

[MS-WSSHP]:

HTTP Windows SharePoint Services Headers Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1	New	Initial Availability
6/27/2008	1.0	Major	Revised and edited the technical content
10/6/2008	1.01	Editorial	Revised and edited the technical content
12/12/2008	1.02	Editorial	Revised and edited the technical content
7/13/2009	1.03	Major	Revised and edited the technical content
8/28/2009	1.04	Editorial	Revised and edited the technical content
11/6/2009	1.05	Editorial	Revised and edited the technical content
2/19/2010	2.0	Editorial	Revised and edited the technical content
3/31/2010	2.01	Editorial	Revised and edited the technical content
4/30/2010	2.02	Editorial	Revised and edited the technical content
6/7/2010	2.03	Editorial	Revised and edited the technical content
6/29/2010	2.04	Minor	Clarified the meaning of the technical content.
7/23/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	2.04	None	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	2.04	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	3.0	Major	Significantly changed the technical content.
4/11/2012	3.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	3.0	None	No changes to the meaning, language, or formatting of the technical content.
9/12/2012	3.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	4.0	Major	Significantly changed the technical content.
2/11/2013	4.0	None	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	4.0	None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
11/18/2013	4.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	4.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	4.0	None	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	4.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	5.0	Major	Significantly changed the technical content.
3/16/2015	6.0	Major	Significantly changed the technical content.
6/30/2015	7.0	Major	Significantly changed the technical content.
2/26/2016	8.0	Major	Significantly changed the technical content.
4/14/2016	9.0	Major	Significantly changed the technical content.

Table of Contents

1	Introduction	6
1.1	Glossary	6
1.2	References	7
1.2.1	Normative References	8
1.2.2	Informative References	8
1.3	Protocol Overview (Synopsis)	8
1.4	Relationship to Other Protocols	10
1.5	Prerequisites/Preconditions	10
1.6	Applicability Statement	10
1.7	Versioning and Capability Negotiation	10
1.8	Vendor-Extensible Fields	10
1.9	Standards Assignments.....	10
2	Messages.....	11
2.1	Transport	11
2.2	Message Syntax	11
2.2.1	Expires Header.....	11
2.2.2	Content-Disposition Header	11
2.2.3	x-virus-infected Header	11
2.2.4	x-irm-cantdecrypt Header.....	12
2.2.5	x-irm-rejected Header.....	12
2.2.6	x-irm-notowner Header	12
2.2.7	x-irm-timeout Header	12
2.2.8	x-irm-crashed Header	13
2.2.9	x-irm-unknown-failure Header.....	13
2.2.10	SharePointError Header	13
2.2.11	X-RequestDigest Header.....	13
2.2.12	X-Forms_Based_Auth_Required Header.....	14
2.2.13	X-Forms_Based_Auth_Return_Url Header.....	14
2.2.14	X-MS-File-Checked-Out Header	14
2.2.15	X-RequestToken Header	14
2.2.16	SPRequestGuid Header	15
2.2.17	X-UseWebLanguage Header	15
2.2.18	X-RequestForceAuthentication Header.....	15
2.2.19	X-SharePointHealthScore.....	15
2.2.20	X-MS-InvokeApp Header	15
2.2.21	Distinguishing Clients with HTTP Headers	16
2.2.21.1	Browser Client	16
2.2.21.2	Crawler.....	16
2.2.21.3	Publishing Client	16
2.2.21.4	WebDAV Client	16
2.2.21.5	SOAP Client.....	16
2.2.22	ClientFileId Header	16
2.2.23	ContentChangeUnit Header	16
3	Protocol Details.....	17
3.1	Server Details.....	17
3.1.1	Abstract Data Model.....	17
3.1.2	Timers	17
3.1.3	Initialization.....	17
3.1.4	Higher-Layer Triggered Events	17
3.1.5	Message Processing Events and Sequencing Rules	17
3.1.5.1	Expires Header.....	17
3.1.5.2	Handling Files that are Intended to be Downloaded by a Registered Client ...	17
3.1.5.3	SharePointError Header	17

