

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

[MS-ASHTTP]:

Exchange ActiveSync: HTTP Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec              = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name           = 1*ALPHA
user-name              = 1*VCHAR
device-id              = 1*32 (ALPHA / DIGIT)
device-type            = 1*VCHAR

```

```
parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR
```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line  
Response-headers  
CR/LF  
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line  
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section [2.2.3](#). The client SHOULD [≤5>](#) use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section [2.2.1](#) and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section [2.2.1.1.2.2](#)) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [\[RFC2616\]](#)), the response is interpreted as specified in [\[MS-ASCMD\]](#) section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [\[MS-ASCMD\]](#) section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in the response, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```


Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

[MS-ASHTTP]:

Exchange ActiveSync: HTTP Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```

parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR

```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line
Response-headers
CR/LF
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line  
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

[MS-ASHTTP]:

Exchange ActiveSync: HTTP Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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						
Tag										Length										Value (variable)																	
...																																					

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOF) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
*('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```



```

parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR

```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line  
Response-headers  
CR/LF  
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request


```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

[MS-ASHTTP]:

Exchange ActiveSync: HTTP Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOF): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and HTTP **OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- HTTP **OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```
parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR
```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line
Response-headers
CR/LF
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

[MS-XOAUTH]:

OAuth 2.0 Authorization Protocol Extensions

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
4/27/2012	0.1	New	Released new document.
7/16/2012	0.1	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	1.0	Major	Significantly changed the technical content.
2/11/2013	2.0	Major	Significantly changed the technical content.
7/26/2013	2.1	Minor	Clarified the meaning of the technical content.
11/18/2013	2.1	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
5/26/2015	3.0	Major	Significantly changed the technical content.
9/14/2015	3.1	Minor	Clarified the meaning of the technical content.
6/13/2016	3.2	Minor	Clarified the meaning of the technical content.
9/14/2016	3.2	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1	Introduction	5
1.1	Glossary	5
1.2	References	6
1.2.1	Normative References	6
1.2.2	Informative References	7
1.3	Overview	7
1.4	Relationship to Other Protocols	7
1.5	Prerequisites/Preconditions	7
1.6	Applicability Statement	7
1.7	Versioning and Capability Negotiation	7
1.8	Vendor-Extensible Fields	7
1.9	Standards Assignments.....	7
2	Messages.....	8
2.1	Transport	8
2.2	Message Syntax	8
3	Protocol Details.....	10
3.1	Client Details.....	10
3.1.1	Abstract Data Model.....	10
3.1.2	Timers	10
3.1.3	Initialization	10
3.1.4	Higher-Layer Triggered Events	10
3.1.5	Message Processing Events and Sequencing Rules	10
3.1.5.1	Realm Autodiscovery Through HTTP 401 Challenge	10
3.1.5.2	Server-to-Server Security Token Contents	10
3.1.6	Timer Events.....	10
3.1.7	Other Local Events.....	10
3.2	Server Details.....	10
3.2.1	Abstract Data Model.....	10
3.2.2	Timers	11
3.2.3	Initialization	11
3.2.4	Higher-Layer Triggered Events	11
3.2.5	Message Processing Events and Sequencing Rules	11
3.2.5.1	Authentication Within a Single Organization	11
3.2.5.2	Authentication with User Information Within a Single Organization.....	11
3.2.5.3	Authentication with Third-Party Application	12
3.2.5.4	Realm Autodiscovery Through HTTP 401 Challenge	13
3.2.5.5	Server-to-Server Security Token Contents	13
3.2.5.6	Server-to-Server Validation Criteria.....	13
3.2.6	Timer Events.....	14
3.2.7	Other Local Events.....	14
4	Protocol Examples	15
4.1	Security Token Issued by STS	15
4.2	Security Token Self-Issued by Client	15
4.3	Security Token Issued by STS with User Information Added by Client	16
4.4	Security Token Self-Issued By Client with User Information.....	17
4.5	Security Token for Accessing a Third-Party Service with Extensions	18
4.6	Realm Autodiscovery Through HTTP 401 Challenge	19
5	Security	20
5.1	Security Considerations for Implementers	20
5.2	Index of Security Parameters	20
6	Appendix A: Product Behavior	21

7	Change Tracking.....	22
8	Index.....	23

1 Introduction

The OAuth 2.0 Authorization Protocol Extensions extend the OAuth 2.0 Authentication Protocol and the JSON Web Token (JWT) to enable server-to-server authentication.

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

1.1 Glossary

This document uses the following terms:

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

private key: One of a pair of keys used in public-key cryptography. The private key is kept secret and is used to decrypt data that has been encrypted with the corresponding public key. For an introduction to this concept, see [\[CRYPTO\]](#) section 1.8 and [\[IEEE1363\]](#) section 3.1.

public key: One of a pair of keys used in public-key cryptography. The public key is distributed freely and published as part of a digital certificate. For an introduction to this concept, see [\[CRYPTO\]](#) section 1.8 and [\[IEEE1363\]](#) section 3.1.

realm: An administrative boundary that uses one set of authentication servers to manage and deploy a single set of unique identifiers. A realm is a unique logon space.

realm autodiscovery: A process used by client applications to obtain the name of a server resource's source **realm** and then use that information to locate a **security token service (STS)** that can issue access tokens to the resource.

security principal: A unique entity that is identifiable through cryptographic means by at least one key. It frequently corresponds to a human user, but also can be a service that offers a resource to other security principals. Also referred to as principal.

security principal identifier: A value that is used to uniquely identify a **security principal**. In Windows-based systems, it is a security identifier (SID). In other types of systems, it can be a user identifier or other type of information that is associated with a **security principal**.

security token: An opaque message or data packet produced by a Generic Security Services (GSS)-style authentication package and carried by the application protocol. The application has no visibility into the contents of the token.

security token service (STS): A web service that issues claims (2) and packages them in encrypted security tokens.

Transmission Control Protocol (TCP): A protocol used with the Internet Protocol (IP) to send data in the form of message units between computers over the Internet. TCP handles keeping

track of the individual units of data (called packets) that a message is divided into for efficient routing through the Internet.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

user principal name (UPN): A user account name (sometimes referred to as the user logon name) and a domain name that identifies the domain in which the user account is located. This is the standard usage for logging on to a Windows domain. The format is: someone@example.com (in the form of an email address). In Active Directory, the userPrincipalName attribute (2) of the account object, as described in [\[MS-ADTS\]](#).

X.509: An ITU-T standard for public key infrastructure subsequently adapted by the IETF, as specified in [\[RFC3280\]](#).

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

1.2 References

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

1.2.1 Normative References

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

[IETF-DRAFT-JWT-LATEST] Jones, M., Bradley, J., and Sakimura, N., "JSON Web Token (JWT) draft-ietf-oauth-json-web-token-08", draft-ietf-oauth-json-web-token-08, May 2013, <http://datatracker.ietf.org/doc/draft-ietf-oauth-json-web-token/>

[MS-DTYP] Microsoft Corporation, "[Windows Data Types](#)".

[MS-OAUTH2EX] Microsoft Corporation, "[OAuth 2.0 Authentication Protocol Extensions](#)".

[MS-ODATA] Microsoft Corporation, "[Open Data Protocol \(OData\)](#)".

[MS-SPSTWS] Microsoft Corporation, "[SharePoint Security Token Service Web Service 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.rfc-editor.org/rfc/rfc2616.txt>

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., et al., "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999, <http://www.rfc-editor.org/rfc/rfc2617.txt>

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

[RFC793] Postel, J., Ed., "Transmission Control Protocol: DARPA Internet Program Protocol Specification", RFC 793, September 1981, <http://www.rfc-editor.org/rfc/rfc793.txt>

1.2.2 Informative References

[MS-SPS2SAUTH] Microsoft Corporation, "[OAuth 2.0 Authentication Protocol: SharePoint Profile](#)".

1.3 Overview

These extensions specify how applications can perform server-to-server authentication using a **security token service (STS)**. For example, an email service might use these extensions to authenticate itself when it makes a call to an instant messaging service. Both of these services are server applications. However, in the scope of this protocol, the email service would be the client, and the instant messaging service would be the server. For an example of a server-to-server **security token** that a client might send to authenticate itself, see section [4.2](#).

1.4 Relationship to Other Protocols

These extensions extend the OAuth 2.0 Authentication Protocol, as described in [\[MS-OAUTH2EX\]](#), and JSON Web Token (JWT), as described in [\[IETFDRAFT-JWT-LATEST\]](#).

For information on how to implement an STS, see [\[MS-SPSTWS\]](#).

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

1.5 Prerequisites/Preconditions

The client is required to reside in the same **realm** as the STS to request a server-to-server security token from it.

1.6 Applicability Statement

These extensions apply only when a service call is made to or from an application that supports server-to-server authentication.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

These extensions transport messages over **TCP**, as specified in [\[RFC793\]](#), and do not pass any specific parameters to the transport. Transport-layer security **MUST** be used to secure the security tokens. These extensions use **HTTPS**, as specified in [\[RFC2818\]](#), to do so.

Messages sent using these extensions are not encoded by Open-Data, as specified in [\[MS-ODATA\]](#), and use the default character set defined by the client or the server.

Security tokens are JWTs and are sent using **HTTP Authorization** headers, as specified in [\[IETFdraft-jwt-latest\]](#). **HTTP Authorization** headers are specified in [\[RFC2617\]](#).

2.2 Message Syntax

A **security principal** is represented as a **security principal identifier** in the messages sent by applications. A security principal identifier is a **GUID**.

For more details about the messages typically exchanged between a client and an STS, see [\[MS-SPSTWS\]](#).

For clarity, this document uses different names to refer to the server-to-server security tokens that are exchanged in various scenarios. An actor token is a signed security token that is issued by an STS, or by the client itself if the server trusts it to do so. An outer token is an unsigned security token that is constructed by the client and contains user information in addition to an actor token. In this scenario, the actor token is referred to as the inner token. All of these security tokens are formatted in the same way, as specified in [\[IETFdraft-jwt-latest\]](#), and contain the claims and header fields specified in this section.

The following table describes claims that are exchanged in server-to-server security tokens. The claim values are all of data type **STRING**, as specified in [\[MS-DTYP\]](#).

Claim type	Claim value description	Example claim values
aud	The targeted service for which the client issued the server-to-server security token.	<security principal identifier>/<hostname>@<realm>
iss	The security principal identifier of the server-to-server security token issuer.	<security principal identifier>@<realm>
nameid	The logged on user's user principal name (UPN) value for the security principal that made the request.	user@contoso.com
nbf	The time at which the server-to-server security token was created.	129592882368666656
exp	The time at which the server-to-server security token expires.	129592882368666656
trustedfordelegation	"true" if the client is trusted to delegate a user identity; otherwise, "false".	true false
identityprovider	The identity provider that authenticated the caller.	windows forms trusted

Claim type	Claim value description	Example claim values
actort	The security token issued and signed by the STS. An actor token has the same format as any other security token.	See section 4.3 and section 4.4 .
smtp	The logged on user's email address.	user@contoso.com
sip	The logged on user's sip address.	user@contoso.com
msexchuid	A unique identifier that the STS can give the user. This is an additional claim that the STS adds and is not required by the OAuth 2.0 Authentication Protocol, as specified in [MS-OAUTH2EX] .	objectGUID@contoso.com
appctx	The application context. This claim contains a subset of claims that is specific to the service accessed by the client.	See section 4.5 .

The following list describes the header fields in a server-to-server security token. The field values are all of data type **STRING**, as specified in [MS-DTYP].

- **typ**. The token type. The value MUST always be "JWT".
- **alg**. The algorithm used to encrypt the contents of the token. The value of this field MUST be either "none" or "rs256". Actor tokens are always signed and have **alg** fields that contain the value "rs256". Outer tokens that contain inner signed tokens, as described in section 4.3 and section 4.4, are not signed and have **alg** fields that contain the value "none".
- **x5t**. The **base64** encoded thumbprint of the certificate used to sign the security token. This field is optional.

The header fields are contained in a separate part of the security token, as specified in [IETF-DRAFT-JWT-LATEST].

3 Protocol Details

3.1 Client Details

Note that the client and server are in fact both server applications. However, in the scope of this protocol, one server application acts as the client, and one server application acts as the server.

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 Realm Autodiscovery Through HTTP 401 Challenge

A client can use **realm autodiscovery** to obtain the name of its realm, which it then uses to locate an STS. It uses realm autodiscovery by sending a request to the server that contains an empty **Bearer HTTP Authorization** header to the server. The **HTTP Authorization** header is specified in [\[RFC2617\]](#).

For an example of realm autodiscovery through HTTP 401 challenge, see section [4.6](#).

3.1.5.2 Server-to-Server Security Token Contents

The server-to-server security token sent by the client MUST be compatible with the JWT format specified in [\[IETFDRAFT-JWT-LATEST\]](#) and [\[MS-OAUTH2EX\]](#).

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

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 Authentication Within a Single Organization

The following procedure shows the authentication that takes place when a client makes a call to a server in the same organization using these extensions.

1. The organization's IT administrator sets up an STS and configures it with the security principal identifiers for the client and server. The client and server each exchange **public keys**, carried in **X.509** certificates, with the STS. The administrator also configures the client and server to trust security tokens issued by the STS.
2. The client makes an anonymous request to the server.
3. The server responds with an HTTP 401 challenge. HTTP 401 is specified in [\[RFC2616\]](#) and [\[RFC2617\]](#).
4. The client requests a security token from the STS. It does this by sending a self-issued security token that is signed with its **private key**. The security token contains the **aud**, **iss**, **nameid**, **nbf**, **exp**, and **trustedfordelegation** claims as specified in section [2.2](#). The client request also includes a *resource* parameter and a *realm* parameter, as specified in [\[MS-OAUTH2EX\]](#). The value of the *resource* parameter is the **Uniform Resource Identifier (URI)** of the server. For an example of a self-issued security token, see section [4.2](#).
5. The STS validates the public key of the security token provided by the client, verifies that the client is authorized to access the requested resource, and responds to the client with a server-to-server security token that is signed with a public key that the server trusts. The security token contains the **aud**, **iss**, **nameid**, **nbf**, **exp**, and **identityprovider** claims, as specified in section [2.2](#). For an example of a server-to-server security token issued by an STS, see section [4.1](#).
6. The client sends the server-to-server security token to the server.
7. The server validates the server-to-server security token by checking the values of the **aud**, **iss**, and **exp** claims and the public key provided by the STS. It performs additional validation checks to ensure that the client is authorized to access the requested resource. It then responds to the client with the requested resource.

3.2.5.2 Authentication with User Information Within a Single Organization

The following procedure shows the authentication that takes place when a client makes a call to a server in the same organization using these extensions. The client also sends user information to the server, and the server uses the information to determine whether to return the requested resource.

1. The organization's IT administrator sets up an STS and configures it with the security principal identifiers for the client and server. The client and server each exchange public keys, carried in

X.509 certificates, with the STS. The administrator also configures the client and server to trust security tokens issued by the STS.

2. The client makes an anonymous request to the server.
3. The server responds with an HTTP 401 challenge, as specified in [\[RFC2616\]](#) and [\[RFC2617\]](#).
4. The client requests a security token from the STS. It does this by sending a self-issued security token that is signed with its private key. The security token contains the **aud**, **iss**, **nameid**, **trustedfordelegation**, **nbf**, and **exp** claims, as specified in section [2.2](#). The client request also includes a *resource* parameter and a *realm* parameter, as specified in [\[MS-OAUTH2EX\]](#). The value of the *resource* parameter is the URI of the server. For an example of a self-issued security token, see section [4.2](#).
5. The STS validates the public key of the security token provided by the client, verifies that the client is authorized to access the requested resource, and responds to the client with a server-to-server security token that is signed with a public key that the server trusts. The security token contains the **aud**, **iss**, **nameid**, **nbf**, and **exp** claims. For an example of a server-to-server security token issued by an STS that contains user information, see section [4.3](#).
6. The client sends a self-issued security token to the server that includes the server-to-server security token it received from the STS, as well as additional claims that contain the user information. The additional claims are the **aud**, **iss**, **nameid**, **nbf**, **exp**, **smtp**, **sip**, **msexchuid**, and **actort** claims, as specified in section [2.2](#). The **actort** claim contains the server-to-server security token provided by the STS. Note that the server-to-server security token is signed, but the user information is not. For an example of a self-issued server-to-server security token that contains user information, see section [4.4](#).
7. The server validates the request by checking the user information contained in the **aud**, **iss**, and **exp** claims and the public key used to sign the security token provided by the STS. Because the user information is not signed, the server validates the user information by checking that the values of the **aud** and **iss** claims in the user information match the values of the **aud** and **iss** claims in the server-to-server security token contained in the **actort** claim. For security considerations regarding the unsigned user information, see section [5.1](#).
8. The server performs additional validation checks to ensure that the client is authorized to access the requested resource. It then responds to the client with the requested resource.

