table that the server has. This field is present when the value of the **HasVersion** field is nonzero and is not present otherwise.

HasRows (1 byte): A Boolean value that specifies whether the **RowCount** and **Rows** fields are present.

RowCount (4 bytes): An unsigned integer that specifies the number of structures in the **Rows** field. This field is present when the value of the **HasRows** field is nonzero and is not present otherwise.

Rows (variable): An array of **AddressBookPropertyValueList** structures (section [2.2.1.1](#)), each of which contains a row of the table that the client requested. This field is present when the value of the **HasRows** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.8.3 GetSpecialTable Request Type Failure Response Body

The **GetSpecialTable** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.9 GetTemplateInfo Request Type

The **GetTemplateInfo** request type is used by the client to get information about a template that is used by the address book. For information regarding the server-side processing of the **GetTemplateInfo** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.18.

2.2.5.9.1 GetTemplateInfo Request Type Request Body

The **GetTemplateInfo** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
DisplayType																															
HasTemplateDn								TemplateDn (variable)																							
...																															
CodePage																															
LocaleId																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Flags (4 bytes): A set of bit flags that specify options to the server. The server MUST ignore values other than the bit flags **TI_HELPFILE_NAME** (0x00000020), **TI_HELPFILE_CONTENTS** (0x00000040), **TI_SCRIPT** (0x00000004), **TI_TEMPLATE** (0x00000001), and **TI_EMT** (0x00000010).

DisplayType (4 bytes): An unsigned integer that specifies the display type of the template for which information is requested. The valid values for this field are specified in [\[MS-OXNSPI\]](#) section 2.2.1.3.

HasTemplateDn (1 byte): A Boolean value that specifies whether the **TemplateDn** field is present.

TemplateDn (variable): A null-terminated ASCII string that specifies the DN (1) of the template requested. This field is present when the **HasTemplateDn** field is nonzero and is not present otherwise.

CodePage (4 bytes): An unsigned integer that specifies the code page of the template for which information is requested.

LocaleId (4 bytes): An unsigned integer that specifies the language code identifier (LCID), as specified in [\[MS-LCID\]](#), of the template for which information is requested.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.9.2 GetTemplateInfo Request Type Success Response Body

The **GetTemplateInfo** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
CodePage																															
HasRow								Row (variable)																							
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field **MUST** be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

CodePage (4 bytes): An unsigned integer that specifies the code page the server used to express string values of properties.

HasRow (1 byte): A Boolean value that specifies whether the **Row** field is present.

Row (variable): A **AddressBookPropertyValueList** structure (section [2.2.1.1](#)) that specifies the information that the client requested. This field is present when the value of the **HasRow** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.9.3 GetTemplateInfo Request Type Failure Response Body

The **GetTemplateInfo** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.10 ModLinkAtt Request Type

The **ModLinkAtt** request type is used by the client to modify a specific property of a row in the address book. For information regarding the server-side processing of the **ModLinkAtt** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.15.

2.2.5.10.1 ModLinkAtt Request Type Request Body

The **ModLinkAtt** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
PropertyTag																															
MinimalId																															
HasEntryIds										EntryIdCount																					
...										EntryIds (variable)																					
...																															

AuxiliaryBufferSize
AuxiliaryBuffer (variable)
...

Flags (4 bytes): A set of bit flags that specify options to the server. The server MUST ignore values other than the bit flag **fDelete** (0x00000001).

PropertyTag (4 bytes): A **PropertyTag** structure ([\[MS-OXCDATA\]](#) section 2.9) that specifies the property to be modified.

The **PidTagAddressBookMember** property ([\[MS-OXOABK\]](#) section 2.2.6.1) can be modified only on an **Address Book object** that has a display type of **DT_DISTLIST**. The **PidTagAddressBookPublicDelegates** property ([\[MS-OXOABK\]](#) section 2.2.5.5) can be modified only on an Address Book object that has a display type of **DT_MAILUSER**. For details about display types, see [\[MS-OXNSPI\]](#) section 2.2.1.3.

MinimalId (4 bytes): A **MinimalEntryID** structure ([\[MS-OXNSPI\]](#) section 2.2.9.1) that specifies the Minimal Entry ID of the address book row to be modified.

HasEntryIds (1 byte): A Boolean value that specifies whether the **EntryIdCount** and **EntryIds** fields are present.

EntryIdCount (4 bytes): An unsigned integer that specifies the count of structures in the **EntryIds** field. This field is present when the value of the **HasEntryIds** field is nonzero and is not present otherwise.

EntryIds (variable): An array of **entry IDs**, each of which is either an **EphemeralEntryID** structure ([\[MS-OXNSPI\]](#) section 2.2.9.2) or a **PermanentEntryID** structure ([\[MS-OXNSPI\]](#) section 2.2.9.3). Each entry ID in the array specifies an address book row in which the specified property is to be modified. This field is present when the value of the **HasEntryIds** field is nonzero and is not present otherwise.

For details about how entry IDs are used to identify objects in the address book, see [\[MS-OXNSPI\]](#) section 3.1.4.6.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.10.2 ModLinkAtt Request Type Success Response Body

The **ModLinkAtt** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															

AuxiliaryBufferSize
AuxiliaryBuffer (variable)
...

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.10.3 ModLinkAtt Request Type Failure Response Body

The **ModLinkAtt** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.11 ModProps Request Type

The **ModProps** request type is used by the client to modify the specified properties of an Address Book object. For information regarding the server-side processing of the **ModProps** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.14.

2.2.5.11.1 ModProps Request Type Request Body

The **ModProps** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
HasState										State																					
...																															
(State field continues for 7 rows)																															
...										HasPropsToRemove										PropertiesToRemove (variable)											
...																															
HasPropertyValues										PropertyValues (variable)																					
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasPropertyTags (1 byte): A Boolean value that specifies whether the **PropertyTags** field is present.

PropertyTags (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the properties to be removed. This field is present when the value of the **HasPropertyTags** field is nonzero and is not present otherwise.

HasPropertyValues (1 byte): A Boolean value that specifies whether the **PropertyValues** field is present.

PropertyValues (variable): An **AddressBookPropertyValueList** structure (section [2.2.1.1](#)) that specifies the values of the properties to be modified. This field is present when the value of the **HasPropertyValues** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.11.2 ModProps Request Type Success Response Body

The **ModProps** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.11.3 ModProps Request Type Failure Response Body

The **ModProps** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.12 QueryRows Request Type

The **QueryRows** request type is used by the client to get a number of rows from the specified Explicit Table. For details about Explicit Tables, see [\[MS-OXNSPI\]](#) section 3.1.4.4.2. For information regarding the server-side processing of the **QueryRows** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.8.

Although the protocol places no boundary or requirements on the minimum number of rows the server returns, implementations SHOULD return as many rows as possible to improve usability of the server for clients.

2.2.5.12.1 QueryRows Request Type Request Body

The **QueryRows** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
HasState																State															
...																															
(State field continues for 7 rows)																															
...																ExplicitTableCount															
...																ExplicitTable (variable)															
...																															
RowCount																															
HasColumns																Columns (variable)															
...																															
AuxiliaryBufferSize																															

AuxiliaryBuffer (variable)
...