3.1.5.4	X-RequestDigest Header	18
3.1.5.5	Server Handling of the SOAPAction Header	18
3.1.5.6	Server Handling of the DELETE Verb	18
3.1.5.7	Handling Access Denied Scenarios on Different Clients.....	18
3.1.5.8	Handling Requests to Virus-Infected Resources.....	19
3.1.5.9	Handling Requests to IRM-Protected Resources	19
3.1.5.10	Handling Requests from Crawlers.....	20
3.1.5.11	Form Digest and Headers.....	20
3.1.5.12	Handling Multipart Content Types.....	20
3.1.5.13	Handling Requests that have not been Previously Authenticated	21
3.1.5.14	Handling Files that have been Checked Out	21
3.1.5.15	Enabling Clients or Applications to Make Requests on Behalf of the User	21
3.1.5.16	Providing Diagnostic Information	21
3.1.5.17	Handling Cultural Information.....	21
3.1.5.18	Handling Authentication Requests	22
3.1.5.19	X-SharePointHealthScore Header	22
3.1.6	Timer Events.....	22
3.1.7	Other Local Events.....	22
4	Protocol Examples	23
4.1	Request Using the Content-Disposition Header for a Thicket Supporting File.....	23
4.2	x-virus-infected Header	23
4.2.1	Client Request.....	23
4.2.2	Server Response	23
4.3	IRM Headers.....	23
4.3.1	Client Request.....	23
4.3.2	Response Using the x-irm-cantdecrypt Header	24
4.3.3	Response Using the x-irm-rejected Header	24
4.3.4	Response Using the x-irm-notowner Header.....	24
4.3.5	Response Using the x-irm-timeout Header	24
4.3.6	Response Using the x-irm-crashed Header.....	24
4.3.7	Response Using the x-irm-unknown-failure Header	25
4.4	Response Using the SharePointError Header	25
4.5	Deleting a Resource in a Document Library Forms Folder	25
4.5.1	Client Request.....	25
4.5.2	Server Response	26
5	Security	27
5.1	Security Considerations for Implementers	27
5.2	Index of Security Parameters	27
6	Appendix A: Product Behavior	28
7	Change Tracking.....	30
8	Index.....	32

1 Introduction

The HTTP Windows SharePoint Services Headers Protocol extends the HTTP mechanisms to include new headers and messages that enable previously undefined behaviors, such as authenticating client connections, communicating error conditions, sending complex data, and interacting with Information Rights Management (IRM) systems, antivirus systems, and Web crawlers.

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:

absolute URL: The full Internet address of a page or other World Wide Web resource. The absolute URL includes a protocol, such as "http," a network location, and an optional path and file name — for example, <http://www.treyresearch.net/>.

antivirus status page: A page that is presented to a protocol client and displays antivirus information for the requested resource.

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 [\[RFC5234\]](#).

authentication mode: One of several modes in which an authentication exchange may be performed.

crawl: The process of traversing a URL space to acquire items to record in a search catalog.

crawler: A process that browses and indexes content from a content source.

document library: A type of list that is a container for documents and folders.

form digest: An object that is inserted into a page and is used by a protocol server to validate client requests. The validation is specific to a user, site, and time period.

forms authentication: An authentication (2) method in which protocol clients redirect unauthenticated requests to an HTML form by using **HTTP**. If the protocol client authenticates the request, the system issues a cookie that stores the credentials or a key for reacquiring the identity. In subsequent requests, the cookie is submitted in request headers and the requests are authenticated and authorized by an ASP.NET event handler that uses the validation method that is specified by the protocol client.

front-end web server: A server that hosts webpages, performs processing tasks, and accepts requests from protocol clients and sends them to the appropriate back-end server for further processing.

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.

Information Rights Management (IRM): A technology that provides persistent protection to digital data by using encryption, certificates (1), and authentication (2). Authorized recipients or users acquire a license to gain access to the protected files according to the rights or business rules that are set by the content owner.

IRM protector: An application that converts a file to an encrypted format when a user downloads the file and to a nonencrypted format when a user uploads a rights-managed file. See also **Information Rights Management (IRM)**.

leaf name: The segment of a URL that follows the last slash. If the resource is a directory, the leaf name can be an empty string (1).