3.2.5.3 Authentication with Third-Party Application

The following procedure shows the authentication that takes place when a client makes a call to a third-party application in the same organization using these extensions.

1. The organization's IT administrator sets up an STS and configures it with the security principal identifiers for the client and third-party application. The client and third-party application each exchange public keys, carried in X.509 certificates, with the STS. The administrator also configures the client and third-party application to trust security tokens issued by the STS.
2. The client makes an anonymous request to the third-party application.
3. The third-party application responds with an HTTP 401 challenge, as specified in [\[RFC2616\]](#) and [\[RFC2617\]](#).
4. The client requests a security token from the STS. It does this by sending a self-issued security token that is signed with its private key. The security token contains the **aud**, **iss**, **nameid**, **nbf**, and **exp** claims, as specified in section [2.2](#).
5. The STS validates the public key of the security token provided by the client, verifies that the client is authorized to access the requested resource, and returns a server-to-server security

token that is signed with a public key that the third-party application trusts. The security token contains the **aud**, **iss**, **nameid**, **nbf**, **exp**, and **appctx** claims, as specified in section 2.2. The **appctx** claim contains information that is implementation-specific to the third-party application. For an example of a server-to-server security token that is used to access a third-party application, see section [4.5](#).

6. The client sends the server-to-server security token to the third-party application.
7. The third-party application validates the server-to-server security token by checking the values of the **aud**, **iss**, and **exp** claims and the public key provided by the STS. It performs additional validation checks to ensure that the client is authorized to access the requested resource. It then responds to the client with the requested resource.

3.2.5.4 Realm Autodiscovery Through HTTP 401 Challenge

When a server receives a request that contains an empty **Bearer HTTP Authorization** header, the server responds with an HTTP 401 challenge, as specified in [\[RFC2616\]](#) and [\[RFC2617\]](#). The **Bearer WWW-Authenticate** header of the HTTP 401 challenge contains the following fields:

- **client_id**. The client's security principal identifier.
- **realm**. The server MAY [<1>](#) return this field. This is the source realm of the client.
- **trusted_issuers**. A comma-separated list of all security token issuers the server trusts. The client can then select a security token issuer to request a security token.

For an example of realm autodiscovery through HTTP 401 challenge, see section [4.6](#).

3.2.5.5 Server-to-Server Security Token Contents

The server can accept server-to-server security tokens with the claim types specified in section [2.2](#).

3.2.5.6 Server-to-Server Validation Criteria

The server accepts a server-to-server security token that meets the following criteria:

- The server-to-server security token is signed with a trusted signing certificate from an STS that the server trusts.
- The server-to-server security token contains an **iss** claim whose value shows that the security token is issued by an STS that the server trusts.
- The server-to-server security token contains a **nameid** claim with the UPN value of the logged-on user.
- If the client constructs an unsigned outer security token to contain user information as well as a signed actor token (that is, an inner token), as described in section [4.3](#) and section [4.4](#), the value of the **iss** claim in the outer token matches the value of the **nameid** claim in the inner token. The server performs a case-sensitive comparison.
- The server-to-server security token contains an **aud** claim whose value meets the following criteria:
 - The **aud** claim value MUST contain three parts: **client_id**, **hostname**, and **realm**.
 - The value of the **client_id** part is the security principal identifier of a security principal that the server trusts. The server performs a case-sensitive comparison.

- The value of the hostname part is the host name of the server. The server performs a case-insensitive comparison to verify that it is the target of the request.
- The value of the realm part is the source realm. The server performs a case-sensitive comparison.

The STS uses the claims in the server-to-server security token to authenticate the caller.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 Security Token Issued by STS

In this example, a client attempts to access a resource on the server. The server responds with an HTTP 401 challenge that lists the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```
HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0
```

1. The client sends its credentials to the indicated token issuer, which is an STS.
2. The STS authenticates the client and issues an actor token to the client.
3. The client uses the actor token to access the resource it requested on the server.

The following is an example of an actor token issued by an STS. For more information about the claim values contained in this security token, see section [2.2](#).

```
actor:
{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "XqrnFEfsS55_vMBpHvF0pTnqeaM"
}.{
  "aud": "00000003-0000-0ff1-ce00-000000000000/contoso.com@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "iss": "00000001-0000-0000-c000-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "nbf": "1323380070",
  "exp": "1323383670",
  "nameid": "00000002-0000-0ff1-ce00-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "identityprovider": "00000001-0000-0000-c000-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8"
}
```

4.2 Security Token Self-Issued by Client

In this example, the client tries to access a resource on the server. The server responds with an HTTP 401 challenge that indicates the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```
HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
```

Content-Length: 0

1. The client is one of the token issuers trusted by the server, so it creates an actor token and signs it with its credentials.
2. The client uses the actor token to access the resource it requested on the server.

The following is an example of an actor token that is self-issued by a client. For more information about the claim values contained in this security token, see section [2.2](#).

```
actor:
{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "mH-TTlt-HAXC9-vjKVfTX6bAsR0"
}. {
  "aud": "00000003-0000-0ff1-ce00-000000000000/contoso.com@EXHB-88371dom.extest.contoso.com",
  "iss": "00000002-0000-0ff1-ce00-000000000000@EXHB-88371dom.extest.contoso.com",
  "nbf": "1323380605",
  "exp": "1323409405",
  "nameid": "00000002-0000-0ff1-ce00-000000000000@EXHB-88371dom.extest.contoso.com",
  "trustedfordelegation": "true"
}
```

4.3 Security Token Issued by STS with User Information Added by Client

In this example, the client tries to access a resource on the server. The server responds with an HTTP 401 challenge that lists the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```
HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@*"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0
```

1. The client sends its credentials to the indicated security token issuer, which is an STS.
2. The STS authenticates the client and issues an actor token to the client.
3. The client constructs an unsigned outer token that contains additional user information to provide to the server. The outer token also contains the signed actor token issued by the STS.
4. The client uses the outer token to access the resource it requested on the server.

The following is an example of an outer token that is constructed by the client and contains user information, as well as an actor token issued by an STS. For more information about the claim values contained in this security token, see section [2.2](#).

```
{
  "typ": "JWT",
  "alg": "none"
}. {
```

```

    "aud": "00000003-0000-0ff1-ce00-000000000000/fabrikam.com@fabrikam-oauth-
929.extest.fabrikam.com",
    "iss": "00000001-0000-0ff1-ce00-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
    "nbf": "1323380069",
    "exp": "1323408869",
    "nameid": "00000002-0000-0ff1-ce00-000000000000@BLID-EXHB-90232dom.extest.contoso.com
    "actort": "..actor token.."
}

```

The following is an example of an actor token, as mentioned in the previous listing.

```

{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "hEAW-SXzTNaDBUwfAh2YScnBOxA"
}. {
  "aud": "00000002-0000-0ff1-ce00-000000000000/contoso.com@EXHB-
88371dom.extest.contoso.com",
  "iss": "00000003-0000-0ff1-ce00-000000000000@e54c2f60-0ad3-4ef8-8ba2-b3ae01b35494",
  "nbf": "1346674665",
  "exp": "1346804265",
  "nameid": "00000003-0000-0ff1-ce00-000000000000@e54c2f60-0ad3-4ef8-8ba2-b3ae01b35494"
}

```

4.4 Security Token Self-Issued By Client with User Information

In this example, the client tries to access a resource on the server. The server responds with an HTTP 401 challenge that indicates the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```

HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-cl69a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0

```

1. The client is one of the token issuers trusted by the server, so it creates an actor token and signs it with its credentials.
2. The client constructs an unsigned outer token that contains additional user information to provide to the server. The outer token also contains the signed actor token issued by the client.
3. The client uses the outer token to access the resource it requested on the server.

The following is an example of an outer token that is constructed by the client and contains user information, as well as a security token that is self-issued by the client. For more information about the claim values contained in this security token, see section [2.2](#).

```

{
  "typ": "JWT",
  "alg": "none"
}. {
  "aud": "00000003-0000-0ff1-ce00-000000000000/contoso.com@EXHB-
88371dom.extest.contoso.com",

```

```

    "iss": "00000002-0000-0ff1-ce00-000000000000@EXHB-88371dom.extest.contoso.com",
    "nbf": "1323380605",
    "exp": "1323409405",
    "nameid": "ewsuser-55a83300@EXHB-88371dom.extest.contoso.com",
    "smtp": "ewsuser-55a83300@exhb-88371dom.extest.contoso.com",
    "sip": "ewsuser-55a83300@exhb-88371dom.extest.contoso.com",
    "msexchuid": "842e4c3a-0879-4973-83f9-495bb9863e18@exhb-88371dom.extest.contoso.com",
    "actort": "..actor token.."
}

```

The following is an example of an actor token, as mentioned in the previous listing.

```

{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "hEAW-SXzTNaDBUwfAh2YScnBOxA"
}. {
  "aud": "00000002-0000-0ff1-ce00-000000000000/contoso.com@EXHB-88371dom.extest.contoso.com",
  "iss": "00000003-0000-0ff1-ce00-000000000000@e54c2f60-0ad3-4ef8-8ba2-b3ae01b35494",
  "nbf": "1346674665",
  "exp": "1346804265",
  "nameid": "00000003-0000-0ff1-ce00-000000000000@e54c2f60-0ad3-4ef8-8ba2-b3ae01b35494"
}

```

4.5 Security Token for Accessing a Third-Party Service with Extensions

In this example, the client tries to access a resource on a third-party service. The service responds with an HTTP 401 challenge that lists the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```

HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0

```

1. The client sends its credentials to the indicated security token issuer, which is an STS. The credentials include an **appctx** claim that contains additional user information to be provided to the service. The STS does not examine or modify the user information in the **appctx** claim.
2. The STS authenticates the client and issues an actor token to the client that includes the **appctx** claim provided by the client.
3. The client uses the actor token to access the resource it requested on the service. The actor token includes the **appctx** claim that was previously supplied by the client and contains additional user information to be provided to the service. This scenario is analogous to the use of outer and inner tokens to convey additional user information to a server beyond what is required to authenticate to an STS, as described in section [4.3](#) and section [4.4](#).

The following is an example of a security token that is used to access a third-party service and contains user information. For more information about the claim values contained in this security token, see section [2.2](#).

```
{
  "aud": "00000002-0000-0ff1-ce00-000000000000/exhb-42702.exhb-42702dom.extest.contoso.com@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "iss": "00000002-0000-0ff1-ce00-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "nbf": "1323197012",
  "exp": "1323225812",
  "nameid": "https://www.fabrikam.com/weprintem",
  "appctx": "{
    "nameid": "ewsuser-cff3d495@BLID-EXHB-42702dom.extest.contoso.com",
    "smtp": "ewsuser-cff3d495@BLID-EXHB-42702dom.extest.contoso.com",
    "msexchuid": "842e4c3a-0879-4973-83f9-495bb9863e18@exhb-88371dom.extest.contoso.com"
  }"
}
```

4.6 Realm Autodiscovery Through HTTP 401 Challenge

In this example, the client tries to access a resource on a server. It also tries to use realm autodiscovery by including an empty **Bearer** authorization header in its request. An example of such a request is as follows.

```
POST https://contoso.com/autodiscover/autodiscover.svc HTTP/1.1
Content-Type: text/xml; charset=utf-8
Accept: text/xml
User-Agent: Test/1.0 (ContosoServicesClient/15.00.0424.000)
client-request-id: 00000000-0000-0000-0000-000000000000
Authorization: Bearer
Host: contoso.com
Content-Length: 1368
Expect: 100-continue
```

The server responds with an HTTP 401 challenge that lists the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```
HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: XJSUI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0
```

In this example, the server determines that the value in the **trusted_issuers** field contains sufficient information for the client to locate the STS, so the server does not include a **realm** field.

5 Security

5.1 Security Considerations for Implementers

The security considerations described in [\[MS-SPS2SAUTH\]](#) apply when implementing these extensions.

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
- Microsoft Exchange Server 2016
- Microsoft SharePoint Server 2013
- Microsoft SharePoint Foundation 2013
- Microsoft SharePoint Server 2016

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

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

[<1> Section 3.2.5.4](#): SharePoint Server 2013, SharePoint Foundation 2013, and SharePoint Server 2016 return this parameter.

7 Change Tracking

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

8 Index

A

Abstract data model

[client](#) 10

[server](#) 10

[Applicability](#) 7

[Authentication third-party application](#) 12

[Authentication with user information within a single organization](#) 11

[Authentication within a single organization](#) 11

C

[Capability negotiation](#) 7

[Change tracking](#) 22

Client

[abstract data model](#) 10

[higher-layer triggered events](#) 10

[initialization](#) 10

[other local events](#) 10

[timer events](#) 10

[timers](#) 10

D

Data model - abstract

[client](#) 10

[server](#) 10

E

Examples

[security token for accessing a third-party service with extensions](#) 18

[security token issued by STS](#) 15

[security token issued by STS with user information added by client](#) 16

[security token self-issued by client application](#) 15

[security token self-issued by client application with user information](#) 17

F

[Fields - vendor-extensible](#) 7

G

[Glossary](#) 5

H

Higher-layer triggered events

[client](#) 10

[server](#) 11

I

[Implementer - security considerations](#) 20

[Index of security parameters](#) 20

[Informative references](#) 7

Initialization

[client](#) 10

[server](#) 11

[Introduction](#) 5

M

Message processing - client

[realm autodiscovery through HTTP 401 challenge](#) 10

[server-to-server token contents](#) 10

Message processing - server

[authentication with third-party application](#) 12

[authentication with user information within a single organization](#) 11

[authentication within a single organization](#) 11

[realm autodiscovery through HTTP 401 challenge](#) 13

[server-to-server security token contents](#) 13

[server-to-server validation criteria](#) 13

Messages

[syntax](#) 8

[transport](#) 8

N

[Normative references](#) 6

O

Other local events

[client](#) 10

[server](#) 14

[Overview \(synopsis\)](#) 7

P

[Parameters - security index](#) 20

[Preconditions](#) 7

[Prerequisites](#) 7

[Product behavior](#) 21

R

Realm autodiscovery through HTTP 401 challenge

[client](#) 10

[server](#) 13

References

[informative](#) 7

[normative](#) 6

[Relationship to other protocols](#) 7

S

Security

[implementer considerations](#) 20

[parameter index](#) 20

[Security token for accessing a third-party service with extensions example](#) 18

[Security token issued by STS example](#) 15

[Security token issued by STS with user information added by client example](#) 16

[Security token self-issued by client application example](#) 15

[Security token self-issued by client application with user information example](#) 17
Sequencing rules - client
 [realm autodiscovery through HTTP 401 challenge](#) 10
 [server-to-server token contents](#) 10
Sequencing rules - server
 [authentication with third-party application](#) 12
 [authentication with user information within a single organization](#) 11
 [authentication within a single organization](#) 11
 [realm autodiscovery through HTTP 401 challenge](#) 13
 [server-to-server security token contents](#) 13
 [server-to-server validation criteria](#) 13
Server
 [abstract data model](#) 10
 [higher-layer triggered events](#) 11
 [initialization](#) 11
 [other local events](#) 14
 [timer events](#) 14
 [timers](#) 11
[Server-to-server security token contents - server](#) 13
Server-to-server token contents
 [client](#) 10
[Server-to-server validation criteria](#) 13
[Standards assignments](#) 7
[Syntax - messages](#) 8

T

Timer events
 [client](#) 10
 [server](#) 14
Timers
 [client](#) 10
 [server](#) 11
[Tracking changes](#) 22
[Transport - messages](#) 8

V

[Vendor-extensible fields](#) 7
[Versioning](#) 7

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOF) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
*('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```
parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR
```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line
Response-headers
CR/LF
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOF): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOF): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```
parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR
```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line
Response-headers
CR/LF
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and HTTP **OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- HTTP **OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section [2.2.1.1.1.2](#), or **base64 encoding**, as specified in section [2.2.1.1.1.1](#). Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [\[RFC2045\]](#). The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section [2.2.1.1.1.2](#), **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```

parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR

```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line
Response-headers
CR/LF
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section [2.2.3](#). The client SHOULD [≤5>](#) use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section [2.2.1](#) and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section [2.2.1.1.2.2](#)) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [\[RFC2616\]](#)), the response is interpreted as specified in [\[MS-ASCMD\]](#) section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [\[MS-ASCMD\]](#) section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in the response, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```


The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section [2.2.1.1.1.2](#), or **base64 encoding**, as specified in section [2.2.1.1.1.1](#). Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [\[RFC2045\]](#). The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section [2.2.1.1.1.2](#), **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
*('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```
parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR
```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWw0SE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line
Response-headers
CR/LF
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```


If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

[MS-ASHTTP]:

Exchange ActiveSync: HTTP Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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						
Tag										Length										Value (variable)																	
...																																					