Flags (4 bytes): A set of bit flags that specify options to the server. The server MUST ignore values other than the bit flags **fEphID** (0x00000002) and **fSkipObjects** (0x00000001).

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

ExplicitTableCount (4 bytes): An unsigned integer that specifies the number of structures present in the **ExplicitTable** field. This value is limited to 100,000.

ExplicitTable (variable): An array of **MinimalEntryID** structures ([\[MS-OXNSPI\]](#) section 2.2.9.1) that constitute the Explicit Table.

RowCount (4 bytes): An unsigned integer that specifies the number of rows the client is requesting.

HasColumns (1 byte): A Boolean value that specifies whether the **Columns** field is present.

Columns (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the properties that the client requires for each row returned. This field is present when the value of the **HasColumns** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.12.2 QueryRows Request Type Success Response Body

The **QueryRows** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
HasState										State																					
...																															
(State field continues for 7 rows)																															
...										HasColsAndRows										Columns (variable)											

...
RowCount
RowData (variable)
...
AuxiliaryBufferSize
AuxiliaryBuffer (variable)
...

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasColsAndRows (1 byte): A Boolean value that specifies whether the **Columns**, **RowCount**, and **RowData** fields are present.

Columns (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the columns for the returned rows. This field is present when the value of the **HasColsAndRows** field is nonzero and is not present otherwise.

RowCount (4 bytes): An unsigned integer that specifies the number of structures in the **RowData** field. This field is present when the value of the **HasColsAndRows** field is nonzero and is not present otherwise.

RowData (variable): An array of **AddressBookPropertyRow** structures (section [2.2.1.2](#)), each of which specifies the row data of the Explicit Table. This field is present when the **HasColsAndRows** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.12.3 QueryRows Request Type Failure Response Body

The **QueryRows** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.13 QueryColumns Request Type

The **QueryColumns** request type is used by the client to get a list of all of the properties that exist in the address book. For information regarding the server-side processing of the **QueryColumns** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.5.

2.2.5.13.1 QueryColumns Request Type Request Body

The **QueryColumns** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
MapiFlags																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

MapiFlags (4 bytes): A set of bit flags that specify options to the server. The server MUST ignore values other than the bit flag **NspiUnicodeProptypes** (0x80000000).

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.13.2 QueryColumns Request Type Success Response Body

The **QueryColumns** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
HasColumns										Columns (variable)																					
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

HasColumns (1 byte): A Boolean value that specifies whether the **Columns** field is present.

Columns (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the properties that exist on the address book. This field is present when the **HasColumns** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.13.3 QueryColumns Request Type Failure Response Body

The **QueryColumns** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.14 ResolveNames Request Type

The **ResolveNames** request type is used by the client to perform **ambiguous name resolution (ANR)**, as specified in [\[MS-OXNSPI\]](#) section 3.1.4.7. For information regarding the server-side processing of the **ResolveNames** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.16

2.2.5.14.1 ResolveNames Request Type Request Body

The **ResolveNames** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
HasState																State															
...																															
(State field continues for 7 rows)																															
...																HasPropertyTags										PropertyTags (variable)					
...																															
HasNames																Names (variable)															

...
AuxiliaryBufferSize
AuxiliaryBuffer (variable)
...

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasPropertyTags (1 byte): A Boolean value that specifies whether the **PropertyTags** field is present.

PropertyTags (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the properties that client requires for the rows returned. This field is present when the value of the **HasPropertyTags** field is nonzero and is not present otherwise.

HasNames (1 byte): A Boolean value that specifies whether the **Names** field is present.

Names (variable): A **WStringsArray_r** structure ([\[MS-OXNSPI\]](#) section 2.2.7.2) that specifies the values on which the client is requesting that the server perform ANR. This field is present when the value of the **HasNames** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.14.2 ResolveNames Request Type Success Response Body

The **ResolveNames** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
StatusCode																															
ErrorCode																															
CodePage																															
HasMinimalIds										MinimalIdCount																					

...	MinimalIds (variable)
...	
HasRowsAndCols	PropertyTags (variable)
...	
RowCount	
RowData (variable)	
...	
AuxiliaryBufferSize	
AuxiliaryBuffer (variable)	
...	

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

CodePage (4 bytes): An unsigned integer that specifies the code page the server used to express string values of properties.

HasMinimalIds (1 byte): A Boolean value that specifies whether the **MinimalIdCount** and **MinimalIds** fields are present.

MinimalIdCount (4 bytes): An unsigned integer that specifies the number of structures in the **MinimalIds** field. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

MinimalIds (variable): An array of **MinimalEntryID** structures ([\[MS-OXNSPI\]](#) section 2.2.9.1), each of which specifies a Minimal Entry ID matching a name requested by the client. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

HasRowsAndCols (1 byte): A Boolean value that specifies whether the **PropertyTags**, **RowCount**, and **RowData** fields are present.

PropertyTags (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the properties returned for the rows in the **RowData** field. This field is present when the value of the **HasRowsAndCols** field is nonzero and is not present otherwise.

RowCount (4 bytes): An unsigned integer that specifies the number of structures in the **RowData** field. This field is present when the value of the **HasRowsAndCols** field is nonzero and is not present otherwise.

RowData (variable): An array of **AddressBookPropertyRow** structures (section [2.2.1.2](#)), each of which specifies the row data requested. This field is present when the value of the **HasRowsAndCols** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.14.3 ResolveNames Request Type Failure Response Body

The **ResolveNames** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.15 ResortRestriction Request Type

The **ResortRestriction** request type is used by the client to sort the objects in a restricted address book container. For information regarding the server-side processing of the **ResortRestriction** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.11.

2.2.5.15.1 ResortRestriction Request Type Request Body

The **ResortRestriction** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
HasState																State															

...		
(State field continues for 7 rows)		
...	HasMinimalIds	MinimalIdCount
...		MinimalIds (variable)
...		
AuxiliaryBufferSize		
AuxiliaryBuffer (variable)		
...		

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasMinimalIds (1 byte): A Boolean value that specifies whether the **MinimalIdCount** and **MinimalIds** fields are present.

MinimalIdCount (4 bytes): An unsigned integer that specifies the number of structures in the **MinimalIds** field. This field is present when the value of the **HasMinimalIds** field is present and is not present otherwise.

MinimalIds (variable): An array of **MinimalEntryID** structures ([\[MS-OXNSPI\]](#) section 2.2.9.1) that compose a restricted address book container. The number of structures contained in this field is specified by the **MinimalIdCount** field. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.15.2 ResortRestriction Request Type Success Response Body

The **ResortRestriction** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															

ErrorCode		
HasState	State	
...		
(State field continues for 7 rows)		
...	HasMinimalIds	MinimalIdCount
...		MinimalIds (variable)
...		
AuxiliaryBufferSize		
AuxiliaryBuffer (variable)		
...		

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasMinimalIds (1 byte): A Boolean value that specifies whether the **MinimalIdCount** and **MinimalIds** fields are present.