MIME Encapsulation of Aggregate HTML Documents (MHTML): A MIME-encapsulated HTML document, as described in [\[RFC2557\]](#).

permission: A rule that is associated with an object and that regulates which users can gain access to the object and in what manner. See also rights.

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 (2) using X.509 certificates (2). 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 [\[SSL3\]](#).

site-relative URL: A URL that is relative to the site that contains a resource and does not begin with a leading slash (/).

thicket: A means of storing a complex HTML document with its related files. It consists of a thicket main file and a hidden thicket folder that contains a thicket manifest and a set of thicket supporting files that, together, store the referenced content of the document.

thicket supporting file: A file that contains a graphic element, a picture, or other media that is referenced by the thicket main file and is stored in the thicket folder.

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. See [\[RFC4346\]](#).

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 [\[RFC1738\]](#).

User-Agent header: An HTTP request-header field, as described in [\[RFC2616\]](#). It contains information about the user agent that originated a request.

virus scanner: Software that is used to search for and remove computer viruses, worms, and Trojan horses.

web crawler: A search component that traverses websites, downloads content from those sites, and submits that content for indexing.

WebDAV client: A computer that uses WebDAV, as described in [\[RFC2518\]](#) or [\[RFC4918\]](#), to retrieve data from a WebDAV server.

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-FSSHTTP] Microsoft Corporation, "[File Synchronization via SOAP over HTTP Protocol](#)".

[MS-WDV] Microsoft Corporation, "[Web Distributed Authoring and Versioning \(WebDAV\) Protocol: Client Extensions](#)".

[MS-WSSFO2] Microsoft Corporation, "[Windows SharePoint Services \(WSS\): File Operations Database Communications Version 2 Protocol](#)".

[RFC2046] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996, <http://www.rfc-editor.org/rfc/rfc2046.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>

[RFC2183] Troost, R., Dorner, S., and Moore, K., Ed., "Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field", RFC 2183, August 1997, <http://www.rfc-editor.org/rfc/rfc2183.txt>

[RFC2246] Dierks, T., and Allen, C., "The TLS Protocol Version 1.0", RFC 2246, January 1999, <http://www.rfc-editor.org/rfc/rfc2246.txt>

[RFC2518] Goland, Y., Whitehead, E., Faizi, A., et al., "HTTP Extensions for Distributed Authoring - WebDAV", RFC 2518, February 1999, <http://www.ietf.org/rfc/rfc2518.txt>

[RFC2557] Palme, J., Hopmann, A., and Shelness, N., "MIME Encapsulation of Aggregate Documents, such as HTML (MHTML)", RFC 2557, March 1999, <http://www.rfc-editor.org/rfc/rfc2557.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>

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", May 2000, <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

1.2.2 Informative References

[MC-FPSEWM] Microsoft Corporation, "[FrontPage Server Extensions: Website Management Protocol](#)".

1.3 Protocol Overview (Synopsis)

This protocol is used by protocol client and protocol server applications to communicate specific application behaviors, such as authenticating connections, communicating error conditions, sending complex data, and interacting with **Information Rights Management (IRM)** systems, antivirus systems, and **Web crawlers**. The modifications specified by this protocol affect a number of other base protocols.

This protocol specifies the following extensions to the base **HTTP**, as described in [\[RFC2616\]](#):

- An unused **Exires** [*sic*] header specified in **Exires Header** (section [2.2.1](#)).
- An extension to the **Content-Disposition** header to support **thickets**. The extended **Content-Disposition** header is specified in **Content-Disposition Header** (section [2.2.2](#)).