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec              = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name              = 1*VCHAR
device-id              = 1*32 (ALPHA / DIGIT)
device-type            = 1*VCHAR

```

```

parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR

```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line
Response-headers
CR/LF
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section [2.2.3](#). The client SHOULD [≤5>](#) use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section [2.2.1](#) and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section [2.2.1.1.2.2](#)) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [\[RFC2616\]](#)), the response is interpreted as specified in [\[MS-ASCMD\]](#) section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [\[MS-ASCMD\]](#) section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in the response, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```


5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

[MS-ASHTTP]:

Exchange ActiveSync: HTTP Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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						
Tag										Length										Value (variable)																	
...																																					

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```
parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR
```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line  
Response-headers  
CR/LF  
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section [2.2.3](#). The client SHOULD [≤5>](#) use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section [2.2.1](#) and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section [2.2.1.1.2.2](#)) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [\[RFC2616\]](#)), the response is interpreted as specified in [\[MS-ASCMD\]](#) section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [\[MS-ASCMD\]](#) section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in the response, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server **MUST** conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it **SHOULD** include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it **MAY** [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it **MUST** include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server **SHOULD** include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server **SHOULD** include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server **MAY** [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking **SHOULD** use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers **SHOULD** limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but **MAY** [<9>](#) use different values for the number of changes or the time period. The server **SHOULD** block clients that exceed this limit for 14 hours, but **MAY** [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response **MUST** contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```


Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

[MS-ASHTTP]:

Exchange ActiveSync: HTTP Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
*('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```

parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR

```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line  
Response-headers  
CR/LF  
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

[MS-ASHTTP]:

Exchange ActiveSync: HTTP Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section [2.2.1.1.1.2](#), or **base64 encoding**, as specified in section [2.2.1.1.1.1](#). Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [\[RFC2045\]](#). The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section [2.2.1.1.1.2](#), **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```



```
parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR
```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client MUST NOT send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client MUST complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value MUST be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header SHOULD be set to "text/xml" or MAY [\[3\]](#) be set to "text/html". If the request has no body, the Content-Type header SHOULD NOT be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version MUST be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version MUST provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and SHOULD NOT be used for other command requests. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter SHOULD be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line
Response-headers
CR/LF
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section [2.2.3](#). The client SHOULD [≤5>](#) use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section [2.2.1](#) and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section [2.2.1.1.2.2](#)) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [\[RFC2616\]](#)), the response is interpreted as specified in [\[MS-ASCMD\]](#) section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [\[MS-ASCMD\]](#) section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in the response, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request


```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

[MS-XOAUTH]:

OAuth 2.0 Authorization Protocol Extensions

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
4/27/2012	0.1	New	Released new document.
7/16/2012	0.1	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	1.0	Major	Significantly changed the technical content.
2/11/2013	2.0	Major	Significantly changed the technical content.
7/26/2013	2.1	Minor	Clarified the meaning of the technical content.
11/18/2013	2.1	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
5/26/2015	3.0	Major	Significantly changed the technical content.
9/14/2015	3.1	Minor	Clarified the meaning of the technical content.
6/13/2016	3.2	Minor	Clarified the meaning of the technical content.
9/14/2016	3.2	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1	Introduction	5
1.1	Glossary	5
1.2	References	6
1.2.1	Normative References	6
1.2.2	Informative References	7
1.3	Overview	7
1.4	Relationship to Other Protocols	7
1.5	Prerequisites/Preconditions	7
1.6	Applicability Statement	7
1.7	Versioning and Capability Negotiation	7
1.8	Vendor-Extensible Fields	7
1.9	Standards Assignments.....	7
2	Messages.....	8
2.1	Transport	8
2.2	Message Syntax	8
3	Protocol Details.....	10
3.1	Client Details.....	10
3.1.1	Abstract Data Model.....	10
3.1.2	Timers	10
3.1.3	Initialization	10
3.1.4	Higher-Layer Triggered Events	10
3.1.5	Message Processing Events and Sequencing Rules	10
3.1.5.1	Realm Autodiscovery Through HTTP 401 Challenge	10
3.1.5.2	Server-to-Server Security Token Contents	10
3.1.6	Timer Events.....	10
3.1.7	Other Local Events.....	10
3.2	Server Details.....	10
3.2.1	Abstract Data Model.....	10
3.2.2	Timers	11
3.2.3	Initialization	11
3.2.4	Higher-Layer Triggered Events	11
3.2.5	Message Processing Events and Sequencing Rules	11
3.2.5.1	Authentication Within a Single Organization	11
3.2.5.2	Authentication with User Information Within a Single Organization.....	11
3.2.5.3	Authentication with Third-Party Application	12
3.2.5.4	Realm Autodiscovery Through HTTP 401 Challenge	13
3.2.5.5	Server-to-Server Security Token Contents	13
3.2.5.6	Server-to-Server Validation Criteria.....	13
3.2.6	Timer Events.....	14
3.2.7	Other Local Events.....	14
4	Protocol Examples	15
4.1	Security Token Issued by STS	15
4.2	Security Token Self-Issued by Client	15
4.3	Security Token Issued by STS with User Information Added by Client	16
4.4	Security Token Self-Issued By Client with User Information.....	17
4.5	Security Token for Accessing a Third-Party Service with Extensions	18
4.6	Realm Autodiscovery Through HTTP 401 Challenge	19
5	Security	20
5.1	Security Considerations for Implementers	20
5.2	Index of Security Parameters	20
6	Appendix A: Product Behavior	21

7	Change Tracking.....	22
8	Index.....	23

1 Introduction

The OAuth 2.0 Authorization Protocol Extensions extend the OAuth 2.0 Authentication Protocol and the JSON Web Token (JWT) to enable server-to-server authentication.

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

1.1 Glossary

This document uses the following terms:

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

private key: One of a pair of keys used in public-key cryptography. The private key is kept secret and is used to decrypt data that has been encrypted with the corresponding public key. For an introduction to this concept, see [\[CRYPTO\]](#) section 1.8 and [\[IEEE1363\]](#) section 3.1.

public key: One of a pair of keys used in public-key cryptography. The public key is distributed freely and published as part of a digital certificate. For an introduction to this concept, see [\[CRYPTO\]](#) section 1.8 and [\[IEEE1363\]](#) section 3.1.

realm: An administrative boundary that uses one set of authentication servers to manage and deploy a single set of unique identifiers. A realm is a unique logon space.

realm autodiscovery: A process used by client applications to obtain the name of a server resource's source **realm** and then use that information to locate a **security token service (STS)** that can issue access tokens to the resource.

security principal: A unique entity that is identifiable through cryptographic means by at least one key. It frequently corresponds to a human user, but also can be a service that offers a resource to other security principals. Also referred to as principal.

security principal identifier: A value that is used to uniquely identify a **security principal**. In Windows-based systems, it is a security identifier (SID). In other types of systems, it can be a user identifier or other type of information that is associated with a **security principal**.

security token: An opaque message or data packet produced by a Generic Security Services (GSS)-style authentication package and carried by the application protocol. The application has no visibility into the contents of the token.

security token service (STS): A web service that issues claims (2) and packages them in encrypted security tokens.

Transmission Control Protocol (TCP): A protocol used with the Internet Protocol (IP) to send data in the form of message units between computers over the Internet. TCP handles keeping

track of the individual units of data (called packets) that a message is divided into for efficient routing through the Internet.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

user principal name (UPN): A user account name (sometimes referred to as the user logon name) and a domain name that identifies the domain in which the user account is located. This is the standard usage for logging on to a Windows domain. The format is: someone@example.com (in the form of an email address). In Active Directory, the userPrincipalName attribute (2) of the account object, as described in [\[MS-ADTS\]](#).

X.509: An ITU-T standard for public key infrastructure subsequently adapted by the IETF, as specified in [\[RFC3280\]](#).

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

1.2 References

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

1.2.1 Normative References

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

[IETF-DRAFT-JWT-LATEST] Jones, M., Bradley, J., and Sakimura, N., "JSON Web Token (JWT) draft-ietf-oauth-json-web-token-08", draft-ietf-oauth-json-web-token-08, May 2013, <http://datatracker.ietf.org/doc/draft-ietf-oauth-json-web-token/>

[MS-DTYP] Microsoft Corporation, "[Windows Data Types](#)".

[MS-OAUTH2EX] Microsoft Corporation, "[OAuth 2.0 Authentication Protocol Extensions](#)".

[MS-ODATA] Microsoft Corporation, "[Open Data Protocol \(OData\)](#)".

[MS-SPSTWS] Microsoft Corporation, "[SharePoint Security Token Service Web Service 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.rfc-editor.org/rfc/rfc2616.txt>

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., et al., "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999, <http://www.rfc-editor.org/rfc/rfc2617.txt>

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

[RFC793] Postel, J., Ed., "Transmission Control Protocol: DARPA Internet Program Protocol Specification", RFC 793, September 1981, <http://www.rfc-editor.org/rfc/rfc793.txt>

1.2.2 Informative References

[MS-SPS2SAUTH] Microsoft Corporation, "[OAuth 2.0 Authentication Protocol: SharePoint Profile](#)".

1.3 Overview

These extensions specify how applications can perform server-to-server authentication using a **security token service (STS)**. For example, an email service might use these extensions to authenticate itself when it makes a call to an instant messaging service. Both of these services are server applications. However, in the scope of this protocol, the email service would be the client, and the instant messaging service would be the server. For an example of a server-to-server **security token** that a client might send to authenticate itself, see section [4.2](#).

1.4 Relationship to Other Protocols

These extensions extend the OAuth 2.0 Authentication Protocol, as described in [\[MS-OAUTH2EX\]](#), and JSON Web Token (JWT), as described in [\[IETFDRIFT-JWT-LATEST\]](#).

For information on how to implement an STS, see [\[MS-SPSTWS\]](#).

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

1.5 Prerequisites/Preconditions

The client is required to reside in the same **realm** as the STS to request a server-to-server security token from it.

1.6 Applicability Statement

These extensions apply only when a service call is made to or from an application that supports server-to-server authentication.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

These extensions transport messages over **TCP**, as specified in [\[RFC793\]](#), and do not pass any specific parameters to the transport. Transport-layer security **MUST** be used to secure the security tokens. These extensions use **HTTPS**, as specified in [\[RFC2818\]](#), to do so.

Messages sent using these extensions are not encoded by Open-Data, as specified in [\[MS-ODATA\]](#), and use the default character set defined by the client or the server.

Security tokens are JWTs and are sent using **HTTP Authorization** headers, as specified in [\[IETFdraft-jwt-latest\]](#). **HTTP Authorization** headers are specified in [\[RFC2617\]](#).

2.2 Message Syntax

A **security principal** is represented as a **security principal identifier** in the messages sent by applications. A security principal identifier is a **GUID**.

For more details about the messages typically exchanged between a client and an STS, see [\[MS-SPSTWS\]](#).

For clarity, this document uses different names to refer to the server-to-server security tokens that are exchanged in various scenarios. An actor token is a signed security token that is issued by an STS, or by the client itself if the server trusts it to do so. An outer token is an unsigned security token that is constructed by the client and contains user information in addition to an actor token. In this scenario, the actor token is referred to as the inner token. All of these security tokens are formatted in the same way, as specified in [\[IETFdraft-jwt-latest\]](#), and contain the claims and header fields specified in this section.

The following table describes claims that are exchanged in server-to-server security tokens. The claim values are all of data type **STRING**, as specified in [\[MS-DTYP\]](#).

Claim type	Claim value description	Example claim values
aud	The targeted service for which the client issued the server-to-server security token.	<security principal identifier>/<hostname>@<realm>
iss	The security principal identifier of the server-to-server security token issuer.	<security principal identifier>@<realm>
nameid	The logged on user's user principal name (UPN) value for the security principal that made the request.	user@contoso.com
nbf	The time at which the server-to-server security token was created.	129592882368666656
exp	The time at which the server-to-server security token expires.	129592882368666656
trustedfordelegation	"true" if the client is trusted to delegate a user identity; otherwise, "false".	true false
identityprovider	The identity provider that authenticated the caller.	windows forms trusted

Claim type	Claim value description	Example claim values
actort	The security token issued and signed by the STS. An actor token has the same format as any other security token.	See section 4.3 and section 4.4 .
smtp	The logged on user's email address.	user@contoso.com
sip	The logged on user's sip address.	user@contoso.com
msexchuid	A unique identifier that the STS can give the user. This is an additional claim that the STS adds and is not required by the OAuth 2.0 Authentication Protocol, as specified in [MS-OAUTH2EX] .	objectGUID@contoso.com
appctx	The application context. This claim contains a subset of claims that is specific to the service accessed by the client.	See section 4.5 .

The following list describes the header fields in a server-to-server security token. The field values are all of data type **STRING**, as specified in [MS-DTYP].

- **typ**. The token type. The value MUST always be "JWT".
- **alg**. The algorithm used to encrypt the contents of the token. The value of this field MUST be either "none" or "rs256". Actor tokens are always signed and have **alg** fields that contain the value "rs256". Outer tokens that contain inner signed tokens, as described in section 4.3 and section 4.4, are not signed and have **alg** fields that contain the value "none".
- **x5t**. The **base64** encoded thumbprint of the certificate used to sign the security token. This field is optional.

The header fields are contained in a separate part of the security token, as specified in [IETF-DRAFT-JWT-LATEST].

3 Protocol Details

3.1 Client Details

Note that the client and server are in fact both server applications. However, in the scope of this protocol, one server application acts as the client, and one server application acts as the server.

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 Realm Autodiscovery Through HTTP 401 Challenge

A client can use **realm autodiscovery** to obtain the name of its realm, which it then uses to locate an STS. It uses realm autodiscovery by sending a request to the server that contains an empty **Bearer HTTP Authorization** header to the server. The **HTTP Authorization** header is specified in [\[RFC2617\]](#).

For an example of realm autodiscovery through HTTP 401 challenge, see section [4.6](#).

3.1.5.2 Server-to-Server Security Token Contents

The server-to-server security token sent by the client MUST be compatible with the JWT format specified in [\[IETFDRAFT-JWT-LATEST\]](#) and [\[MS-OAUTH2EX\]](#).

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

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 Authentication Within a Single Organization

The following procedure shows the authentication that takes place when a client makes a call to a server in the same organization using these extensions.