MinimalIdCount (4 bytes): An unsigned integer that specifies the number of structures present in the **Minimalids** field. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

MinimalIds (variable): An array of **MinimalEntryID** structures ([\[MS-OXNSPI\]](#) section 2.2.9.1) that compose a restricted address book container. The number of structures contained in this field is specified by the **MinimalIdCount** field. This field is present when the value of the **HasMinimalIds** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.15.3 ResortRestriction Request Type Failure Response Body

The **ResortRestriction** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.16 SeekEntries Request Type

The **SeekEntries** request type is used by the client to search for and set the logical position in a specific table to the first entry greater than or equal to a specified value. Optionally, the **SeekEntries** request type can also be used to retrieve information about rows in an Explicit Table. For details about Explicit Tables, see [\[MS-OXNSPI\]](#) section 3.1.4.4.2. For information regarding the server-side processing of the **SeekEntries** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.9.

2.2.5.16.1 SeekEntries Request Type Request Body

The **SeekEntries** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
HasState																State															
...																															
(State field continues for 7 rows)																															
...																HasTarget								Target (variable)							

...	
HasExplicitTable	ExplicitTableCount
...	ExplicitTable (variable)
...	
HasColumns	Columns (variable)
...	
AuxiliaryBufferSize	
AuxiliaryBuffer (variable)	
...	

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasTarget (1 byte): A Boolean value that specifies whether the **Target** field is present.

Target (variable): A **PropertyValue_r** structure ([\[MS-OXNSPI\]](#) section 2.2.2.12) that specifies the property value being sought. This field is present when the value of the **HasTarget** field is nonzero and is not present otherwise.

HasExplicitTable (1 byte): A Boolean value that specifies whether the **ExplicitTableCount** and **ExplicitTable** fields are present.

ExplicitTableCount (4 bytes): An unsigned integer that specifies the number of structures present in the **ExplicitTable** field. The number is limited to 100,000. This field is present when the value of the **HasExplicitTable** field is nonzero and is not present otherwise.

ExplicitTable (variable): An array of **MinimalEntryID** structures ([\[MS-OXNSPI\]](#) section 2.2.9.1) that constitute an Explicit Table. This field is present when the value of the **HasExplicitTable** field is nonzero and is not present otherwise.

HasColumns (1 byte): A Boolean value that specifies whether the **Columns** field is present.

Columns (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the columns that the client is requesting. This field is present when the value of the **HasColumns** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.16.2 SeekEntries Request Type Success Response Body

The **SeekEntries** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
HasState										State																					
...																															
(State field continues for 7 rows)																															
...										HasColsAndRows										Columns (variable)											
...																															
RowCount																															
RowData (variable)																															
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasColsAndRows (1 byte): A Boolean value that specifies whether the **Columns**, **RowCount**, and **RowData** fields are present.

Columns (variable): A **LargePropertyTagArray** structure (section [2.2.1.3](#)) that specifies the columns used for the rows returned. This field is present when the value of the **HasColsAndRows** field is nonzero and is not present otherwise.

RowCount (4 bytes): An unsigned integer that specifies the number of structures contained in the **RowData** field. This field is present when the value of the **HasColsAndRows** field is nonzero and is not present otherwise.

RowData (variable): An array of **AddressBookPropertyRow** structures (section [2.2.1.2](#)), each of which specifies the row data for the entries queried. This field is present when the **HasColsAndRows** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.16.3 SeekEntries Request Type Failure Response Body

The **SeekEntries** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.17 UpdateStat Request Type

The **UpdateStat** request type is used by the client to update the **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) to reflect the client's changes. For information regarding the server-side processing of the **UpdateStat** request type, see Server Processing Rules in [\[MS-OXNSPI\]](#) section 3.1.4.1.4.

2.2.5.17.1 UpdateStat Request Type Request Body

The **UpdateStat** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reserved																															
HasState										State																					
...																															
(State field continues for 7 rows)																															
...										DeltaRequested										AuxiliaryBufferSize											
...										AuxiliaryBuffer (variable)																					
...																															

Reserved (4 bytes): Reserved. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

DeltaRequested (1 byte): A Boolean value that specifies whether the client is requesting a value to be returned in the **Delta** field of the response. The value zero (0x00) indicates that the value is not requested. A nonzero value indicates that the value is requested.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.17.2 UpdateStat Request Type Success Response Body

The **UpdateStat** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
HasState										State																					
...																															

(State field continues for 7 rows)		
...	HasDelta	Delta
...		AuxiliaryBufferSize
...		AuxiliaryBuffer (variable)
...		

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

HasState (1 byte): A Boolean value that specifies whether the **State** field is present.

State (36 bytes): A **STAT** structure ([\[MS-OXNSPI\]](#) section 2.2.8) that specifies the state of a specific address book container. This field is present when the **HasState** field is nonzero and is not present otherwise.

HasDelta (1 byte): A Boolean value that specifies whether the **Delta** field is present.

Delta (4 bytes): A signed integer that specifies the movement within the address book container that was specified in the **State** field of the request. This field is present when the value of the **HasDelta** field is nonzero and is not present otherwise.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.17.3 UpdateStat Request Type Failure Response Body

The **UpdateStat** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.18 GetMailboxUrl Request Type

The **GetMailboxUrl** request type is used by the client to get the Uniform Resource Locator (URL) of the specified mailbox server endpoint (4). For information regarding the server-side processing of the **GetMailboxUrl** request type, see [\[MS-OXABREF\]](#) section 3.1.4.2.

2.2.5.18.1 GetMailboxUrl Request Type Request Body

The **GetMailboxUrl** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
ServerDn (variable)																															
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Flags (4 bytes): Not used. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

ServerDn (variable): A null-terminated Unicode string that specifies the distinguished name (DN) (1) of the mailbox server for which to look up the URL.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.18.2 GetMailboxUrl Request Type Success Response Body

The **GetMailboxUrl** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
ServerUrl (variable)																															
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

ServerUrl (variable): A null-terminated Unicode string that specifies URL of the **EMSMDB** server.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.18.3 GetMailboxUrl Request Type Failure Response Body

The **GetMailboxUrl** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.19 GetAddressBookUrl Request Type

The **GetAddressBookUrl** request type is used by the client to get the URL of the specified address book server endpoint (4). For information regarding the server-side processing of the **GetAddressBookUrl** request type, see [\[MS-OXABREF\]](#) section 3.1.4.1.

2.2.5.19.1 GetAddressBookUrl Request Type Request Body

The **GetAddressBookUrl** request type request body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
UserDn (variable)																															
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

Flags (4 bytes): Not used. The client MUST set this field to 0x00000000 and the server MUST ignore this field.

UserDn (variable): A null-terminated Unicode string that specifies the distinguished name (DN) (1) of the user's mailbox. This DN (1) is used by the server to determine which NSPI server is used.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data sent from the client. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.19.2 GetAddressBookUrl Request Type Success Response Body

The **GetAddressBookUrl** request type success response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
ErrorCode																															
ServerUrl (variable)																															
...																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST be set to 0x00000000.

ErrorCode (4 bytes): An unsigned integer that specifies the return status of the operation. The error code values are specified in [\[MS-OXNSPI\]](#) section 2.2.1.2.

ServerUrl (variable): A null-terminated Unicode string that specifies the URL of the NSPI server.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.5.19.3 GetAddressBookUrl Request Type Failure Response Body

The **GetAddressBookUrl** request type failure response body contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
StatusCode																															
AuxiliaryBufferSize																															
AuxiliaryBuffer (variable)																															
...																															