- A header to indicate the presence of a virus in documents. The **x-virus-infected** header is specified in **x-virus-infected Header** (section [2.2.3](#)).
- A header to indicate Information Rights Management (IRM) decryption problems. The **x-irm-cantdecrypt** header is specified in **x-irm-cantdecrypt** (section [2.2.4](#)).
- A header to indicate an IRM rejection. The **x-irm-rejected** header is specified in **x-irm-rejected Header** (section [2.2.5](#)).
- A header to indicate conflicts for IRM ownership. The **x-irm-notowner** header is specified in **x-irm-notowner Header** (section [2.2.6](#)).
- A header to indicate an IRM time out. The **x-irm-timeout** header is specified in **x-irm-timeout Header** (section [2.2.7](#)).
- A header to indicate an IRM software failure. The **x-irm-crashed** header is specified in **x-irm-crashed Header** (section [2.2.8](#)).
- A header to indicate an unidentified IRM failure. The **x-irm-unknown-failure** header is specified in **x-irm-unknown-failure Header** (section [2.2.9](#)).
- A header to indicate **front-end Web server** errors. The **SharePointError** header is specified in **SharePointError Header** (section [2.2.10](#)).
- A header that contains the **form digest**. The **X-RequestDigest** header is specified in **X-RequestDigest Header** (section [2.2.11](#)).
- A header that indicates form authentication is required for the request to succeed. The **X-Forms_Based_Auth_Required** header is specified in **X-Forms_Based_Auth_Required Header** (section [2.2.12](#)).
- A header that specifies the URL to which the protocol client is redirected after a successful authentication. The **X-Forms_Based_Auth_Return_Url** header is specified in **X-Forms_Based_Auth_Return_Url Header** (section [2.2.13](#)).
- A header that indicates that a file is checked out after being uploaded to the protocol server. The **X-MS-File-Checked-Out** header is specified in **X-MS-File-Checked-Out Header** (section [2.2.14](#)).
- A header that enables a protocol client or a third-party application to make a request on behalf of a user. This header is specified in section [2.2.15](#).
- A header that enables a protocol client to obtain log information from the protocol server. This header is specified in section [2.2.16](#).
- A header that causes the protocol server to use the primary culture information associated with a site when retrieving information sensitive to culture settings. This header is specified in section [2.2.17](#).
- A header that causes the protocol server to return a 401 error code if the protocol client is not authenticated by the protocol server, even when the client does have permission to the requested resource. This header is specified in section [2.2.18](#).
- A header that the protocol server uses to return a server health status to the client. This header is specified in section [2.2.19](#).
- A header that the protocol server uses to identify the file contents on the protocol client. This header is specified in section [2.2.22](#).
- A header that the protocol server uses to identify the synchronization version of the file contents. This header is specified in section [2.2.23](#).

This protocol specifies the following extensions to the base WebDAV Protocol, as described in [\[RFC2518\]](#).

- A behavior change for the **DELETE** verb.

1.4 Relationship to Other Protocols

This protocol is an extension of the HTTP protocol as described in [\[RFC2616\]](#). It also includes aspects of the following protocols:

- MIME Part Two as described in [\[RFC2046\]](#).
- WebDAV as described in [\[RFC2518\]](#).
- SOAP 1.1 as described in [\[SOAP1.1\]](#).
- The FrontPage Server Extensions: Web Management Specification as described in [\[MC-FPSEWM\]](#).
- Web Distributed Authoring and Versioning (WebDAV) Protocol: Server Extensions as described in [\[MS-WDV\]](#).

1.5 Prerequisites/Preconditions

This protocol requires a protocol server that supports the HTTP protocol as described in [\[RFC2616\]](#).

Portions of this protocol require a protocol server that supports the WebDAV protocol as described in [\[RFC2518\]](#).

1.6 Applicability Statement

This protocol applies in the following scenarios:

- A browser protocol client interacts with a front-end Web server.
- A protocol client implements file operations through WebDAV as described in [\[RFC2518\]](#).
- An IRM client interacts with a front-end Web server.
- An antivirus system interacts with a front-end Web server.
- A crawler needs to **crawl** content sources.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Messages are transported using HTTP, as specified in [\[RFC2518\]](#) and [\[RFC2616\]](#).

This protocol MAY [<1>](#) be used with **Secure Sockets Layer (SSL)/Transport Layer Security (TLS)**, as specified in [\[RFC2246\]](#).

Port 80 is the standard port assignment for HTTP, and port 443 is the standard port assignment for HTTP over SSL/TLS. Implementations MAY [<2>](#) support other ports.