1. The organization's IT administrator sets up an STS and configures it with the security principal identifiers for the client and server. The client and server each exchange **public keys**, carried in **X.509** certificates, with the STS. The administrator also configures the client and server to trust security tokens issued by the STS.
2. The client makes an anonymous request to the server.
3. The server responds with an HTTP 401 challenge. HTTP 401 is specified in [\[RFC2616\]](#) and [\[RFC2617\]](#).
4. The client requests a security token from the STS. It does this by sending a self-issued security token that is signed with its **private key**. The security token contains the **aud**, **iss**, **nameid**, **nbf**, **exp**, and **trustedfordelegation** claims as specified in section [2.2](#). The client request also includes a *resource* parameter and a *realm* parameter, as specified in [\[MS-OAUTH2EX\]](#). The value of the *resource* parameter is the **Uniform Resource Identifier (URI)** of the server. For an example of a self-issued security token, see section [4.2](#).
5. The STS validates the public key of the security token provided by the client, verifies that the client is authorized to access the requested resource, and responds to the client with a server-to-server security token that is signed with a public key that the server trusts. The security token contains the **aud**, **iss**, **nameid**, **nbf**, **exp**, and **identityprovider** claims, as specified in section [2.2](#). For an example of a server-to-server security token issued by an STS, see section [4.1](#).
6. The client sends the server-to-server security token to the server.
7. The server validates the server-to-server security token by checking the values of the **aud**, **iss**, and **exp** claims and the public key provided by the STS. It performs additional validation checks to ensure that the client is authorized to access the requested resource. It then responds to the client with the requested resource.

3.2.5.2 Authentication with User Information Within a Single Organization

The following procedure shows the authentication that takes place when a client makes a call to a server in the same organization using these extensions. The client also sends user information to the server, and the server uses the information to determine whether to return the requested resource.

1. The organization's IT administrator sets up an STS and configures it with the security principal identifiers for the client and server. The client and server each exchange public keys, carried in

X.509 certificates, with the STS. The administrator also configures the client and server to trust security tokens issued by the STS.

2. The client makes an anonymous request to the server.
3. The server responds with an HTTP 401 challenge, as specified in [\[RFC2616\]](#) and [\[RFC2617\]](#).
4. The client requests a security token from the STS. It does this by sending a self-issued security token that is signed with its private key. The security token contains the **aud**, **iss**, **nameid**, **trustedfordelegation**, **nbfi**, and **exp** claims, as specified in section [2.2](#). The client request also includes a *resource* parameter and a *realm* parameter, as specified in [\[MS-OAUTH2EX\]](#). The value of the *resource* parameter is the URI of the server. For an example of a self-issued security token, see section [4.2](#).
5. The STS validates the public key of the security token provided by the client, verifies that the client is authorized to access the requested resource, and responds to the client with a server-to-server security token that is signed with a public key that the server trusts. The security token contains the **aud**, **iss**, **nameid**, **nbfi**, and **exp** claims. For an example of a server-to-server security token issued by an STS that contains user information, see section [4.3](#).
6. The client sends a self-issued security token to the server that includes the server-to-server security token it received from the STS, as well as additional claims that contain the user information. The additional claims are the **aud**, **iss**, **nameid**, **nbfi**, **exp**, **smtip**, **sip**, **msexchuid**, and **actort** claims, as specified in section [2.2](#). The **actort** claim contains the server-to-server security token provided by the STS. Note that the server-to-server security token is signed, but the user information is not. For an example of a self-issued server-to-server security token that contains user information, see section [4.4](#).
7. The server validates the request by checking the user information contained in the **aud**, **iss**, and **exp** claims and the public key used to sign the security token provided by the STS. Because the user information is not signed, the server validates the user information by checking that the values of the **aud** and **iss** claims in the user information match the values of the **aud** and **iss** claims in the server-to-server security token contained in the **actort** claim. For security considerations regarding the unsigned user information, see section [5.1](#).
8. The server performs additional validation checks to ensure that the client is authorized to access the requested resource. It then responds to the client with the requested resource.

3.2.5.3 Authentication with Third-Party Application

The following procedure shows the authentication that takes place when a client makes a call to a third-party application in the same organization using these extensions.

1. The organization's IT administrator sets up an STS and configures it with the security principal identifiers for the client and third-party application. The client and third-party application each exchange public keys, carried in X.509 certificates, with the STS. The administrator also configures the client and third-party application to trust security tokens issued by the STS.
2. The client makes an anonymous request to the third-party application.
3. The third-party application responds with an HTTP 401 challenge, as specified in [\[RFC2616\]](#) and [\[RFC2617\]](#).
4. The client requests a security token from the STS. It does this by sending a self-issued security token that is signed with its private key. The security token contains the **aud**, **iss**, **nameid**, **nbfi**, and **exp** claims, as specified in section [2.2](#).
5. The STS validates the public key of the security token provided by the client, verifies that the client is authorized to access the requested resource, and returns a server-to-server security

token that is signed with a public key that the third-party application trusts. The security token contains the **aud**, **iss**, **nameid**, **nbf**, **exp**, and **appctx** claims, as specified in section 2.2. The **appctx** claim contains information that is implementation-specific to the third-party application. For an example of a server-to-server security token that is used to access a third-party application, see section [4.5](#).

6. The client sends the server-to-server security token to the third-party application.
7. The third-party application validates the server-to-server security token by checking the values of the **aud**, **iss**, and **exp** claims and the public key provided by the STS. It performs additional validation checks to ensure that the client is authorized to access the requested resource. It then responds to the client with the requested resource.

3.2.5.4 Realm Autodiscovery Through HTTP 401 Challenge

When a server receives a request that contains an empty **Bearer HTTP Authorization** header, the server responds with an HTTP 401 challenge, as specified in [\[RFC2616\]](#) and [\[RFC2617\]](#). The **Bearer WWW-Authenticate** header of the HTTP 401 challenge contains the following fields:

- **client_id**. The client's security principal identifier.
- **realm**. The server MAY [<1>](#) return this field. This is the source realm of the client.
- **trusted_issuers**. A comma-separated list of all security token issuers the server trusts. The client can then select a security token issuer to request a security token.

For an example of realm autodiscovery through HTTP 401 challenge, see section [4.6](#).

3.2.5.5 Server-to-Server Security Token Contents

The server can accept server-to-server security tokens with the claim types specified in section [2.2](#).

3.2.5.6 Server-to-Server Validation Criteria

The server accepts a server-to-server security token that meets the following criteria:

- The server-to-server security token is signed with a trusted signing certificate from an STS that the server trusts.
- The server-to-server security token contains an **iss** claim whose value shows that the security token is issued by an STS that the server trusts.
- The server-to-server security token contains a **nameid** claim with the UPN value of the logged-on user.
- If the client constructs an unsigned outer security token to contain user information as well as a signed actor token (that is, an inner token), as described in section [4.3](#) and section [4.4](#), the value of the **iss** claim in the outer token matches the value of the **nameid** claim in the inner token. The server performs a case-sensitive comparison.
- The server-to-server security token contains an **aud** claim whose value meets the following criteria:
 - The **aud** claim value MUST contain three parts: **client_id**, **hostname**, and **realm**.
 - The value of the **client_id** part is the security principal identifier of a security principal that the server trusts. The server performs a case-sensitive comparison.

- The value of the hostname part is the host name of the server. The server performs a case-insensitive comparison to verify that it is the target of the request.
- The value of the realm part is the source realm. The server performs a case-sensitive comparison.

The STS uses the claims in the server-to-server security token to authenticate the caller.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 Security Token Issued by STS

In this example, a client attempts to access a resource on the server. The server responds with an HTTP 401 challenge that lists the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```
HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0
```

1. The client sends its credentials to the indicated token issuer, which is an STS.
2. The STS authenticates the client and issues an actor token to the client.
3. The client uses the actor token to access the resource it requested on the server.

The following is an example of an actor token issued by an STS. For more information about the claim values contained in this security token, see section [2.2](#).

```
actor:
{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "XqrnFEfsS55_vMBpHvF0pTnqeaM"
}.{
  "aud": "00000003-0000-0ff1-ce00-000000000000/contoso.com@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "iss": "00000001-0000-0000-c000-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "nbf": "1323380070",
  "exp": "1323383670",
  "nameid": "00000002-0000-0ff1-ce00-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "identityprovider": "00000001-0000-0000-c000-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8"
}
```

4.2 Security Token Self-Issued by Client

In this example, the client tries to access a resource on the server. The server responds with an HTTP 401 challenge that indicates the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```
HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
```


Content-Length: 0

1. The client is one of the token issuers trusted by the server, so it creates an actor token and signs it with its credentials.
2. The client uses the actor token to access the resource it requested on the server.

The following is an example of an actor token that is self-issued by a client. For more information about the claim values contained in this security token, see section [2.2](#).

```
actor:
{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "mH-TTlt-HAXC9-vjKVfTX6bAsR0"
}. {
  "aud": "00000003-0000-0ff1-ce00-000000000000/contoso.com@EXHB-88371dom.extest.contoso.com",
  "iss": "00000002-0000-0ff1-ce00-000000000000@EXHB-88371dom.extest.contoso.com",
  "nbf": "1323380605",
  "exp": "1323409405",
  "nameid": "00000002-0000-0ff1-ce00-000000000000@EXHB-88371dom.extest.contoso.com",
  "trustedfordelegation": "true"
}
```

4.3 Security Token Issued by STS with User Information Added by Client

In this example, the client tries to access a resource on the server. The server responds with an HTTP 401 challenge that lists the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```
HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0
```

1. The client sends its credentials to the indicated security token issuer, which is an STS.
2. The STS authenticates the client and issues an actor token to the client.
3. The client constructs an unsigned outer token that contains additional user information to provide to the server. The outer token also contains the signed actor token issued by the STS.
4. The client uses the outer token to access the resource it requested on the server.

The following is an example of an outer token that is constructed by the client and contains user information, as well as an actor token issued by an STS. For more information about the claim values contained in this security token, see section [2.2](#).

```
{
  "typ": "JWT",
  "alg": "none"
}. {
```

```

    "aud": "00000003-0000-0ff1-ce00-000000000000/fabrikam.com@fabrikam-oauth-
929.extest.fabrikam.com",
    "iss": "00000001-0000-0ff1-ce00-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
    "nbf": "1323380069",
    "exp": "1323408869",
    "nameid": "00000002-0000-0ff1-ce00-000000000000@BLID-EXHB-90232dom.extest.contoso.com
    "actort": "..actor token.."
}

```

The following is an example of an actor token, as mentioned in the previous listing.

```

{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "hEAW-SXzTNaDBUwfAh2YScnBOxA"
}. {
  "aud": "00000002-0000-0ff1-ce00-000000000000/contoso.com@EXHB-
88371dom.extest.contoso.com",
  "iss": "00000003-0000-0ff1-ce00-000000000000@e54c2f60-0ad3-4ef8-8ba2-b3ae01b35494",
  "nbf": "1346674665",
  "exp": "1346804265",
  "nameid": "00000003-0000-0ff1-ce00-000000000000@e54c2f60-0ad3-4ef8-8ba2-b3ae01b35494"
}

```

4.4 Security Token Self-Issued By Client with User Information

In this example, the client tries to access a resource on the server. The server responds with an HTTP 401 challenge that indicates the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```

HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-cl69a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0

```

1. The client is one of the token issuers trusted by the server, so it creates an actor token and signs it with its credentials.
2. The client constructs an unsigned outer token that contains additional user information to provide to the server. The outer token also contains the signed actor token issued by the client.
3. The client uses the outer token to access the resource it requested on the server.

The following is an example of an outer token that is constructed by the client and contains user information, as well as a security token that is self-issued by the client. For more information about the claim values contained in this security token, see section [2.2](#).

```

{
  "typ": "JWT",
  "alg": "none"
}. {
  "aud": "00000003-0000-0ff1-ce00-000000000000/contoso.com@EXHB-
88371dom.extest.contoso.com",

```

```

    "iss": "00000002-0000-0ff1-ce00-000000000000@EXHB-88371dom.extest.contoso.com",
    "nbf": "1323380605",
    "exp": "1323409405",
    "nameid": "ewsuser-55a83300@EXHB-88371dom.extest.contoso.com",
    "smtp": "ewsuser-55a83300@exhb-88371dom.extest.contoso.com",
    "sip": "ewsuser-55a83300@exhb-88371dom.extest.contoso.com",
    "msexchuid": "842e4c3a-0879-4973-83f9-495bb9863e18@exhb-88371dom.extest.contoso.com",
    "actort": "..actor token.."
}

```

The following is an example of an actor token, as mentioned in the previous listing.

```

{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "hEAW-SXzTNaDBUwfAh2YScnBOxA"
}. {
  "aud": "00000002-0000-0ff1-ce00-000000000000/contoso.com@EXHB-88371dom.extest.contoso.com",
  "iss": "00000003-0000-0ff1-ce00-000000000000@e54c2f60-0ad3-4ef8-8ba2-b3ae01b35494",
  "nbf": "1346674665",
  "exp": "1346804265",
  "nameid": "00000003-0000-0ff1-ce00-000000000000@e54c2f60-0ad3-4ef8-8ba2-b3ae01b35494"
}

```

4.5 Security Token for Accessing a Third-Party Service with Extensions

In this example, the client tries to access a resource on a third-party service. The service responds with an HTTP 401 challenge that lists the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```

HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: DUXYI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0

```

1. The client sends its credentials to the indicated security token issuer, which is an STS. The credentials include an **appctx** claim that contains additional user information to be provided to the service. The STS does not examine or modify the user information in the **appctx** claim.
2. The STS authenticates the client and issues an actor token to the client that includes the **appctx** claim provided by the client.
3. The client uses the actor token to access the resource it requested on the service. The actor token includes the **appctx** claim that was previously supplied by the client and contains additional user information to be provided to the service. This scenario is analogous to the use of outer and inner tokens to convey additional user information to a server beyond what is required to authenticate to an STS, as described in section [4.3](#) and section [4.4](#).

The following is an example of a security token that is used to access a third-party service and contains user information. For more information about the claim values contained in this security token, see section [2.2](#).

```
{
  "aud": "00000002-0000-0ff1-ce00-000000000000/exhb-42702.exhb-42702dom.extest.contoso.com@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "iss": "00000002-0000-0ff1-ce00-000000000000@b84c5afe-7ced-4ce8-aa0b-df0e2869d3c8",
  "nbf": "1323197012",
  "exp": "1323225812",
  "nameid": "https://www.fabrikam.com/weprintem",
  "appctx": "{
    "nameid": "ewsuser-cff3d495@BLID-EXHB-42702dom.extest.contoso.com",
    "smtp": "ewsuser-cff3d495@BLID-EXHB-42702dom.extest.contoso.com",
    "msexchuid": "842e4c3a-0879-4973-83f9-495bb9863e18@exhb-88371dom.extest.contoso.com"
  }"
}
```

4.6 Realm Autodiscovery Through HTTP 401 Challenge

In this example, the client tries to access a resource on a server. It also tries to use realm autodiscovery by including an empty **Bearer** authorization header in its request. An example of such a request is as follows.

```
POST https://contoso.com/autodiscover/autodiscover.svc HTTP/1.1
Content-Type: text/xml; charset=utf-8
Accept: text/xml
User-Agent: Test/1.0 (ContosoServicesClient/15.00.0424.000)
client-request-id: 00000000-0000-0000-0000-000000000000
Authorization: Bearer
Host: contoso.com
Content-Length: 1368
Expect: 100-continue
```

The server responds with an HTTP 401 challenge that lists the security token issuers it trusts in the **trusted_issuers** field. An example of such a challenge is as follows.