StatusCode (4 bytes): An unsigned integer that specifies the status of the request. This field MUST NOT be set to 0x00000000.

AuxiliaryBufferSize (4 bytes): An unsigned integer that specifies the size, in bytes, of the **AuxiliaryBuffer** field.

AuxiliaryBuffer (variable): An array of bytes that constitute the auxiliary payload data returned from the server. The size of this field, in bytes, is specified by the **AuxiliaryBufferSize** field.

2.2.6 PING Request Type

The **PING** request type allows a client to determine whether a server's endpoint (4) is reachable and operational. The **PING** request type has no request body and no response body.

The **PING** request type is supported by both the mailbox server endpoint (4) and the address book server endpoint (4). For details about these endpoints (4), see section [2.2.4](#) and section [2.2.5](#), respectively.

2.2.7 Response Meta-Tags

The protocol defines three meta-tags that are used to inform the client as to the state of processing a request on the server. A meta-tag is returned at the beginning of the response body. For details, see section [3.2.5.2.1](#).

The following table specifies the three meta-tags that are used by this protocol.

Meta-tag	Meaning
PROCESSING	The server has queued the request to be processed.
PENDING	The server is processing the request.
DONE	The server has completed the processing of the request and additional response headers and the response body follow the DONE meta-tag.

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 specification does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this specification.

Cookies: All cookies returned to the client **MUST** be saved during the life of the Session Context.

3.1.2 Timers

An idle-time timer measures the time during which the Session Context has no pending requests.

3.1.3 Initialization

The client **MUST** establish a Session Context, as specified in section [3.1.5.1](#).

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 Creating a Session Context by Using the Connect or Bind Request Type

The client establishes a Session Context in the **EMSMDB** interface by issuing a request, as specified in section [2.2.2.1](#), using the **Connect** request type as specified in section [2.2.3.3.1](#), including the contents of the **Connect** request type request body, as specified in section [2.2.4.1.1](#). The client establishes a Session Context in the **NSPI** interface by issuing a request, as specified in section [2.2.2.1](#), using the **Bind** request type as specified in section [2.2.5.1](#), including the contents of the **Bind** request type request body, as specified in section [2.2.5.1.1](#). An example of using the **Connect** request type to establish a new Session Context is given in section [4.1](#).

The client can pass an **X-ClientInfo** header, as specified in section [2.2.3.3.4](#). This information is simply returned to the client in the response.

As specified in section [3.2.5.1](#), the server returns cookies used to identify the Session Context that has been created. The client **MUST** store all returned cookies and associate them with the Session Context. The client **MUST** include all cookies received from the previous call to the server for a given Session Context when issuing the next request to the server for that Session Context. If the server uses a session sequence cookie to guarantee the sequencing of requests, the client **MUST** pass this cookie, along with the session context cookie, to the server on the next request. The **Cookie** header, specified in section [2.2.3.2.4](#), is used to pass the cookies.

3.1.5.2 Sending Remote Operations by Using the Execute Request Type

The client submits ROPs, as specified in [\[MS-OXCROPS\]](#), by sending a request as specified in section [2.2.2.1](#), using the **Execute** request type as specified in section [2.2.3.3.1](#), including an **Execute**

request type request body as specified in section [2.2.4.2.1](#). Before using the **Execute** request type, the client MUST have created a valid Session Context on the server as specified in section [3.1.5.1](#).

The client MUST include all cookies received from the previous call to the server for a given Session Context when issuing the next request to the server for that Session Context. The **Cookie** header, specified in section [2.2.3.2.4](#), is used to pass the cookies.

A scenario that describes issuing ROP commands using the **Execute** request type is presented in section [4.2](#).

3.1.5.3 Keeping a Session Context Alive by Using the PING Request Type

Since the Session Context will automatically expire on the server after a period of inactivity, the client MUST keep the Session Context alive. The client can do this by issuing a **PING** request type request, as specified in section [2.2.6](#). The client MUST pass all of the session context cookies that are returned from the server in the previous response. Additional cookies, such as the return sequence cookie, are also passed if available.

The client gets verification that the Session Context idle-time timer (section [3.1.2](#)) was refreshed when it receives a successful **PING** response containing the **X-ExpirationInfo** header. The client SHOULD allow for network latency and issue a **PING** request type request at an interval that is less than what is specified in the **X-ExpirationInfo** header.

3.1.5.4 Destroying a Session Context by Using the Disconnect or Unbind Request Type

The client destroys the Session Context by sending a request using the **Disconnect** request type, as specified in section [2.2.4.3.1](#), or the **Unbind** request type, as specified in section [2.2.5.2.1](#). Destroying the Session Context releases all state information associated with the Session Context. The **Disconnect** or **Unbind** request type request MUST include the session context cookie returned when the Session Context was created, plus any other cookies returned by the server while the Session Context is active.

The client MUST NOT consider the Session Context destroyed until it receives a successful response to the **Disconnect** or **Unbind** request, as specified in section [2.2.4.3.2](#) and section [2.2.5.2.2](#), respectively.

3.1.5.5 Requesting Notification by Using the NotificationWait Request Type

A request, as specified in section [2.2.2.1](#), that uses the **NotificationWait** request type, as specified in section [2.2.3.3.1](#), with the associated **NotificationWait** request body, as specified in section [2.2.4.4.1](#), creates an operation on the server that will not complete until events pending on the Session Context complete, or at the end of a 5-minute duration during which there has been no event activity on the Session Context.

Unlike other request types, the client can send this request to the server while another request is pending (such as an **Execute** request type).

This request type is part of notification handling. For more information about notifications, see [\[MS-OXCNOTIF\]](#).

3.1.5.6 Handling a Chunked Response

To facilitate a positive connection between the server and client, the server uses the **Transfer-Encoding** header, as specified in section [2.2.3.2.5](#), with "chunked" transfer encoding, as specified

in [\[RFC2616\]](#). By using "chunked" transfer encoding, the server is able to return data to the client while the request is still being processed. This gives the client the expectation of always getting an immediate acknowledgment of the request. If the client doesn't get this immediate acknowledgment, it can abandon the request and recover if the server's initial response is not received within a reasonable time.

After the initial immediate response, the client periodically receives a keep-alive response indicating that the server is still processing the request. The keep-alive response contains the **PENDING** meta-tag. As with the client's ability to quickly detect whether a request was received by the server, the client can abandon a request if no periodic keep-alive response is received within a reasonable amount of time. A secondary advantage of receiving the periodic keep-alive data is that the underlying HTTP connection remains open. This is particularly useful when there are devices in the middle that might be prone to timing out long-running requests.

The immediate response includes an **X-PendingPeriod** header, specified in section [2.2.3.3.5](#), to tell the client the number of milliseconds to be expected between keep-alive **PENDING** meta-tags in the response stream during the time a request is currently being processed on the server. The default value for the keep-alive interval is 15 seconds, until the request is done.