2.2 Message Syntax

This section specifies the protocol syntax and the data types that are used when protocol clients post messages to the protocol server. It also specifies the syntax that is used by the protocol server to respond to protocol client requests. The syntax and data types are defined by using **Augmented Backus-Naur Form (ABNF)** as specified in [\[RFC2616\]](#), section 2.1.

2.2.1 Expires Header

This protocol uses the Expires [*sic*] header. This header is defined in ABNF syntax as follows. [<3>](#)

```
expires-header = "Expires" ":" expires-value
expires-value = *TEXT
```

The **Expires** [*sic*] header and its value have no meaning. The protocol server SHOULD use the **Expires** header instead as specified in [\[RFC2616\]](#) section 14.21. [<4>](#)

2.2.2 Content-Disposition Header

This protocol uses the **Content-Disposition** header as specified in [\[RFC2183\]](#) and [\[RFC2616\]](#) section 19.5.1.

This protocol also extends the **Content-Disposition** header with the *thicket* and *name* parameters as follows:

- The *thicket* parameter indicates whether the content is part of a **thicket supporting file**. If the **thicket-value** element is set to 1, the content is part of a thicket supporting file. If the **thicket-value** element is set to zero, the content is not part of a thicket supporting file.
- The *name* parameter indicates the name of the field with which the content is associated.

These parameters are specified in ABNF syntax as follows.

```
thicket-parameter = "thicket" "=" thicket-value
thicket-value = "0" | "1"
name-parameter = "name" "=" name-value
name-value = 1*TEXT
```

2.2.3 x-virus-infected Header

This protocol uses the **x-virus-infected** response header to indicate that the requested resource cannot be returned because it has been infected by a virus.

This header is defined in ABNF syntax as follows.

```
virus-infected-header = "x-virus-infected" ":" virus-information-value
virus-information-value = 1*TEXT
```

The **virus-information-value** element is a text string that specifies the virus information of the requested resource. This string is not significant to protocol operation and is intended only for display and logging purposes.

2.2.4 x-irm-cantdecrypt Header

This protocol uses the **x-irm-cantdecrypt** response header to indicate that the requested resource cannot be returned because the **IRM protector** is unable to decrypt the requested resource.

This header is defined in ABNF syntax as follows. [<5>](#)

```
irm-cantdecrypt-header = "x-irm-cantdecrypt" ":" irm-information-value
irm-information-value = 1*TEXT
```

The **irm-information-value** element is a text string that specifies the IRM information of the requested resource. This string is not significant to protocol operation and is intended only for display and logging purposes.

2.2.5 x-irm-rejected Header

This protocol uses the **x-irm-rejected** response header to indicate that the requested resource cannot be returned because the IRM system does not accept files of the requested type.

This header is defined in ABNF syntax as follows.

```
irm-rejected-header = "x-irm-rejected" ":" irm-information-value
irm-information-value = 1*TEXT
```

The **irm-information-value** element is a text string that specifies the IRM information of the requested resource. This string is not significant to protocol operation and is intended only for display and logging purposes.

2.2.6 x-irm-notowner Header

The **x-irm-notowner** header is reserved and the protocol client MUST ignore it. [<6>](#)

This **x-irm-notowner** header is defined in ABNF syntax as follows.

```
irm-notowner-header = "x-irm-notowner" ":" irm-information-value
irm-information-value = 1*TEXT
```

The **irm-information-value** element is a text string.

2.2.7 x-irm-timeout Header

This protocol uses the **x-irm-timeout** response header to signal that the requested resource cannot be returned because the IRM system has timed out.

This header is defined in ABNF syntax as follows.

```
irm-timeout-header = "x-irm-timeout" ":" irm-information-value
irm-information-value = 1*TEXT
```

The **irm-information-value** element is a text string that specifies the IRM information of the requested resource. This string is not significant to protocol operation and is intended only for display and logging purposes.

2.2.8 x-irm-crashed Header

This protocol uses the **x-irm-crashed** response header to indicate that the requested resource cannot be returned because the IRM system has crashed.

This header is defined in ABNF syntax as follows.

```
irm-crashed-header = "x-irm-crashed" ":" irm-information-value
irm-information-value = 1*TEXT
```

The **irm-information-value** element is a text string that specifies the IRM information of the requested resource. This string is not significant to protocol operation and is intended only for display and logging purposes.