```
HTTP/1.1 401 Unauthorized
Server: Fabrikam/7.5
request-id: 443ce338-377a-4c16-b6bc-c169a75f7b00
X-FEServer: XJSUI01CA101
WWW-Authenticate: Bearer client_id="00000002-0000-0ff1-ce00-000000000000",
trusted_issuers="00000001-0001-0000-c000-000000000000@"
WWW-Authenticate: Basic Realm=""
X-Powered-By: ASP.NET
Date: Thu, 19 Apr 2012 17:04:16 GMT
Content-Length: 0
```

In this example, the server determines that the value in the **trusted_issuers** field contains sufficient information for the client to locate the STS, so the server does not include a **realm** field.

5 Security

5.1 Security Considerations for Implementers

The security considerations described in [\[MS-SPS2SAUTH\]](#) apply when implementing these extensions.

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
- Microsoft Exchange Server 2016
- Microsoft SharePoint Server 2013
- Microsoft SharePoint Foundation 2013
- Microsoft SharePoint Server 2016

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

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

[<1> Section 3.2.5.4](#): SharePoint Server 2013, SharePoint Foundation 2013, and SharePoint Server 2016 return this parameter.

7 Change Tracking

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

8 Index

A

Abstract data model

[client](#) 10

[server](#) 10

[Applicability](#) 7

[Authentication third-party application](#) 12

[Authentication with user information within a single organization](#) 11

[Authentication within a single organization](#) 11

C

[Capability negotiation](#) 7

[Change tracking](#) 22

Client

[abstract data model](#) 10

[higher-layer triggered events](#) 10

[initialization](#) 10

[other local events](#) 10

[timer events](#) 10

[timers](#) 10

D

Data model - abstract

[client](#) 10

[server](#) 10

E

Examples

[security token for accessing a third-party service with extensions](#) 18

[security token issued by STS](#) 15

[security token issued by STS with user information added by client](#) 16

[security token self-issued by client application](#) 15

[security token self-issued by client application with user information](#) 17

F

[Fields - vendor-extensible](#) 7

G

[Glossary](#) 5

H

Higher-layer triggered events

[client](#) 10

[server](#) 11

I

[Implementer - security considerations](#) 20

[Index of security parameters](#) 20

[Informative references](#) 7

Initialization

[client](#) 10

[server](#) 11

[Introduction](#) 5

M

Message processing - client

[realm autodiscovery through HTTP 401 challenge](#) 10

[server-to-server token contents](#) 10

Message processing - server

[authentication with third-party application](#) 12

[authentication with user information within a single organization](#) 11

[authentication within a single organization](#) 11

[realm autodiscovery through HTTP 401 challenge](#) 13

[server-to-server security token contents](#) 13

[server-to-server validation criteria](#) 13

Messages

[syntax](#) 8

[transport](#) 8

N

[Normative references](#) 6

O

Other local events

[client](#) 10

[server](#) 14

[Overview \(synopsis\)](#) 7

P

[Parameters - security index](#) 20

[Preconditions](#) 7

[Prerequisites](#) 7

[Product behavior](#) 21

R

Realm autodiscovery through HTTP 401 challenge

[client](#) 10

[server](#) 13

References

[informative](#) 7

[normative](#) 6

[Relationship to other protocols](#) 7

S

Security

[implementer considerations](#) 20

[parameter index](#) 20

[Security token for accessing a third-party service with extensions example](#) 18

[Security token issued by STS example](#) 15

[Security token issued by STS with user information added by client example](#) 16

[Security token self-issued by client application example](#) 15

[Security token self-issued by client application with user information example](#) 17
Sequencing rules - client
 [realm autodiscovery through HTTP 401 challenge](#) 10
 [server-to-server token contents](#) 10
Sequencing rules - server
 [authentication with third-party application](#) 12
 [authentication with user information within a single organization](#) 11
 [authentication within a single organization](#) 11
 [realm autodiscovery through HTTP 401 challenge](#) 13
 [server-to-server security token contents](#) 13
 [server-to-server validation criteria](#) 13
Server
 [abstract data model](#) 10
 [higher-layer triggered events](#) 11
 [initialization](#) 11
 [other local events](#) 14
 [timer events](#) 14
 [timers](#) 11
[Server-to-server security token contents - server](#) 13
Server-to-server token contents
 [client](#) 10
[Server-to-server validation criteria](#) 13
[Standards assignments](#) 7
[Syntax - messages](#) 8

T

Timer events
 [client](#) 10
 [server](#) 14
Timers
 [client](#) 10
 [server](#) 11
[Tracking changes](#) 22
[Transport - messages](#) 8

V

[Vendor-extensible fields](#) 7
[Versioning](#) 7

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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
Tag										Length										Value (variable)											
...																															

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```



```

parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR

```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWw0SE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line  
Response-headers  
CR/LF  
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request


```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOF): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Revision Summary

Date	Revision History	Revision Class	Comments
12/3/2008	1.0.0	Major	Initial release.
2/4/2009	1.0.1	Editorial	Revised and edited technical content.
3/4/2009	1.0.2	Editorial	Revised and edited technical content.
4/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	5.0.0	Major	Updated and revised the technical content.
5/5/2010	6.0.0	Major	Updated and revised the technical content.
8/4/2010	7.0	Major	Significantly changed the technical content.
11/3/2010	7.1	Minor	Clarified the meaning of the technical content.
3/18/2011	8.0	Major	Significantly changed the technical content.
8/5/2011	8.1	Minor	Clarified the meaning of the technical content.
10/7/2011	8.2	Minor	Clarified the meaning of the technical content.
1/20/2012	9.0	Major	Significantly changed the technical content.
4/27/2012	9.1	Minor	Clarified the meaning of the technical content.
7/16/2012	10.0	Major	Significantly changed the technical content.
10/8/2012	11.0	Major	Significantly changed the technical content.
2/11/2013	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	12.0	Major	Significantly changed the technical content.
11/18/2013	12.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	12.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	13.0	Major	Significantly changed the technical content.
7/31/2014	13.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	13.1	Minor	Clarified the meaning of the technical content.
5/26/2015	14.0	Major	Significantly changed the technical content.
6/30/2015	15.0	Major	Significantly changed the technical content.
9/14/2015	16.0	Major	Significantly changed the technical content.
6/9/2016	17.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
2/28/2017	18.0	Major	Significantly changed the technical content.
9/19/2017	19.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	7
1.1	Glossary	7
1.2	References	9
1.2.1	Normative References	9
1.2.2	Informative References	10
1.3	Overview	10
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
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	HTTP POST Request	12
2.2.1.1	Request Format	12
2.2.1.1.1	Request Line.....	12
2.2.1.1.1.1	Base64-Encoded Query Value.....	13
2.2.1.1.1.1.1	Encoded Parameter.....	14
2.2.1.1.1.1.2	Command Codes	14
2.2.1.1.1.1.3	Command Parameters.....	15
2.2.1.1.1.2	Plain Text Query Value	16
2.2.1.1.1.2.1	Command.....	17
2.2.1.1.1.2.2	User Name	17
2.2.1.1.1.2.3	Device ID	17
2.2.1.1.1.2.4	Device type	17
2.2.1.1.1.2.5	Command-Specific URI Parameters.....	17
2.2.1.1.2	Request Headers	18
2.2.1.1.2.1	Accept-Language	18
2.2.1.1.2.2	Authorization.....	18
2.2.1.1.2.3	Content-Type	19
2.2.1.1.2.4	Cookie.....	19
2.2.1.1.2.5	MS-ASAcceptMultiPart	19
2.2.1.1.2.6	MS-ASProtocolVersion	20
2.2.1.1.2.7	User-Agent	20
2.2.1.1.2.8	X-MS-PolicyKey	20
2.2.1.1.3	Request Body.....	20
2.2.2	HTTP POST Response	20
2.2.2.1	Response Format	20
2.2.2.1.1	Status Line	20
2.2.2.1.2	Response Headers	21
2.2.2.1.2.1	Cache-Control	23
2.2.2.1.2.2	Content-Encoding	23
2.2.2.1.2.3	Content-Length	23
2.2.2.1.2.4	Content-Type	23
2.2.2.1.2.5	MS-Server-ActiveSync	23
2.2.2.1.2.6	X-MS-Location	23
2.2.2.1.2.7	MS-ASProtocolCommands	23
2.2.2.1.2.8	MS-ASProtocolVersions	24
2.2.2.1.2.9	X-MS-RP	24
2.2.2.1.2.10	X-MS-Credential-Service-Url.....	24
2.2.2.1.2.11	X-MS-Credentials-Expire.....	24
2.2.2.1.2.12	Set-Cookie	24

2.2.2.1.2.13	X-MS-AShrottle	24
2.2.2.1.2.14	X-BEServer	24
2.2.2.1.2.15	X-FEServer	24
2.2.2.1.2.16	request-id	25
2.2.2.1.3	Response Body	25
2.2.3	HTTP OPTIONS Request	25
2.2.3.1	Request Format	25
2.2.3.1.1	Request Line	25
2.2.3.1.2	Request Headers	25
2.2.4	HTTP OPTIONS Response	25
2.2.4.1	Response Format	26
2.2.4.1.1	Status Line	26
2.2.4.1.2	Response Headers	26
2.2.4.1.2.1	MS-ASProtocolCommands	26
2.2.4.1.2.2	MS-ASProtocolVersions	26
2.2.4.1.2.3	Set-Cookie	26
3	Protocol Details	27
3.1	Client Details	27
3.1.1	Abstract Data Model	27
3.1.2	Timers	27
3.1.3	Initialization	27
3.1.4	Higher-Layer Triggered Events	27
3.1.4.1	Sending a Command Request	27
3.1.5	Message Processing Events and Sequencing Rules	27
3.1.5.1	Handling a Successful Response	27
3.1.5.2	Handling a Failed Response	28
3.1.5.2.1	HTTP Error 401, 403, and 500	28
3.1.5.2.2	HTTP Error 451	28
3.1.5.2.3	HTTP Error 503	28
3.1.5.2.4	HTTP Error 456 and 457	29
3.1.6	Timer Events	29
3.1.7	Other Local Events	29
3.2	Server Details	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	29
3.2.3	Initialization	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Handling HTTP POST Command Requests	30
3.2.5.1.1	User-Agent Change Tracking	30
3.2.5.2	Handling HTTP OPTIONS Command Requests	30
3.2.6	Timer Events	31
3.2.7	Other Local Events	31
4	Protocol Examples	32
4.1	FolderSync Request and Response	32
4.2	FolderSync Request and Redirect Response	32
4.3	HTTP OPTIONS Command Request and Response	33
4.4	SendMail Request and Response	33
4.5	CreateFolder Request and Response	34
5	Security	36
5.1	Security Considerations for Implementers	36
5.2	Index of Security Parameters	36
6	Appendix A: Product Behavior	37
7	Change Tracking	39

8	Index.....	40
----------	-------------------	-----------

1 Introduction

The Exchange ActiveSync: HTTP Protocol enables client devices to synchronize data with the data that is stored on the server.

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

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [\[RFC5234\]](#).

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [\[RFC4648\]](#).

calendar: A date range that shows availability, **meetings**, and appointments for one or more users or resources. See also Calendar object.

contact: (1) A presence entity (presentity) whose presence information can be tracked.

(2) An object of the contact class that represents a company or person whom a user can contact.

encrypted message: An Internet email message that is in the format described by [\[RFC5751\]](#) and uses the EnvelopedData CMS content type described in [\[RFC3852\]](#), or the **Message object** that represents such a message.

Global Address List (GAL): An address list that conceptually represents the default address list for an address book.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [\[RFC4122\]](#) or [\[C706\]](#) must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [\[SSL3\]](#) and [\[RFC5246\]](#).

Inbox folder: A special folder that is the default location for **Message objects** received by a user or resource.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

meeting: An event with attendees.

meeting request: An instance of a Meeting Request object.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

MIME message: A message that is as described in [\[RFC2045\]](#), [\[RFC2046\]](#), and [\[RFC2047\]](#).

OAuth: The OAuth 2.0 authorization framework [\[RFC6749\]](#).

Out of Office (OOO): One of the possible values for the free/busy status on an appointment. It indicates that the user will not be in the office during the appointment.

plain text: Text that does not have markup. See also plain text message body.

recipient: An entity that can receive email messages.

S/MIME (Secure/Multipurpose Internet Mail Extensions): A set of cryptographic security services, as described in [\[RFC5751\]](#).

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication using X.509 certificates. For more information, see [\[X509\]](#). The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [\[SSL3\]](#).

Sent Items folder: A special folder that is the default location for storing copies of **Message objects** after they are submitted or sent.

server ID: A unique identifier that is assigned by the server to each object that can be synchronized. A client stores the server ID for each object and is able to locate an object when given a server ID.

SSL/TLS handshake: The process of negotiating and establishing a connection protected by **Secure Sockets Layer (SSL)** or **Transport Layer Security (TLS)**. For more information, see [\[SSL3\]](#) and [\[RFC2246\]](#).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [\[X509\]](#)). **TLS** is standardized in the IETF TLS working group.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [\[RFC3986\]](#).

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [\[RFC1738\]](#).

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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									
Protocol version								Command code								Locale																								
Device ID length								Device ID (variable)																																
...																																								
Policy key length								Policy key (optional)																																
...								Device type length								Device type (variable)																								
...																																								

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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						
Tag										Length										Value (variable)																	
...																																					

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOF) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
*('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```

parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR

```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line  
Response-headers  
CR/LF  
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server **MUST** conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it **SHOULD** include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it **MAY** [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it **MUST** include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server **SHOULD** include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server **SHOULD** include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server **MAY** [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking **SHOULD** use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers **SHOULD** limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but **MAY** [<9>](#) use different values for the number of changes or the time period. The server **SHOULD** block clients that exceed this limit for 14 hours, but **MAY** [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response **MUST** contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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
Protocol version								Command code								Locale															
Device ID length								Device ID (variable)																							
...																															
Policy key length								Policy key (optional)																							
...								Device type length								Device type (variable)															
...																															

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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						
Tag										Length										Value (variable)																	
...																																					

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec             = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name          = 1*ALPHA
user-name             = 1*VCHAR
device-id             = 1*32 (ALPHA / DIGIT)
device-type           = 1*VCHAR

```

```
parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR
```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWw0SE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [<3>](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line  
Response-headers  
CR/LF  
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section 2.2.3. The client SHOULD **<5>** use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section 2.2.1 and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section 2.2.1.1.2.2) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [RFC2616]), the response is interpreted as specified in [MS-ASCMD] section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [MS-ASCMD] section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section 2.2.2.1.2.11) in the response, the client SHOULD send an **Autodiscover** command request ([MS-ASCMD] section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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

Wireless Application Protocol (WAP) Binary XML (WBXML): A compact binary representation of **XML** that is designed to reduce the transmission size of XML documents over narrowband communication channels.

XML: The Extensible Markup Language, as described in [\[XML1.0\]](#).

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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

1.2 References

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

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[Exchange ActiveSync: Command Reference Protocol](#)".

[MS-ASPROV] Microsoft Corporation, "[Exchange ActiveSync: Provisioning Protocol](#)".

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, <http://www.ietf.org/rfc/rfc1945.txt>

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, <http://www.rfc-editor.org/rfc/rfc2045.txt>

[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.rfc-editor.org/rfc/rfc2616.txt>

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC4985] Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, August 2007, <http://www.rfc-editor.org/rfc/rfc4985.txt>

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <http://www.rfc-editor.org/rfc/rfc5234.txt>

[RFC5246] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008, <http://www.ietf.org/rfc/rfc5246.txt>

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011, <https://tools.ietf.org/html/rfc6265>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", W3C Note, June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

1.2.2 Informative References

[MSDN-APM] Marquardt, T., "ASP.NET Performance Monitoring, and When to Alert Administrators", <http://msdn.microsoft.com/en-us/library/ms972959.aspx>

1.3 Overview

This protocol is used to synchronize server data with a client mobile device. The protocol relies on a client/server architecture. In this specification, the term "client" is used to refer to the software that is running on the device and communicating to the server by means of the ActiveSync protocol. The term "server" refers to the synchronization engine that communicates the synchronization protocol to the client.

All communication between the client and server is initiated by the client and is based on request/response messages. When the client communicates with the server, the client sends a request to the server as an **HTTP POST** method, using UTF-8 encoding. The server sends back a response to the **HTTP POST**. The request and response each have a start-line, headers, and might have a body. The format is dictated by the HTTP/1.1 standard. The **HTTP POST** request header contains certain parameters that are set by the client, as specified later in this document. The **HTTP POST** response header is created by the server, and its contents are specified later in this document. The format of the body for both request and response depends on the type of request. Generally, the request/response body contains **Wireless Application Protocol (WAP) Binary XML (WBXML)** formatted data. Each **HTTP POST** request contains a single command, such as the **Sync** command. A typical session includes several commands and, therefore, several **HTTP POST** requests.