The client **MUST** be able to receive a "chunked" response for any request and know how to parse the chunked transfer encoding and work with the inner response stream (the response body) as if the chunked transfer encoding wasn't present. The initial response, plus the intermediate keep-alive transmissions, and the final response body are all part of the inner response stream. The use of "chunked" transfer encoding is a means to return an unknown amount of data to the client. The client **MUST** isolate the receiving of response "chunks" from the parsing and interpreting of the inner response stream. For more details about the "chunked" response, see section [3.2.5.2](#).

The inner response stream contains response meta-tags, as specified in section [2.2.7](#), to be used by the client in interpreting where the server is in the process of completing the request, the keep-alive messages, and finally the response body. For more details about the use of meta-tags, see section [3.2.5.3.4](#).

3.1.5.7 Reconnecting and Establishing a New Session Context

To re-establish a Session Context after a failure, a prolonged period of inactivity, or a forced action by the user, the client sends a new request using the **Connect** request type, along with the **Connect** request type request body, as specified in section [2.2.4.1.1](#), or by using the **Bind** request type, along with the **Bind** request type request body, as specified in section [2.2.5.1.1](#). The only difference between reconnecting and an initial connection is that the client passes all existing cookies that are associated with the previous Session Context that the client is attempting to re-establish. If the client reconnects, it **SHOULD** always pass any cookie values it has stored for the Session Context to which it is attempting to reconnect. A client can do this if the end user forcefully reconnects. This allows the server to clean up the previous context in a timely fashion to prevent session limits from being reached. The reconnection request will look very much like a request to establish a new Session Context, with the exception of passing all existing cookies that are associated with the previous Session Context that the client is attempting to re-establish.

Since the client is not aware of the semantic meaning of the cookies, it **MUST** pass all cookies that it has that relate to the specific Session Context. For details about the server response, see section [3.2.5.6](#).

Before the client attempts to reconnect, the client **SHOULD** always assume the Session Context is still valid. If it is unable to communicate with the server, no matter how much time has passed, when it finally re-establishes an HTTP connection the client **SHOULD** continue where it left off.

3.1.6 Timer Events

The client can choose to keep the Session Context alive by sending a **PING** request type request, as specified in section [2.2.3.3.1](#), before the idle-time timer specified in section [3.1.2](#) expires.

3.1.7 Other Local Events

Since a request or response can fail, the client and server need to be able to handle communication failures so that they do not get into an inconsistent state. A majority of failures are expected to occur as the client is attempting to send a new request after the server has dropped the connection because of the expiration of the idle-time timer (section [3.1.2](#)). Under any circumstances, the client **MUST** be able to properly handle failures during all aspects of sending and receiving a request or response.

3.1.7.1 Handling Communication Failure Sending a Request

If the client attempts to send a new request to the server and a failure occurs either before streaming the request body or while streaming the request body, the client establishes a new HTTP connection, passing all existing cookies for the Session Context, and then resubmits the request as-is to the server. The client does not need to re-establish a new Session Context in this case.

3.1.7.2 Handling Communication Failure Reading a Response

If the client receives a failure while attempting to read the response after the entire request, including the request body, is streamed to the server, the client reconnects with the server and establishes a new Session Context, passing all existing cookies for the Session Context that is being reconnected. For more details, see section [3.1.5.7](#).

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 specification does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this specification.

Cookies: All cookies returned to the client **MUST** be saved during the life of the Session Context.

3.2.2 Timers

A keep-alive interval timer sets the time when the server returns a keep-alive response to the client while a request is being processed. The maximum interval is configurable, and the default is set to 15 seconds.

A notification timer triggers the server to complete a **NotificationWait** request after 5 minutes of no event activity on the Session Context. For details about how the server responds to a **NotificationWait** request, see section [3.2.5.5](#).

An idle-time timer measures the time during which the Session Context has no pending requests.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

3.2.5.1 Responding to a Connect or Bind Request Type Request

The server issues a response, as specified in section [2.2.2.2](#), to a **Connect** or **Bind** request type request. The server creates a new Session Context and associates it with a session context cookie. If successful, the server's response includes the **Connect** request type success response body, as specified in section [2.2.4.1.2](#), or the **Bind** request type success response body, as specified in section [2.2.5.1.2](#); if unsuccessful, the server's response includes the **Connect** request type failure response body, as specified in section [2.2.4.1.3](#), or the or **Bind** request type failure response body, as specified in section [2.2.5.1.3](#).

The server MUST return the cookie that represents the Session Context as the value of the **Set-Cookie** header field, as specified in section [2.2.3.2.3](#). The names of the cookies are implementation- specific.

The server MUST store the authentication context with the newly created Session Context. The server then MUST validate the authentication context on subsequent requests against the Session Context. If the authentication context differs, the server MUST fail the request with a value of 10 ("Context Not Found" error) in the **X-ResponseCode** header. For details about the **X-ResponseCode** header, see section [2.2.3.3.3](#).

The server MUST insure that the client issues only one request at a time within a Session Context. If the server detects that the client has issued simultaneous requests within a Session Context, the server MUST fail every subsequent request with a value of 15 ("Invalid Sequence" error) in the **X-ResponseCode** header.

3.2.5.2 Responding to All Request Type Requests

The server can respond to all request type requests with a chunked response, as specified in section [2.2.2.2](#). If the entire response is not readily available, the server MUST use the **Transfer-Encoding** header, as specified in section [2.2.3.2.5](#), with a value of "chunked". Chunked transfer coding is specified in [\[RFC2616\]](#) section 3.6.1. The **Transfer-Encoding** header is used instead of the **Content-Length** header field as specified in section [2.2.3.2.1](#). By using "chunked" transfer coding, the server is able to return data to the client while the request is still being processed by the server. It also allows the server to return an amount of data to the client whose length is not determined when the initial response is returned. If the server is able to return the entire response immediately, it can chose to forgo the chunked response.

The response includes all **Set-Cookie** headers as specified in section [2.2.3.2.3](#) associated with the Session Context.

After the request has been authorized, authenticated, parsed, and validated, the server MUST return a **X-ResponseCode** with a value of 0 (zero) to indicate that the request has been accepted. The server MUST respond immediately to a request while the request is being queued and processed by the server. The initial response includes the **PROCESSING** meta-tag, as specified in section [2.2.7](#). If the server is using the "chunked" transfer coding, it MUST flush this to the client (being careful to make sure it disables any internal Nagle algorithms, as described in [\[RFC896\]](#), that might attempt to buffer response data).

Beyond the initial immediate response, the server MUST also periodically return a keep-alive response to the client to let the client know that the request is still being processed. The server maintains a keep-alive timer as specified in section [3.2.2](#) that, when expired, causes the server to send a keep-alive response to the client. The keep-alive response includes the **PENDING** meta-tag, as specified in section [2.2.7](#). Since the keep-alive interval is configurable or auto-adjusted, the server MUST return the **X-PendingPeriod** header, specified in section [2.2.3.3.3](#), within the immediate response to tell the client the number of milliseconds to be expected between keep-alive responses from the server during the time a request is currently being executed on the server. The default value of the **X-PendingPeriod** header is 15000 for 15 seconds.

After the server finishes processing the request it finishes with the **DONE** meta-tag, as specified in section [2.2.7](#); followed by any additional response headers. The **X-ElapsedTime** and **X-StartTime** are present after the DONE meta-tag, as specified in section [2.2.3.3.9](#) and [2.2.3.3.10](#) respectively. The server will include a **X-ResponseCode** header, as specified in section [2.2.3.3.3](#), if a failure occurs after the initial response is sent. More specifically, this gives the server the ability to later fail the request and return a different value in the **X-ResponseCode** header. The request type response body for the initiating request type will immediately follow any additional headers preceded by a CRLF on an empty line. If an additional **X-ResponseCode** header is returned and if it indicates a failure, then the response body can be empty or can include failure data in a format that is specified by an additional **Content-Type** header as specified in section [2.2.3.2.2](#).

An example of the request and response using the **Execute** request type is described in section [4.2](#).

3.2.5.3 Responding to a PING Request Type

The server responds to a **PING** request type by returning a **PING** request type response, as specified in section [2.2.6](#). The server MUST refresh the idle-time timer specified in section [3.2.2](#) associated with the Session Context. Meta-tags can be returned, but no response body is returned.

3.2.5.4 Responding to a Disconnect or Unbind Request Type Request

The server sends a response, as specified in section [2.2.2.2](#), to a **Disconnect** or **Unbind** request type request. The server releases all state information associated with the Session Context and invalidates the associated session context cookie passed in the request. If successful, the server's response includes the **Disconnect** request type success response body, as specified in section [2.2.4.3.2](#), or the **Unbind** request type success response body, as specified in section [2.2.5.2.2](#); if unsuccessful, the server's response includes the **Disconnect** request type failure response body, as specified in section [2.2.4.3.3](#), or the or **Unbind** request type failure response body, as specified in section [2.2.5.2.3](#).

Once the session context cookie has been invalidated by the server, the client cannot use it in subsequent requests. If the client attempts to use an invalid session context cookie in a request, the server MUST fail the request to indicate to the client that the Session Context is not valid.

The server's response includes the **Content-Length** header, as specified in section [2.2.3.2.1](#), if the response is not chunked. All relevant **Set-Cookie** headers (section [2.2.3.2.4](#)) are included in the response even though the session context cookie has been invalidated.

3.2.5.5 Responding to a NotificationWait Request Type Request

The server creates a **NotificationWait** request type response, as specified in section [2.2.2.2](#), including the **NotificationWait** request type response body as specified in section [2.2.4.4.2](#) if the request was successful, or the response body specified in section [2.2.4.4.3](#) if unsuccessful. This response is not sent until either the current server event completes or the 5-minute maximum time limit expires. The server MUST keep the Session Context alive for the entire duration of the

NotificationWait request and MUST refresh the Session Context expiration upon completion of the **NotificationWait**.

The response headers include **Set-Cookie** headers as specified in section [2.2.3.2.4](#) for all cookies related to the Session Context. The response also includes the **Content-Length** header, as specified in section [2.2.3.2.1](#), if the response is not chunked.

3.2.5.6 Reconnecting and Establishing a New Session Context Server

When the connection between client and server is dropped for whatever reason, the client reconnects with the existing Session Context by sending a **Connect** or **Bind** request that includes the existing session cookies saved by the client, as specified in section [2.2.4.1.1](#). If the previous Session Context is still valid on the server, the server MUST destroy it before creating a new Session Context. The server returns a new session context cookie that is associated with a new Session Context. The server MUST ignore a sequence validation cookie passed in the reconnect scenario. The response from the server uses the **Set-Cookie** header, as specified in section [2.2.3.2.3](#), to pass any required cookies to the client. The response passes the **Content-Length** header, as specified in section [2.2.3.2.1](#).

As with any new **Session Context**, the client MUST store all returned cookies and MUST NOT comingle the new cookies with cookies from the previous Session Context.

If the Session Context has expired, is no longer valid, or is not valid for the server in which the mailbox currently resides, then the server fails the request with an **X-ResponseCode** value of 10, as specified in section [2.2.3.3.3](#), and the client SHOULD reconnect and establish a new Session Context.

3.2.6 Timer Events

The idle-time timer, as specified in section [3.2.2](#), starts counting when the server has no pending requests for the Session Context. When the idle-time timer reaches the configured time limit, the server destroys the Session Context, deleting all cookies associated with the Session Context.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 Establish a New Session Context

In this scenario the client establishes a new Session Context with the server before sending or receiving emails. The client sends a request (section [2.2.2.1](#)) using an **X-RequestType** header field value of **Connect**, as described in section [2.2.3.3.1](#), and includes the **Connect** request type request body as described in section [2.2.4.1.1](#).

Client request

```
POST <Autodiscover-provided endpoint> HTTP/1.1
Host: <URL of the host server>
Content-Length: <length of REQUEST BODY>
Content-Type: application/mapi-http
X-RequestType: Connect
X-ClientInfo: <opaque string>
X-RequestId: <unique identifier>
X-ClientApplication: <client version>

<REQUEST BODY>
```

The server processes the request and returns a response, as described in section [2.2.2.2](#), that includes the session context cookie that identifies the new Session Context, and the **Connect** request type success response body, as described in section [2.2.4.1.2](#).

Server response

```
HTTP/1.1 200 OK
Content-Length: <length of META-TAGS, ADDITIONAL HEADERS and RESPONSE BODY>
Content-Type: application/mapi-http
Set-Cookie: <session context cookie>=<opaque string>
Set-Cookie: <request sequence cookie>=<opaque string>
X-PendingPeriod: 15000
X-RequestType: Connect
X-RequestId: <unique identifier>
X-ResponseCode: 0
X-ClientInfo: <opaque string>
X-ServerApplication: <server version>
X-ExpirationInfo: <milliseconds>

PROCESSING<CLRF>
DONE<CRLF>
X-ResponseCode: 0
X-ElapsedTime: <milliseconds>
X-StartTime: <date/time>
<CRLF>
<RESPONSE BODY>
```

4.2 Issue ROP Commands to the Server

In this scenario the client sends a request, as described in section [2.2.2.1](#), containing ROPs to the server using the **Execute** request type, as described in section [2.2.3.3.1](#), including a <REQUEST BODY> as described in section [2.2.4.2.1](#). As described in section [3.1.5.2](#), the request includes all the cookies related to the Session Context that the client has saved.

Client request

```
POST <Autodiscover-provided endpoint> HTTP/1.1
Host: <URL of the host server>
Content-Length: <length of REQUEST BODY>
Content-Type: application/mapi-http
Cookie: <session context cookie>=<opaque string>
Cookie: <return sequence cookie>=<opaque string>
X-RequestType: Execute
X-ClientInfo: <opaque string>
X-RequestId: <unique identifier>
X-ClientApplication: <client version>

<REQUEST BODY>
```

The response to an **Execute** request type is "chunked", as described in section [3.2.5.2](#). The initial response contains the **PROCESSING** meta-tag, as described in section [2.2.7](#). Depending on how long the processing takes, the initial response is followed by interim "keep-alive" responses containing the **PENDING** meta-tag. When the processing is completed, the final chunk contains the **DONE** meta-tag; the **X-ResponseCode** value, as described in section [2.2.3.3.3](#), for the completed response; and the **Execute** request type request body.