In addition to the **HTTP POST** request/response commands, the **HTTP OPTIONS** command response provides the supported ActiveSync capabilities of the server, including supported commands and supported protocol versions.

1.4 Relationship to Other Protocols

This protocol uses either an **HTTP** connection or an **HTTPS** connection between the client and server. A TCP/IP network transports messages between a client and server by using either the HTTP protocol or the HTTPS protocol, by means of a series of request and response calls. The protocol specified in [\[MS-ASCMD\]](#) uses this protocol as a transport.

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

1.5 Prerequisites/Preconditions

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol specifies the transport mechanism for the commands defined in [\[MS-ASCMD\]](#) and all data structures associated with those commands. It is applicable to any client or server that synchronizes **calendar**, **contact (2)**, e-mail, task, note, and other data between a mail server and a mobile device.

1.7 Versioning and Capability Negotiation

The **HTTP OPTIONS** command (section [2.2.3](#)) is used by the client to discover which versions of the ActiveSync protocol are supported by the server. To determine the supported versions, the client examines the MS-ASProtocolVersions header (section [2.2.4.1.2.2](#)), which is returned in the HTTP **OPTIONS** command response.

The client uses the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) of the HTTP **POST** command (section [2.2.1](#)) to indicate to the server which ActiveSync protocol version it is using.

The latest version of the ActiveSync protocol that the client or server can support is 16.1. Older versions include 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

1.8 Vendor-Extensible Fields

None

1.9 Standards Assignments

None

2 Messages

2.1 Transport

Messages are transported by using **HTTP POST** and **HTTP OPTIONS**, as specified in [\[RFC2616\]](#). These commands are sent via HTTP or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**. The query parameters in the request **URI** can be encoded with **base64 encoding** (see section [2.2.1.1.1.1](#)) or in plain text (see section [2.2.1.1.1.2](#)). The body of the HTTP message contains the **WBXML** that is required by the command being communicated in the message. The commands are specified in [\[MS-ASCMD\]](#).

2.2 Message Syntax

The XML markup that constitutes the request body (section [2.2.1.1.3](#)) or the response body (section [2.2.2.1.3](#)) is transmitted between client and server by using **Wireless Application Protocol (WAP) Binary XML (WBXML)**, as specified in [\[WBXML1.2\]](#).

The following are the two general types of messages:

- **HTTP POST**
- **HTTP OPTIONS**

2.2.1 HTTP POST Request

The client creates a request by using the **HTTP POST** command to initiate communications between the client and the server.

2.2.1.1 Request Format

Each command is sent from the client to the server as an **HTTP POST** containing command data. As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers  
CR/LF  
Request Body
```

2.2.1.1.1 Request Line

The request line consists of the method indicator, **POST**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
POST <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path, followed by a query value. The relative URI consists of the path and the query value.

The path and query value in the URI have the following format.

```
/<ActiveSync virtual directory name>?<query value>
```

The query value in the URI contains all of the URI parameters and can contain some of the request headers. The format can be either **plain text**, as specified in section 2.2.1.1.1.2, or **base64 encoding**, as specified in section 2.2.1.1.1.1. Either format can be used with protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. The base64 encoding format is not supported by protocol versions 2.5 and 12.0; the plain text format is supported by all protocol versions.

The following two examples are equivalent. The first example uses the plain text query value, and the second example uses the base64-encoded query value.

```
POST /Microsoft-Server-ActiveSync?Cmd=Sync&User=rmjones&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com
Accept-Language: en-us
Content-Length: 868

POST /Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
User-Agent: ASOM
Host: Contoso.com
Content-Length: 866
```

2.2.1.1.1.1 Base64-Encoded Query Value

The base64-encoded query value uses **base64 encoding** to specify the **URI** parameters and request headers. The URI parameters and request headers are contained within the fields of a byte sequence. Once the byte sequence is created, it is converted to base64 as specified in [RFC2045]. The base64-encoded query value is then appended to the request URI.

The following is an example of a URI that contains a base64-encoded query value.

```
/Microsoft-Server-ActiveSync?jAAJBAP2MTQwRGV2aWNlAApTbWFydFBob25l
```

NOTE: The base64-encoded query value is supported only by protocol versions 12.1, 14.0, 14.1, 16.0, and 16.1. If the client uses protocol version 2.5 or 12.0, the plain text query value, as specified in section 2.2.1.1.1.2, **MUST** be used in the request URI.

The fields of the byte sequence are as follows.

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									
Protocol version								Command code								Locale																								
Device ID length								Device ID (variable)																																
...																																								
Policy key length								Policy key (optional)																																
...								Device type length								Device type (variable)																								
...																																								

Command parameters (variable)
...

Protocol version (1 byte): An integer that specifies the version of the ActiveSync protocol that is being used. This value SHOULD [<1>](#) be 141, 160 or 161. This value MAY [<2>](#) be 140 or 121.

Command code (1 byte): An integer that specifies the command (see table of command codes in section [2.2.1.1.1.2](#)).

Locale (2 bytes): An integer that specifies the **locale** of the language that is used for the response. Locale integer values are specified in [\[MS-LCID\]](#).

Device ID length (1 byte): An integer that specifies the length of the **Device ID** field. This value MUST be greater than 0.

Device ID (variable): A string or a **GUID** that identifies the device. For details, see section [2.2.1.1.1.2.3](#). The length of this field is specified by the **Device ID length** field.

Policy key length (1 byte): An integer that specifies the length of the policy key. The only valid values are 0 or 4. A value of 0 indicates that the policy key field is absent.

Policy key (4 bytes, optional): An unsigned integer that indicates the state of policy settings on the client device, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. If the value of the **Policy key length** field is 0, this field is absent.

Device type length (1 byte): An integer that specifies the length of the **Device type** field.

Device type (variable): A string that specifies the type of client device. For details, see section [2.2.1.1.1.2.4](#). The length of this field is specified by the **Device type length** field.

Command parameters (variable): An array of **Encoded Parameter** structures as specified in section [2.2.1.1.1.1](#). This field is only present if there are command-specific parameters associated with the command specified by the **Command code** field. See section [2.2.1.1.1.3](#) for a list of command-specific parameters.

2.2.1.1.1.1.1 Encoded Parameter

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						
Tag										Length										Value (variable)																	
...																																					

Tag (1 byte): An integer that identifies the parameter. See section [2.2.1.1.1.3](#) for a list of tags and their corresponding parameters.

Length (1 byte): An integer that specifies the length of the parameter value. Valid values are from 0 to 255 characters.

Value (variable): The value of the parameter. The size of this field is specified by the **Length** field.

2.2.1.1.1.1.2 Command Codes

The following table provides the numeric codes that correspond to the ActiveSync commands. The numeric code is used in the **Command code** field of the base64 encoded **URI** to specify the command. For more details, see [\[MS-ASCMD\]](#).

Code	Command	Description
0	Sync	Synchronizes changes in a folder between the client and the server.
1	SendMail	Sends mail to the server. This command is issued in the HTTP POST command's URI, and does not contain an XML body. The body will instead contain the MIME message .
2	SmartForward	Forwards a Message object without retrieving the full Message object from the server.
3	SmartReply	Replies to a Message object without retrieving the full Message object from the server.
4	GetAttachment	Retrieves an e-mail attachment from the server.
9	FolderSync	Synchronizes the folder hierarchy but does not synchronize the items in the folders.
10	FolderCreate	Creates an e-mail, calendar , or contacts folder on the server.
11	FolderDelete	Deletes a folder from the server.
12	FolderUpdate	Moves a folder from one location to another on the server and is used to rename folders.
13	MoveItems	Moves items from one folder to another.
14	GetItemEstimate	Gets an estimate of the number of items in a folder that is synchronized.
15	MeetingResponse	Used to accept, tentatively accept, or decline a meeting request in the user's Inbox folder .
16	Search	Finds and retrieves information about contacts (2) and recipients in the Global Address List .
17	Settings	Supports getting and setting global properties, such as Out of Office (OOO) and device information.
18	Ping	Requests that the server monitor specified folders for changes that would require the client to resynchronize.
19	ItemOperations	Identifies the body of the request or response as containing a set of commands operating on items.
20	Provision	Gets the security policy settings set by the server administrator, such as the user's minimum password length requirement.
21	ResolveRecipients	Resolves a list of supplied recipients and optionally fetches their S/MIME certificates so that clients can send encrypted messages .
22	ValidateCert	Validates a certificate that has been received through an S/MIME mail.
23	Find	Searches for items in the mailbox using KQL syntax.

2.2.1.1.1.3 Command Parameters

The following table lists the tag values that correspond to the names of the command parameters. For additional details about the **AttachmentName**, **CollectionId**, **ItemId**, **LongId**, and **Occurrence** command parameters, see section [2.2.1.1.1.2.5](#).

Tag	Parameter Name
0	AttachmentName
1	CollectionId
3	ItemId
4	LongId
6	Occurrence
7	Options
8	User

The following table describes the **Options** and **User** command parameters.

Parameter	Description	Used By
Options	A single-byte bitmask that specifies command options. See the table below for valid flags for this bitmask.	SmartReply , SmartForward , SendMail , ItemOperations
User	A string that specifies the user ID in a format that can be logged in the Web server log.	Any command

The following table specifies the valid bit flags for the **Options** parameter.

Flag	Value	Meaning
SaveInSent	0x01	Set this flag to instruct the server to save the Message object in the user's Sent Items folder . Valid for SendMail , SmartForward , and SmartReply .
AcceptMultiPart	0x02	Set this flag to instruct the server to return the requested item in multipart format. Valid for ItemOperations . For more details, see section 2.2.1.1.2.5 .

2.2.1.1.1.2 Plain Text Query Value

The plain text query value uses **plain text** to specify the **URI** parameters. The **Augmented Backus-Naur Form (ABNF)** notation, as specified in [\[RFC5234\]](#), is used to define the syntax.

```

plain-text-query      = command-spec '&' user-spec '&' device-id-spec '&' device-type-spec
                        *('&' parameter-spec)

command-spec          = "Cmd=" command-name
user-spec              = "User=" user-name
device-id-spec        = "DeviceId=" device-id
device-type-spec      = "DeviceType=" device-type
parameter-spec        = parameter-name "=" parameter-value

command-name           = 1*ALPHA
user-name              = 1*VCHAR
device-id              = 1*32 (ALPHA / DIGIT)
device-type            = 1*VCHAR

```

```
parameter-name      = 1*ALPHA
parameter-value     = 1*VCHAR
```

2.2.1.1.1.2.1 Command

The ActiveSync command to be executed is specified by the `command-spec` **ABNF** rule portion of the plain text query value. Valid values, represented by the `command-name` ABNF rule, are specified in the "Command" column of the table in section [2.2.1.1.1.1.2](#).

2.2.1.1.1.2.2 User Name

The user ID of the user is specified by the `user-spec` **ABNF** rule portion of the plain text query value.

2.2.1.1.1.2.3 Device ID

The device ID is specified by the `device-id-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-id` ABNF rule, is a string that specifies the device. Each device MUST have a unique device ID string. Each request from the device MUST include the same device ID string.

2.2.1.1.1.2.4 Device type

The device type is specified by the `device-type-spec` **ABNF** rule portion of the plain text query value. The value, represented by the `device-type` ABNF rule, is any string that specifies a device type. "SP" specifies a SmartPhone and "PPC" specifies a PocketPC. Other client devices send unique strings for their specific device type. Each request from a client device MUST include the same device type string.

2.2.1.1.1.2.5 Command-Specific URI Parameters

The following **URI** parameters, also called command parameters, are specific to the ActiveSync commands. They are specified by the `parameter-spec` **ABNF** rule portion of the plain text query value. Valid values for the parameter name, represented by the `parameter-name` ABNF rule, are specified by the "Parameter" column in the following table. Valid parameter values, represented by the `parameter-value` ABNF rule, are specified in the "Description" column.

Parameter	Description	Used by
AttachmentName	A string that specifies the name of the attachment file to be retrieved.	GetAttachment
CollectionId	A string that specifies the server ID of the folder that contains the Message object to be forwarded or replied to.	SmartForward, SmartReply
ItemId	A string that specifies the server ID of the Message object to be forwarded or replied to.	SmartForward, SmartReply
LongId	A string that references a result set that was returned in the Search command response.	SmartForward, SmartReply
Occurrence	A string that specifies the ID of a particular occurrence in a recurring meeting .	SmartForward, SmartReply
SaveInSent	A character that specifies whether a copy of the Message object will be saved in the Sent Items folder . Set this parameter to T to instruct the server to save the Message object in the user's Sent Items folder; otherwise, set the parameter to F. The SaveInSent parameter is set to F by default.	SmartForward, SmartReply, SendMail

For more details about specific commands, see [\[MS-ASCMD\]](#).

2.2.1.1.2 Request Headers

The HTTP/1.1 protocol ([\[RFC2616\]](#)) defines several headers that can be sent from the client to the server on an **HTTP POST** request. The headers follow the request line in the HTTP portion of a request. The following headers are used in ActiveSync synchronization protocol requests. Note that requests are UTF-8 encoded.

Header	Required	Notes
Accept-Language	No.	For details, see section 2.2.1.1.2.1 .
Authorization	Yes, if using basic or OAuth authentication.	For details, see section 2.2.1.1.2.2 .
Content-Type	Depends on the command.	Specifies that the media type of the request body is WBXML. Other types of content, such as [RFC2822] , can also be specified, depending on the command. For more details, see section 2.2.1.1.2.3 .
Cookie	Depends on the contents of previous server responses and the protocol version in use.	Contains one or more cookies that the client previously received from the server in a Set-Cookie header. For more details, see section 2.2.1.1.2.4 .
MS-ASAcceptMultiPart	No	Specifies that the client wants items returned in multipart format. For more details, see section 2.2.1.1.2.5 .
MS-ASProtocolVersion	No if using a base64 encoded query value; yes if using a plain text query value.	Specifies the version of the ActiveSync protocol that the client supports. For more details, see section 2.2.1.1.2.6 .
User-Agent	No	Contains information about the client sending the request. For more details, see section 2.2.1.1.2.7 .
X-MS-PolicyKey	Depends on the command.	Specifies the policy key assigned by the server to the client. For more details, see section 2.2.1.1.2.8 .

2.2.1.1.2.1 Accept-Language

The Accept-Language header is used to define the **locale** of the client which is used when performing searches with the **Find** or **Search** command requests. If the accept language is not specified, the search is conducted by using the server language.

2.2.1.1.2.2 Authorization

Users authenticate through the ActiveSync protocol by using **HTTP** basic authentication, **OAuth** or a client certificate. Credentials are passed in different formats depending upon the form of authentication.

For HTTP basic authentication, credentials are encoded with **base64 encoding**. For user *fakename* and password *x\$pIAK9@p9!*, the following is the authorization header:

Authorization: Basic ZmFrZXVzZXI6eCRwSUFLbWwOSE=

For details about HTTP basic authentication, see [\[RFC1945\]](#) section 11.1.

For OAuth, an access token is obtained from the authorization server in response to an authorization grant. The access token is then used to obtain a protected resource from the resource server. The following is an example of an authorization header:

Authorization: Bearer <<token>>

For details about the OAuth 2.0 framework, see [\[RFC6749\]](#).

For authentication using a client certificate, the client **MUST NOT** send an authorization header. The server prompts the client for a certificate as part of the initial **SSL/TLS handshake** or as part of a **TLS** renegotiation.

If no client certificate exists, the client **MUST** complete the SSL/TLS handshake.

For details about providing a client certificate during a SSL/TLS handshake, see [\[RFC5246\]](#) section 7.4.6.

2.2.1.1.2.3 Content-Type

The Content-Type header indicates the format of the data sent in the request body. When the request body for a command is in **WBXML** format, the Content-Type header value **MUST** be set to either "application/vnd.ms-sync.wbxml", or the shortened string "application/vnd.ms-sync". The shortened string is not allowed by protocol versions 2.5 and 12.0.

For the **Autodiscover** command ([\[MS-ASCMD\]](#) section 2.2.1.1), which specifies an **XML** request body format, the Content-Type header **SHOULD** be set to "text/xml" or **MAY** [\[3\]](#) be set to "text/html". If the request has no body, the Content-Type header **SHOULD NOT** be present.

2.2.1.1.2.4 Cookie

The Cookie header contains one or more cookies that the client previously received from the server in a Set-Cookie header (section [2.2.2.1.2.12](#)). Each cookie consists of the name and value. Multiple cookies are separated by a semi-colon. For details about the syntax, see [\[RFC6265\]](#).

Clients using protocol version 16.0 or 16.1 or clients using **OAuth** authentication for any protocol version **MUST** be able to parse and interpret a Set-Cookie header that is received from the server. If the server response includes the Set-Cookie header, clients using protocol version 16.0 or 16.1 or clients using OAuth authentication for any protocol version **MUST** provide these cookies in a Cookie header when sending future requests to the server.

2.2.1.1.2.5 MS-ASAcceptMultiPart

The MS-ASAcceptMultiPart header is used to control the delivery of the content requested by the **Fetch** element ([\[MS-ASCMD\]](#) section 2.2.3.67.1) in an **ItemOperations** request ([\[MS-ASCMD\]](#) section 2.2.1.10). This header is optional for **ItemOperations** requests, and **SHOULD NOT** be used for other command requests. This header **SHOULD NOT** be used if the base64-encoded query value is being used. Instead, the **AcceptMultiPart** flag in the **Options** parameter **SHOULD** be used, as specified in section [2.2.1.1.1.3](#).

If this header is present and the value is 'T', the client is requesting that the server return content in multipart format. If the header is not present, or is present and set to 'F', the client is requesting that the server return content in inline format. For more details, see [\[MS-ASCMD\]](#) section 2.2.1.10.1.

This header is not supported by protocol version 2.5.

2.2.1.1.2.6 MS-ASProtocolVersion

The MS-ASProtocolVersion header indicates the protocol version that the client is using to format the request. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Protocol version** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

The following values, which correspond to the ActiveSync protocol versions, are valid: "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.1.1.2.7 User-Agent

The format of the User-Agent header is specified in [\[RFC2616\]](#) section 14.43. This header SHOULD be included in command requests.

2.2.1.1.2.8 X-MS-PolicyKey

The X-MS-PolicyKey header contains the client's current policy key, as specified in [\[MS-ASPROV\]](#) section 2.2.2.42. This header SHOULD NOT be used if the base64-encoded query value is being used. Instead, the **Policy key** field of the base64-encoded query value SHOULD be set, as specified in section [2.2.1.1.1.1](#).

2.2.1.1.3 Request Body

The request body contains data sent to the server. The request body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Three commands have no body in certain contexts: **GetAttachment**, **Sync**, and **Ping**. For more details about the request bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.2 HTTP POST Response

After receiving and interpreting a request, a server responds with an **HTTP** response that contains data returned from the server.

2.2.2.1 Response Format

Each command response is sent from the server to the client as an **HTTP POST** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests.