Server response

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
Content-Type: application/mapi-http
Cookie: MapiContext=<opaque string>
Cookie: MapiSequence=<opaque string>
X-PendingPeriod: 15000
X-RequestType: Execute
X-RequestId: <unique identifier>
X-ResponseCode: 0
X-ClientInfo: <opaque string>
X-ServerApplication: <server version>
X-ExpirationInfo: <milliseconds>
<CRLF>
C<CRLF>
PROCESSING<CRLF>
<CRLF>
```

Every 15 seconds, as indicated by the value of the **X-PendingPeriod** header described in section [2.2.3.3.5](#), the server returns another keep-alive chunk:

```
9<CRLF>
PENDING<CRLF>
<CRLF>
```

When the request finally completes, the server returns the **DONE** meta-tag, followed by the <RESPONSE BODY>. The final character is "0", which is the last-chunk indicator, as described in [\[RFC2616\]](#) section 3.6.1.

```
?<CRLF>
DONE<CRLF>
```

```
X-ResponseCode: 0<CRLF>
X-ElapsedTime: <milliseconds>
X-StartTime: <date/time>
<CRLF>
<RESPONSE BODY><CRLF>
<CRLF>
0<CRLF>
<CRLF>
```

4.3 Refresh an Idle Session Context

This scenario describes refreshing an idle Session Context. To keep the server from timing out the Session Context, the client issues a **PING** request type, as described in section [2.2.6](#). Client creation of the request is described in section [3.1.5.3](#). Server handling of the response is described in section [3.2.5.3](#).

Client request

```
POST <Autodiscover-provided endpoint> HTTP/1.1
Host: <URL of the host server>
Content-Length: 0
Content-Type: application/mapi-http
Cookie: <session context cookie>=<opaque string>
Cookie: <return sequence cookie>=<opaque string>
X-RequestType: PING
X-ClientInfo: <opaque string>
X-RequestId: <unique identifier>
X-ClientApplication: <client version>
```

Server response

```
HTTP/1.1 200 OK
Content-Length: <length of META-TAGS, ADDITIONAL HEADERS>
Content-Type: application/mapi-http
X-RequestType: PING
X-RequestId: <unique identifier>
X-ResponseCode: 0
X-ClientInfo: <opaque string>
X-ServerApplication: <server version>
X-ExpirationInfo: <milliseconds>

PROCESSING<CRLF>
DONE<CRLF>
X-ResponseCode: 0<CRLF>
X-ElapsedTime: <milliseconds>
X-StartTime: <date/time>
<CRLF>
```

4.4 Re-Establish a Timed-Out Connection to the Server

This scenario describes re-establishing a timed-out Session Context. This is similar to the process of establishing a new Session Context as described in section [3.1.5.1](#), but **Cookie** headers, as described in section [2.2.3.2.4](#), are passed with the session context cookie associated with the

expired Session Context. New session context cookies are passed back in the response for the re-established **Session Context** using the **Set-Cookie** header, as described in section [2.2.3.2.3](#).

Client request

```
POST <Autodiscover-provided endpoint> HTTP/1.1
Host: <URL of the host server>
Content-Length: <length of REQUEST BODY>
Content-Type: application/mapi-http
Cookie: <session context cookie>=<opaque string>
Cookie: <request sequence cookie>=<opaque string>
X-RequestType: Connect
X-ClientInfo: <opaque string>
X-RequestId: <unique identifier>
X-ClientApplication: <client version>

<REQUEST BODY>
```

Server response

```
HTTP/1.1 200 OK
Content-Length: <length of META-TAGS, ADDITIONAL HEADERS and RESPONSE BODY>
Content-Type: application/mapi-http
Set-Cookie: <session context cookie>=<new opaque string>
Set-Cookie: <request sequence cookie>=<new opaque string>
X-RequestType: Connect
X-RequestId: <unique identifier>
X-ResponseCode: 0
X-ClientInfo: <opaque string>
X-ServerApplicaiton: <server version>
X-ExpirationInfo: <milliseconds>
<CRLF>
PROCESSING<CRLF>
DONE<CRLF>
X-ResponseCode: 0<CRLF>
X-ElapsedTime: <milliseconds>
X-StartTime: <date/time>
<CRLF>
<RESPONSE BODY>
```

4.5 Disconnect from the Server

This scenario describes disconnecting from server. The client passes the session context cookie to the server as described in section [3.1.5.4](#). The server responds as described in section [3.2.5.4](#).

Client request