```
Status-line  
Response-headers  
CR/LF  
Message Body
```

2.2.2.1.1 Status Line

The status line consists of the **HTTP** version and a status code. The following is an example of a response status line.

```
HTTP/1.1 200 OK
```

The following table lists some common HTTP status codes.

Status code	Description
200 OK	The command succeeded.
400 Bad Request	The request could not be understood by the server due to malformed syntax. If the client repeats the request without modifications, then the same error occurs.
401 Unauthorized	The resource requires authorization or authorization was refused. For details about the client's handling of this error, see section 3.1.5.2.1 .
403 Forbidden	The user is not enabled for ActiveSync synchronization. For details about the client's handling of this error, see section 3.1.5.2.1 .
404 Not Found	The specified URI could not be found or the server is not a valid server with ActiveSync.
451 Redirect	The device is trying to connect to a server that cannot access the user's mailbox , or there is a more efficient server to use to reach the user's mailbox. For details about the client's handling of this error, see section 3.1.5.2.2 .
456 Blocked	The user's account is blocked. For details about the client's handling of this error, see section 3.1.5.2.4 .
457 Expired Password	The user's password has expired. For details about the client's handling of this error, see section 3.1.5.2.4 .
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request. For details about the client's handling of this error, see section 3.1.5.2.1 .
501 Not Implemented	The server does not support the functionality that is required to fulfill the request. This status code SHOULD be returned by the server when the server does not recognize the request method or is not able to support it for any resource. In the case of other malformed requests, the server returns status code 400.
502 Proxy Error	The specified server could not be found.
503 Service Unavailable	The service is unavailable. For details about the client's handling of this error, see section 3.1.5.2.3 .
507 Insufficient Disk Space	The user's mailbox is full.

2.2.2.1.2 Response Headers

This protocol and [\[RFC2616\]](#) define several headers that can be sent from the server to the client in an **HTTP POST** response. The headers follow the status line in the HTTP part of a response. The following table lists some common headers that can be set by the server in response to client requests.

Header	Example value	Notes
Cache-Control	private	Optional. Controls how the response is cached.
Content-Encoding	gzip	Required when the content is compressed; otherwise, this header is not included. Specifies the HTTP

Header	Example value	Notes
		compression format that is used in the response.
Content-Length	56	Optional. Specifies the size of the response body in bytes.
Content-Type	application/vnd.ms-sync.wbxml	Required. Specifies that the media-type of the response body is WBXML . Other types of content, such as RFC2822 , can also be specified.
MS-Server-ActiveSync	15.1	Optional. Indicates the version of the ActiveSync server that was used to handle the request.
X-MS-Location	https://mail.contoso.com/Microsoft-Server-ActiveSync	Optional. Used in conjunction with a 451 Redirect status code. Specifies the URL to use for future requests.
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert, Find	Optional. Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates the protocol versions supported by the server.
X-MS-RP	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Optional. Indicates to the client that the client has to perform a full resynchronization.
X-MS-Credential-Service-Url	https://portal.microsoftonline.com/ChangePassword.aspx	Optional. Contains a URL for reset of user's password.
X-MS-Credentials-Expire	13	Optional. Indicates the number of days remaining until expiration of a user's password.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

Header	Example value	Notes
X-BEServer	EXCH-SERV-1	Optional. Contains the name of the server that processed the request.
X-FEServer	EXCH-SERV-1	Optional. Contains the name of the server(s) that routed the request.
request-id	7faa449e-4912-4a79-aade-afee642c2c36	Optional. Contains a server-generated identifier for the request.

When protocol version 12.1, 14.0, 14.1, 16.0, or 16.1 is used: Some of the headers in the response can be eliminated when the response is to an HTTP **POST** request and the response has HTTP status 200. When these two conditions are met, only the following headers are necessary in the response:

- Content-Length
- Content-Type, required only if Content-Length is greater than zero.

2.2.2.1.2.1 Cache-Control

This header is optional. The value of this header controls how the response is cached, as specified in [\[RFC2616\]](#) section 14.9.

2.2.2.1.2.2 Content-Encoding

This header is required if the response body is compressed. Otherwise, it is omitted. See [\[RFC2616\]](#) section 14.11.

2.2.2.1.2.3 Content-Length

This header is optional. The format is specified in [\[RFC2616\]](#) section 14.13.

2.2.2.1.2.4 Content-Type

This header is required. If the response body is **WBXML**, the value of this header MUST be "application/vnd.ms-sync.wbxml". Otherwise, an appropriate value SHOULD be used as specified in [\[RFC2616\]](#) section 14.17.

2.2.2.1.2.5 MS-Server-ActiveSync

This header is optional. It contains an implementation-specific string indicating the version of the server.

2.2.2.1.2.6 X-MS-Location

This header is optional. It is used when the **HTTP** status code is 451 to provide a **URL** to use for subsequent requests.

2.2.2.1.2.7 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server. It will be returned in an HTTP POST response if the server requires the client to reinitialize its synchronization state.

2.2.2.1.2.8 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports. It will be returned in HTTP POST response headers if the server requires the client to reinitialize its synchronization state.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.2.1.2.9 X-MS-RP

This header is optional. Its presence in a response indicates that a condition on the server (such as a server upgrade) requires the client to discard its local data and resynchronize. The value of this header indicates the protocol versions the server supports.

This header is not supported by protocol version 2.5.

2.2.2.1.2.10 X-MS-Credential-Service-Url

This header is optional. This header contains the **URL** for a self-service web site that allows a user to reset the user password.

This header is required in the response to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) when the user's password is either near expiration or expired. The point at which a password is near expiration is determined by the implementer.

2.2.2.1.2.11 X-MS-Credentials-Expire

This header is optional. This header contains an integer that indicates the number of days remaining until the user's password expires. A value of 0 (zero) indicates that the password will expire in less than 24 hours.

This header can be included in a response to provide advance warning of password expiration.

2.2.2.1.2.12 Set-Cookie

This header is optional. The Set-Cookie header contains one cookie returned by the server. Each cookie consists of a name, a value, and the following attributes: *Expires*, *Path*, *Secure*, and *HttpOnly*. Multiple instances of this header can be returned with a different cookie name in each instance of the header.

For details about the syntax of this header, see [\[RFC6265\]](#).

2.2.2.1.2.13 X-MS-ASThrottle

This header is optional. The X-MS-ASThrottle header specifies the condition under which the server MAY [4](#) throttle the client device.

2.2.2.1.2.14 X-BEServer

This header is optional. The X-BEServer header contains the name of the server that processed the request.

2.2.2.1.2.15 X-FEServer

This header is optional. The X-FEServer header contains the name of the server(s) that routed the request.

2.2.2.1.2.16 request-id

This header is optional. The request-id header contains a server-generated identifier for the request.

2.2.2.1.3 Response Body

The response body contains data returned from the server. The response body, if any, is in **WBXML**, except the **Autodiscover** command, which is in **XML**. Two commands have no XML body in certain contexts: **GetAttachment** and **Sync**. For more details about the response bodies of individual commands, see [\[MS-ASCMD\]](#) section 2.2.1.

2.2.3 HTTP OPTIONS Request

The **HTTP OPTIONS** command, which is specified by [\[RFC2616\]](#), is used to discover what protocol versions are supported, and which protocol commands are supported on the server. The client uses the **HTTP OPTIONS** command to determine whether the server supports the same versions of the protocol that the client supports.

2.2.3.1 Request Format

As specified by [\[RFC2616\]](#), the format is as follows.

```
Request-line  
Request-headers
```

2.2.3.1.1 Request Line

The request line consists of the method indicator, **OPTIONS**, followed by the **URI**, followed by the **HTTP** version, as follows.

```
OPTIONS <URI> HTTP/1.1
```

The URI can be either an absolute URI or a relative URI, as specified in [\[RFC2616\]](#) section 3.2.1. The absolute URI consists of a scheme indicator, the host name, and the path. The relative URI consists of the path.

The path in the URI has the following format.

```
/<ActiveSync virtual directory name>
```

2.2.3.1.2 Request Headers

The authorization header is required. For more information on the authorization header requirements, see section [2.2.1.1.2.2](#).

2.2.4 HTTP OPTIONS Response

After receiving an **HTTP OPTIONS** request, a server responds with an **HTTP OPTIONS** response that specifies the protocol versions it supports.

2.2.4.1 Response Format

Each response is sent from the server to the client as an **HTTP OPTIONS** response. Note that these responses are UTF-8 encoded. As specified by [\[RFC2616\]](#), the format is the same as for the following requests:

```
Status-line
Response-headers
```

2.2.4.1.1 Status Line

The status line for an **HTTP OPTIONS** response is identical to the status line for an HTTP **POST** response, specified in section [2.2.2.1.1](#).

2.2.4.1.2 Response Headers

This protocol defines headers that can be sent from the server to the client in an **HTTP OPTIONS** response in addition to headers defined in [\[RFC2616\]](#). The headers follow the status line in the HTTP part of a response.

Header	Example value	Notes
MS-ASProtocolCommands	Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert,Find	Indicates the commands supported by the server.
MS-ASProtocolVersions	2.5,12.0,12.1,14.0,14.1,16.0,16.1	Indicates the protocol versions supported by the server.
Set-Cookie	X-Cookie=value; expires=Wed, 08-Jul-2015 23:40:27 GMT; path=/Microsoft-Server-ActiveSync; secure; HttpOnly	Optional. Contains one or more cookies returned by the server.

2.2.4.1.2.1 MS-ASProtocolCommands

The MS-ASProtocolCommands header contains a comma-delimited list of the ActiveSync commands supported by the server.

2.2.4.1.2.2 MS-ASProtocolVersions

The MS-ASProtocolVersions header contains a comma-delimited list of the ActiveSync protocol versions that the server supports.

The following values correspond to the ActiveSync protocol versions that are specified by [\[MS-ASCMD\]](#): "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", and "2.5". The latest version is 16.1.

2.2.4.1.2.3 Set-Cookie

For details about this header see section [2.2.2.1.2.12](#).

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

The client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server to determine the correct server **URL** to use for all subsequent commands.

After determining the correct server URL, the client SHOULD send an **HTTP OPTIONS** command to the server, as specified in section [2.2.3](#). The client SHOULD [≤5>](#) use the most recent version (the greatest numbered version) of the protocol that is supported by the client and server.

3.1.4 Higher-Layer Triggered Events

Synchronizing changes on the client requires the client to send a command to the server.

3.1.4.1 Sending a Command Request

Command requests MUST be formatted as specified in section [2.2.1](#) and sent via **HTTP. Secure Sockets Layer (SSL)** SHOULD be enabled between the client and the server whenever the Authorization header (section [2.2.1.1.2.2](#)) is sent. The client SHOULD wait for a server response to the request.

Clients that include the User-Agent HTTP header SHOULD NOT change the value of this header between consecutive command requests, unless a major change to the client has occurred, such as an operating system upgrade.

3.1.5 Message Processing Events and Sequencing Rules

Clients receive HTTP responses from the server only in response to HTTP requests sent by the client.

3.1.5.1 Handling a Successful Response

If the HTTP status code indicates that the request succeeded (its value is between 200 and 299, as specified in [\[RFC2616\]](#)), the response is interpreted as specified in [\[MS-ASCMD\]](#) section 2.2.1.

If the server returns an X-MS-RP header in the response, the client MUST reinitialize its synchronization state as specified in [\[MS-ASCMD\]](#) section 2.2.1.21. If the X-MS-RP header is received in response to a **FolderSync** request that has a synchronization key of 0, the client can ignore the X-MS-RP header.

If the server returns the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in the response, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to retrieve the **URL** for the self-service web site that allows a user to do a password reset. The **Autodiscover**

command response will include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)), which contains the URL for the self-service web site. The client SHOULD provide this URL to the user.

3.1.5.2 Handling a Failed Response

Any **HTTP** status code that is not between 200 and 299, as specified in [\[RFC2616\]](#), indicates that the request failed. The following sections specify the client's handling of certain HTTP status codes that are returned by the server when a request fails. All other HTTP status codes that indicate a failed request are interpreted and handled as specified in [\[RFC2616\]](#).

3.1.5.2.1 HTTP Error 401, 403, and 500

If the server responds to any command with an **HTTP** error 401, 403, or 500, the client SHOULD send an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) to the server.

3.1.5.2.2 HTTP Error 451

If the client is attempting to connect to the wrong server (that is, a server that cannot access the user's mailbox), or if there is a more efficient server to use to reach the user's mailbox, then a 451 Redirect error is returned.

The error returned by the wrong server resembles the following:

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 451
Date: Tue, 08 Dec 2009 19:43:24 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Cache-Control: private
Content-Length: 0
```

If an X-MS-Location header is present in the response, all subsequent requests SHOULD use the URL specified within the X-MS-Location header. If the server does not provide an X-MS-Location header in its response to the client, then the full **Autodiscover** command process is followed, as specified in [\[MS-ASCMD\]](#).

3.1.5.2.3 HTTP Error 503

The server returns an HTTP error 503 when more users than are allowed by the server's request queue limit have sent requests to a single server or when the actions of the client have triggered throttling.

The error returned by the server resembles the following.

```
OPTIONS /Microsoft-Server-ActiveSync
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0

HTTP/1.1 503 Service Unavailable
Connection: close
Date: Mon, 02 Mar 2009 23:51:51 GMT
Server: Microsoft-IIS/7.0
X-Powered-By: ASP.NET
Content-Type: text/html
```

If a Retry-After header ([\[RFC2616\]](#)) is present in the response, the client SHOULD [<6>](#) retry the request after waiting the number of seconds indicated by the Retry-After header. Any such value represents an estimate of when the server is expected to be able to process the request.

If a Retry-Header is not present in the response, the client can retry the request after waiting a few seconds. The time to wait between continuous requests that result in HTTP error 503 responses can be increased exponentially to a predetermined maximum.

For more details about ASP.NET performance monitoring properties, see [\[MSDN-APM\]](#).

3.1.5.2.4 HTTP Error 456 and 457

If the server responds to an **Autodiscover** command request ([\[MS-ASCMD\]](#) section 2.2.1.1) with an **HTTP** error 456, the client SHOULD stop sending requests to the server and then prompt the user to contact the administrator.

If the server responds to an **Autodiscover** command request with an HTTP error 457, the client SHOULD stop sending requests to the server and then prompt the user to reset the user password. The client SHOULD direct the user to the self-service web site that allows a user to do a password reset. The **URL** for this web site is contained in the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) of the server's response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server only responds to client requests by returning HTTP responses as specified in section [2.2.2](#) and never initiates communication with the client.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server can receive **HTTP POST** requests (section [2.2.1](#)) or HTTP **OPTIONS** requests (section [2.2.3](#)) from the client.

3.2.5.1 Handling HTTP POST Command Requests

The server parses HTTP **POST** requests from clients as specified in section [2.2.1](#). The ActiveSync command contained within an HTTP **POST** request is parsed as specified in [\[MS-ASCMD\]](#) section 2.2.1. The server MUST conform to the protocol version that is specified by the MS-ASProtocolVersion header (section [2.2.1.1.2.6](#)) in the client request. The server formats an HTTP **POST** response, as specified in section [2.2.2](#), with an appropriate **HTTP** status code, as specified in section [2.2.2.1.1](#).

If the server returns an HTTP 451 error and knows the **URL** of the correct server, it SHOULD include an X-MS-Location header (section [2.2.2.1.2.6](#)) with the URL of the correct server. If the server returns an HTTP 503 error, it MAY [<7>](#) include a Retry-After header, as specified in [\[RFC2616\]](#), in the response with an estimate of the number of seconds that will elapse before the server is expected to be able to process the request. If the server returns an HTTP 457 error, it MUST include the X-MS-Credential-Service-Url header (section [2.2.2.1.2.10](#)) in its response.

If the user's password is near expiration, the server SHOULD include the X-MS-Credentials-Expire header (section [2.2.2.1.2.11](#)) in its response. This header provides advance warning to the client of password expiration. The point at which the server begins this warning is determined by the implementer.

If the client sends a request to synchronize the folder hierarchy with a synchronization key of 0 or the server requires the client to reinitialize its synchronization state, the server SHOULD include an X-MS-RP header, an MS-ASProtocolCommands header, and an MS-ASProtocolVersions header in its response to the client.

The server MAY [<8>](#) track the value of the User-Agent header for consecutive command requests from a specific device and block devices that change the value of this header more than a configured limit within a configured timespan. Servers that do this tracking SHOULD use the algorithm specified in section [3.2.5.1.1](#).

3.2.5.1.1 User-Agent Change Tracking

Servers SHOULD limit changes to the User-Agent header value from a device to two changes within a 24-hour time period, but MAY [<9>](#) use different values for the number of changes or the time period. The server SHOULD block clients that exceed this limit for 14 hours, but MAY [<10>](#) block clients for a different amount of time.

3.2.5.2 Handling HTTP OPTIONS Command Requests

The server parses HTTP **OPTIONS** requests from clients as specified in section [2.2.3](#) and formats its response as specified in section [2.2.4](#). The server's response MUST contain both the MS-ASProtocolCommands header, as specified in section [2.2.4.1.2.1](#), and the MS-ASProtocolVersions header, as specified in section [2.2.4.1.2.2](#). The server uses these headers to indicate which ActiveSync commands and which ActiveSync protocol versions it supports.

A protocol server can support multiple versions of the ActiveSync protocol. This specification, and any protocol specifications that cite it as a dependency, apply to the server configuration when the value of the MS-ASProtocolVersions header is set to a value that includes "16.1", "16.0", "14.1", "14.0", "12.1", "12.0", or "2.5". [<11>](#)

The latest version of the ActiveSync protocol is 16.1. Older versions are 16.0, 14.1, 14.0, 12.1, 12.0, and 2.5. Some commands and functionality described in the ActiveSync protocol documentation are not supported by all of the protocol versions. See the command and element descriptions in the ActiveSync protocol documents to determine which commands, elements, and capabilities are supported by the protocol versions.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 FolderSync Request and Response

The following is a typical ActiveSync protocol command request. The **FolderSync** command, user **alias**, device ID, and device type are specified as **URI** query parameters. The Content-Type header specifies that the request body is WBXML. The MS-ASProtocolVersion header specifies that protocol 14.0 is being used. Some command requests contain additional URI query parameters or do not specify a request body. The **HTTP POST URI** command parameter is the same as the command in the topmost element of the request XML body. For details about the commands and associated **XML schema definitions (XSDs)**, see [\[MS-ASCMD\]](#). The WBXML-encoded body is decoded for clarity.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>
```

The following is a typical FolderSync command response. The status line specifies the HTTP/1.1 protocol and that the command succeeded. The Content-Length header specifies that the response body is 56 bytes and the Content-Type header shows that the response body is in WBXML format. Some command responses do not contain WBXML bodies.

Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 19:34:31 GMT
Content-Length: 25

<?xml version="1.0" encoding="utf-8"?>
<FolderSync>
  <Status>1</Status>
  <SyncKey>2</SyncKey>
  <Changes>
    <Count>0</Count>
  </Changes>
</FolderSync>
```

4.2 FolderSync Request and Redirect Response

The following is the same request from the example described in section [4.1](#). In this example, configuration changes on the server have caused the "contoso.com" host to no longer be the optimal host for the user.

Request

```

POST /Microsoft-Server-ActiveSync?Cmd=FolderSync&User=fakename&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderSync xmlns="FolderHierarchy:">
  <SyncKey>2</SyncKey>
</FolderSync>

```

The server redirects the client to the "mail.contoso.com" host using an **HTTP** status code 451 and the **X-MS-Location** header.

```

HTTP/1.1 451
Date: Thu, 12 Mar 2009 20:16:22 GMT
X-MS-Location: https://mail.contoso.com/Microsoft-Server-ActiveSync
Content-Length: 0

```

4.3 HTTP OPTIONS Command Request and Response

The following example illustrates the use of the **HTTP OPTIONS** command. The MS-ASProtocolVersions header in the server response shows that versions 1.0, 2.0, 2.1, 2.5, 12.0, 12.1, and 14.0 of the protocol are supported on the server. The MS-ASProtocolCommands header in the server response lists the commands that are supported. It is recommended that protocol clients not trigger on the build number of the protocol server, which can change because of server updates. The build number shown in the examples might differ from those seen in a development or production environment.

Request

```

OPTIONS /Microsoft-Server-ActiveSync HTTP/1.1
Host: Contoso.com

```

Response

```

HTTP/1.1 200 OK
Cache-Control: private
Allow: OPTIONS,POST
Server: Microsoft-IIS/7.0
MS-Server-ActiveSync: 14.00.0536.000
MS-ASProtocolVersions: 2.0,2.1,2.5,12.0,12.1,14.0
MS-ASProtocolCommands: Sync,SendMail,SmartForward,SmartReply,GetAttachment,GetHierarchy,
CreateCollection,DeleteCollection,MoveCollection,FolderSync,FolderCreate,
FolderDelete,FolderUpdate,MoveItems,GetItemEstimate,MeetingResponse,Search,
Settings,Ping,ItemOperations,Provision,ResolveRecipients,ValidateCert
Public: OPTIONS,POST
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 12 Mar 2009 20:03:29 GMT
Content-Length: 0

```

4.4 SendMail Request and Response

The following example illustrates the command to send mail to a specific user.

Request

```
POST /Microsoft-Server-ActiveSync?Cmd=SendMail&User=fakeusername&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
X-MS-PolicyKey: 2034202722
User-Agent: ASOM
Host: BIRSKK-dom.extest.microsoft.com

<?xml version="1.0" encoding="utf-8"?>
<SendMail
  xmlns="ComposeMail:">
  <ClientId>633724606026842453</ClientId>
  <Mime>From: fakeuser@Contoso.com
  To: fakeuser@Contoso.com
  Cc:
  Bcc:
  Subject: From NSync
  MIME-Version: 1.0
  Content-Type: text/plain; charset="iso-8859-1"
  Content-Transfer-Encoding: 7bit
  X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3350
  This is the body text.</Mime>
</SendMail>
```

Response

```
HTTP/1.1 200 OK
S
Date: Thu, 12 Mar 2009 20:16:22 GMT
Content-Length: 0
```

4.5 CreateFolder Request and Response

The following example illustrates the command to create a new folder. For details about the associated XML schema definitions (XSD), see [\[MS-ASCMD\]](#).

Request:

```
POST /Microsoft-Server-ActiveSync?Cmd=FolderCreate&User=fakename@Contoso.com&DeviceId=v140Device&DeviceType=SmartPhone HTTP/1.1
Content-Type: application/vnd.ms-sync.wbxml
MS-ASProtocolVersion: 14.0
User-Agent: ASOM
Host: Contoso.com

<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <SyncKey>3</SyncKey>
  <ParentId>5</ParentId>
  <DisplayName>CreateNewFolder</DisplayName>
  <Type>12</Type>
</FolderCreate>
```

Response:

```
HTTP/1.1 200 OK
```

Content-Type: application/vnd.ms-sync.wbxml

Date: Thu, 12 Mar 2009 20:s26:06 GMT

Content-Length: 24

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate
  xmlns="FolderHierarchy:">
  <Status>1</Status>
  <SyncKey>4</SyncKey>
  <ServerId>23</ServerId>
</FolderCreate>
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this specification. It is recommended that communication between the client and server occur across an HTTP connection secured by the **Secure Sockets Layer (SSL)** protocol.

When connecting to a server using SSL, clients are required to support server certificates that use the Subject Alternative Name for domain names, as specified in [\[RFC4985\]](#), as well as wildcard certificate names, as specified in [\[RFC2818\]](#) and [\[RFC3280\]](#).

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 updates to those products.

- Microsoft Exchange Server 2007 Service Pack 1 (SP1)
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Windows 8.1
- Windows 10 operating system

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

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

[<1> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 and the initial release version of Exchange 2010 do not set the **Protocol version** field to 141, 160 or 161. Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Exchange 2013 do not set the **Protocol version** field to 160 or 161.

[<2> Section 2.2.1.1.1.1](#): Exchange 2007 SP1 sets the **Protocol version** field to 121. The initial release version of Exchange 2010 sets the **Protocol version** field to 140.

[<3> Section 2.2.1.1.2.3](#): Exchange 2007 SP1 accepts a Content-Type header value of either "text/xml" or "text/html" for the **Autodiscover** command.

[<4> Section 2.2.2.1.2.13](#): The X-MS-AShrottle header and throttling are not supported in Exchange 2007 SP1 and Exchange 2010. The X-MS-AShrottle header and throttling are supported in Exchange 2013 and Exchange 2016 but are disabled by default.

[<5> Section 3.1.3](#): Windows Communication Apps only support protocol versions 12.1 and 14.0.

[<6> Section 3.1.5.2.3](#): Windows Communication Apps ignore the Retry-After header and instead retry after a set time. The set time increases exponentially when multiple 503 errors are received.

[<7> Section 3.2.5.1](#): Exchange 2010 and Exchange 2013 sometimes include a Retry-After header with HTTP 503 error responses.

[<8> Section 3.2.5.1](#): Exchange 2013 and Exchange 2016 can be configured to track changes to the User-Agent header, but do not do so by default.

[<9> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to use different values for the allowed number of changes and the time period.

[<10> Section 3.2.5.1.1](#): Exchange 2013 and Exchange 2016 can be configured to block clients for an amount of time other than 14 hours.

[<11> Section 3.2.5.2](#): Exchange 2007 SP1 does not return the value "16.1", "16.0", "14.1", or "14.0" in the MS-ASProtocolVersions header. The initial release version of Exchange 2010 does not return the

value "16.1", "16.0" or "14.1" in the MS-ASProtocolVersions header. Exchange 2010 SP1 and Exchange 2013 do not return the value "16.1" or "16.0" in the MS-ASProtocolVersions header.

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

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

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

Section	Description	Revision class
2.2.1.1.2.2 Authorization	Updated description for the Authorization header.	Minor
2.2.2.1.2 Response Headers	Added optional debugging headers X-BEServer, X-FEServer, and request-id to the table.	Major
2.2.2.1.2.14 X-BEServer	Added section for this header.	Major
2.2.2.1.2.15 X-FEServer	Added section for this header.	Major
2.2.2.1.2.16 request-id	Added section for this header.	Major

8 Index

A

Abstract data model
 [client](#) 27
 [server](#) 29
[Applicability](#) 11

C

[Capability negotiation](#) 11
[Change tracking](#) 39
Client
 [abstract data model](#) 27
 [higher-layer triggered events](#) 27
 [initialization](#) 27
 [message processing](#) 27
 [other local events](#) 29
 [sequencing rules](#) 27
 [timer events](#) 29
 [timers](#) 27
[CreateFolder example](#) 34

D

Data model - abstract
 [client](#) 27
 [server](#) 29

E

Examples
 [CreateFolder](#) 34
 [FolderSync](#) 32
 [FolderSync redirect response](#) 32
 [HTTP OPTIONS](#) 33
 [SendMail](#) 33

F

[Fields - vendor-extensible](#) 11
[FolderSync example](#) 32
[FolderSync redirect response example](#) 32

G

[Glossary](#) 7

H

Higher-layer triggered events
 [client](#) 27
 [server](#) 29
[HTTP OPTIONS example](#) 33
[HTTP OPTIONS Request message](#) 25
[HTTP OPTIONS Response message](#) 25
[HTTP POST Request message](#) 12
[HTTP POST Response message](#) 20

I

[Implementer - security considerations](#) 36

[Index of security parameters](#) 36
[Informative references](#) 10
Initialization
 [client](#) 27
 [server](#) 29
[Introduction](#) 7

M

Message processing
 [client](#) 27
 [server](#) 29
Messages
 [HTTP OPTIONS Request](#) 25
 [HTTP OPTIONS Response](#) 25
 [HTTP POST Request](#) 12
 [HTTP POST Response](#) 20
 [transport](#) 12

N

[Normative references](#) 9

O

Other local events
 [client](#) 29
 [server](#) 31
[Overview \(synopsis\)](#) 10

P

[Parameters - security index](#) 36
[Preconditions](#) 10
[Prerequisites](#) 10
[Product behavior](#) 37

R

[References](#) 9
 [informative](#) 10
 [normative](#) 9
[Relationship to other protocols](#) 10

S

Security
 [implementer considerations](#) 36
 [parameter index](#) 36
 [SendMail example](#) 33
Sequencing rules
 [client](#) 27
 [server](#) 29
Server
 [abstract data model](#) 29
 [higher-layer triggered events](#) 29
 [initialization](#) 29
 [message processing](#) 29
 [other local events](#) 31
 [overview](#) 29
 [sequencing rules](#) 29

[timer events](#) 31
[timers](#) 29
[Standards assignments](#) 11

T

Timer events
 [client](#) 29
 [server](#) 31
Timers
 [client](#) 27
 [server](#) 29
[Tracking changes](#) 39
[Transport](#) 12
Triggered events - higher-layer
 [client](#) 27
 [server](#) 29

V

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