```
POST <Autodiscover-provided endpoint> HTTP/1.1
Host: <URL of the host server>
Content-Length: <length of REQUEST BODY>
Content-Type: application/mapi-http
Cookie: <session context cookie>=<opaque string>
Cookie: <request sequence cookie>=<opaque string>
X-RequestType: Disconnect
X-ClientInfo: <opaque string>
X-RequestId: <unique identifier>
```


X-ClientApplication: <client version>

< REQUEST BODY>

Server response

HTTP/1.1 200 OK
Content-Length: <length of META-TAGS, ADDITIONAL HEADERS and RESPONSE BODY>
Content-Type: application/mapi-http
X-RequestType: Disconnect
X-RequestId: <unique identifier>
X-ResponseCode: 0
X-ClientInfo: <opaque string>
X-ServerApplication: <server version>
X-ExpirationInfo: <milliseconds>
<CRLF>
PROCESSING<CRLF>
DONE<CRLF>
X-ResponseCode: 0<CRLF>
X-ElapsedTime: <milliseconds>
X-StartTime: <date/time>
<CRLF>
<RESPONSE BODY>

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

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

- Microsoft Exchange Server 2013 Service Pack 1 (SP1)
- Microsoft Outlook 2013 Service Pack 1 (SP1)

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

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

7 Change Tracking

This section identifies changes that were made to the [MS-OXCMAPIHTTP] protocol document between the April 2014 and July 2014 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.
- The removal of a document from the documentation set.

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

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

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

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

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

- Obsolete document removed.

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
2.2.1.3 LargePropertyTagArray Structure	Changed "LargePropTagArray" to "LargePropertyTagArray".	N	Content updated.
2.2.2.2 Common Response Format	Removed the Host header from the list of headers returned as response headers.	Y	Content updated.
2.2.2.2 Common Response Format	Specified the X-RequestId header in the response format.	Y	Content updated.
2.2.3.2.1 Content-Length Header Field	Specified that the header is not used in chunked responses.	N	Content updated.
2.2.3.2.2 Content-Type Header Field	Added more detail to the definition for this field to describe required contents under specific conditions.	N	New content added.
2.2.3.3.5 X-PendingPeriod Header Field	Changed the name of the header field described in this section from "X-PendingInterval" to "X-PendingPeriod".	N	Content updated.
2.2.3.3.7 X-ServerApplication Header Field	Updated the format for the value of this header field from "Exchange/15.x.xxx.x" to "Exchange/15.xx.xxxx.xxx" to match the format of data on the wire.	N	Content updated.
2.2.3.3.10 X-StartTime Header	Updated X-StartTime header description to include specification of the header format.	N	Content updated.
2.2.4 Request Types for Mailbox Server Endpoint	Replaced the tables with packet diagrams and field definitions in the 12 subsections that specify request bodies and response bodies for the mailbox server endpoint request types.	N	Content updated.
2.2.5 Request Types for Address Book Server Endpoint	Replaced the tables with packet diagrams and field definitions in the 57 subsections that specify request bodies and response bodies for the address book server endpoint request types.	N	Content updated.
2.2.5.1.3	Updated section with request type failure	Y	Content

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
Bind Request Type Failure Response Body	response field structure.		updated.
2.2.5.2.3 Unbind Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.3.2 CompareMinIds Request Type Success Response Body	Added table of values for the Result field.	N	Content updated.
2.2.5.3.3 CompareMinIds Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.4.3 DnToMinId Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.5.1 GetMatches Request Type Request Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the Columns field.	N	Content updated.
2.2.5.5.2 GetMatches Request Type Success Response Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the Columns field.	N	Content updated.
2.2.5.5.3 GetMatches Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.6.2 GetPropList Request Type Success Response Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the PropertyTags field.	N	Content updated.
2.2.5.6.3 GetPropList Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.7.1 GetProps Request Type Request Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the PropertyTags field.	N	Content updated.
2.2.5.7.2 GetProps Request Type Success Response Body	Changed "AddressBookPropValueList" to "AddressBookPropertyValueList" in the description of the PropertyValues field.	N	Content updated.
2.2.5.7.3 GetProps Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.8.2 GetSpecialTable Request Type Success Response Body	Changed "AddressBookPropValueList" to "AddressBookPropertyValueList" in the description of the Rows field.	N	Content updated.
2.2.5.8.3	Updated section with request type failure	Y	Content

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
GetSpecialTable Request Type Failure Response Body	response field structure.		updated.
2.2.5.9.2 GetTemplateInfo Request Type Success Response Body	Changed "AddressBookPropValueList" to "AddressBookPropertyValueList" in the description of the Row field.	N	Content updated.
2.2.5.9.3 GetTemplateInfo Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.10.3 ModLinkAtt Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.11.1 ModProps Request Type Request Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the PropertiesTags field.	N	Content updated.
2.2.5.11.3 ModProps Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.12.1 QueryRows Request Type Request Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the Columns field.	N	Content updated.
2.2.5.12.2 QueryRows Request Type Success Response Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the Columns field.	N	Content updated.
2.2.5.12.3 QueryRows Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.13.2 QueryColumns Request Type Success Response Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the Columns field.	N	Content updated.
2.2.5.13.3 QueryColumns Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.14.1 ResolveNames Request Type Request Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the PropertyTags field.	N	Content updated.
2.2.5.14.2 ResolveNames Request Type Success Response Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the PropertyTags field.	N	Content updated.
2.2.5.14.3 ResolveNames Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.15.3	Updated section with request type failure	Y	Content

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
ResortRestriction Request Type Failure Response Body	response field structure.		updated.
2.2.5.16.1 SeekEntries Request Type Request Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the Columns field.	N	Content updated.
2.2.5.16.2 SeekEntries Request Type Success Response Body	Changed "LargePropTagArray" to "LargePropertyTagArray" in the description of the Columns field.	N	Content updated.
2.2.5.16.3 SeekEntries Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.17.3 UpdateStat Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.18.3 GetMailboxUrl Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
2.2.5.19.3 GetAddressBookUrl Request Type Failure Response Body	Updated section with request type failure response field structure.	Y	Content updated.
3.1.5.4 Destroying a Session Context by Using the Disconnect or Unbind Request Type	Deleted the statement about the request bodies of the Disconnect and Unbind request types and added links to the request body sections for Disconnect and Unbind.	N	Content updated.
3.2.5.1 Responding to a Connect or Bind Request Type Request	Clarified server's response for success and failure.	N	Content updated.
3.2.5.2 Responding to All Request Type Requests	Changed "Cookie headers" to "Set-Cookie headers".	N	Content updated.
3.2.5.2 Responding to All Request Type Requests	Added content from "Using Response Meta-Tags" section, which is now removed.	N	Content updated.
3.2.5.4 Responding to a Disconnect or Unbind Request Type Request	Clarified server's response for success and failure.	N	Content updated.
3.2.5.4 Responding to a Disconnect or Unbind Request Type Request	Specified that the server's response includes the Content-Length header if the response is not chunked.	Y	Content updated.
3.2.5.4	Specified that all relevant Cookie headers are	Y	Content

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
Responding to a Disconnect or Unbind Request Type Request	included in the response even though the session context cookie has been invalidated.		updated.
3.2.5.5 Responding to a NotificationWait Request Type Request	Specified that the server's response includes the Content-Length header if the response is not chunked.	Y	Content updated.
	Removed "Using Response Meta-Tags" section and moved its content to "Responding to All Request Type Requests" section.	N	Content removed.

8 Index

A

Abstract data model
[client](#) 78
[server](#) 81
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 92
Client
[abstract data model](#) 78
[higher-layer triggered events](#) 78
[initialization](#) 78
[other local events](#) 81
[timer events](#) 81
[timers](#) 78
[Common Data Types message](#) 12

D

Data model - abstract
[client](#) 78
[server](#) 81
[Disconnect from the server example](#) 88

E

[Establish a new session context example](#) 85
Examples
[disconnect from the server](#) 88
[establish a new session context](#) 85
[issue ROP commands to the server](#) 85
[re-establish a timed-out connection to the server](#)
87
[refresh an idle session context](#) 87

F

[Fields - vendor-extensible](#) 11

G

[Glossary](#) 7

H

[Header Fields message](#) 15
Higher-layer triggered events
[client](#) 78
[server](#) 82

I

[Implementer - security considerations](#) 90
[Index of security parameters](#) 90
[Informative references](#) 8

Initialization
[client](#) 78
[server](#) 81
[Introduction](#) 7
[Issue ROP commands to the server example](#) 85

M

Messages
[Common Data Types](#) 12
[Header Fields](#) 15
[PING Request Type](#) 77
[POST Method](#) 13
[Request Types for Address Book Server Endpoint](#)
28
[Request Types for Mailbox Server Endpoint](#) 20
[Response Meta-Tags](#) 77
[transport](#) 12

N

[Normative references](#) 8

O

Other local events
[client](#) 81
[server](#) 84
[Overview \(synopsis\)](#) 9

P

[Parameters - security index](#) 90
[PING Request Type message](#) 77
[POST Method message](#) 13
[Preconditions](#) 11
[Prerequisites](#) 11
[Product behavior](#) 91

R

[Re-establish a timed-out connection to the server](#)
[example](#) 87
[References](#) 8
[informative](#) 8
[normative](#) 8
[Refresh an idle session context example](#) 87
[Relationship to other protocols](#) 10
[Request Types for Address Book Server Endpoint](#)
[message](#) 28
[Request Types for Mailbox Server Endpoint](#)
[message](#) 20
[Response Meta-Tags message](#) 77

S

Security
[implementer considerations](#) 90

[parameter index](#) 90
Server
 [abstract data model](#) 81
 [higher-layer triggered events](#) 82
 [initialization](#) 81
 [other local events](#) 84
 [timer events](#) 84
 [timers](#) 81
[Standards assignments](#) 11

T

Timer events
 [client](#) 81
 [server](#) 84
Timers
 [client](#) 78
 [server](#) 81
[Tracking changes](#) 92
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 78
 [server](#) 82

V

[Vendor-extensible fields](#) 11
[Versioning](#) 11