2.2.9 x-irm-unknown-failure Header

This protocol uses the **x-irm-unknown-failure** response header to indicate that the requested resource cannot be returned because of an unknown failure in the IRM system.

This header is defined in ABNF syntax as follows.

```
irm-unknown-failure-header = "x-irm-unknown-failure" ":" irm-information-value
irm-information-value = 1*TEXT
```

The **irm-information-value** element is a text string that specifies the IRM information of the requested resource. This string is not significant to protocol operation and is intended only for display and logging purposes.

2.2.10 SharePointError Header

This protocol uses the **SharePointError** header to indicate that a protocol server error has occurred.

This header is defined in ABNF syntax as follows.

```
sharepointerror-header = "SharePointError" ":" "0"/"2"
```

2.2.11 X-RequestDigest Header

This protocol uses the **X-RequestDigest** header as an alternative to the `__REQUESTDIGEST` form parameter on an HTTP POST request.

This header is defined in ABNF syntax as follows.

```
requestdigest-header = "X-RequestDigest" ":" requestdigest-value
```

```
requestdigest-value = 1*TEXT
```

The **requestdigest-value** element is the form digest.

2.2.12 X-Forms_Based_Auth_Required Header

This protocol uses the **X-Forms_Based_Auth_Required** header for handling requests that have not been previously authenticated.

This header is defined in ABNF syntax as follows.

```
x-forms_based_auth_required-header = "X-Forms_Based_Auth_Required" ":" x-  
forms_based_auth_required-value  
x-forms_based_auth_required-value = 1*TEXT
```

The **x-forms_based_auth_required-value** element is the **URL** of a form authentication logon page.

2.2.13 X-Forms_Based_Auth_Return_Url Header

The **x-forms_based_auth_return_url-value** element is a URL to which the client is redirected after a successful **forms authentication**.

This header is defined in ABNF syntax as follows.

```
x-forms_based_auth_return_url-header = "X-Forms_Based_Auth_Return_Url" ":" x-  
forms_based_auth_return_url-value  
x-forms_based_auth_return_url-value = 1*TEXT
```

If this header is missing, it is equivalent to the header being specified with the value of the URL of the original request.

2.2.14 X-MS-File-Checked-Out Header

This protocol uses the **X-MS-File-Checked-Out** header for handling files that have been checked out after being uploaded to the protocol server.

This header is specified in ABNF syntax as follows.

```
X-MS-File-Checked-Out-header = "X-MS-File-Checked-Out" : X-MS-File-Checked-Out-value  
X-MS-File-Checked-Out-value = "1"
```

2.2.15 X-RequestToken Header

This protocol uses the **X-RequestToken** value to enable the protocol client or a third-party application to make requests on behalf of the current user. The **X-RequestToken** value is an opaque string.

This header is defined in ABNF syntax as follows.

```
requesttoken-header = "X-RequestToken" ":" requestdigest-value  
requesttoken-value = 1*TEXT
```

2.2.16 SPRequestGuid Header

This protocol uses the **SPRequestGuid** header for communicating diagnostic information about server problems. This header **MUST** be a GUID. The GUID provides diagnostic information to users.

This header is defined in ABNF syntax as follows.

```
sprequestguid-header = "SPRequestGuid" ":" sprequestguid-value
sprequestguid-value = 1*TEXT
```

2.2.17 X-UseWebLanguage Header

This protocol uses the **X-UseWebLanguage** header [<7>](#) to provide the primary culture information.

This header is defined in ABNF syntax as follows.

```
x-useweblanguage-header = "X-UseWebLanguage" ":" x-useweblanguage-value
x-useweblanguage-value = "true"
```

2.2.18 X-RequestForceAuthentication Header

This protocol uses the **X-RequestForceAuthentication** header [<8>](#) to force authentication by the server.

This header is defined in ABNF syntax as follows.

```
x-requestforceauthentication-header = "X-RequestForceAuthentication" ":" x-
requestforceauthentication-value
x-requestforceauthentication-value = "true"
```

2.2.19 X-SharePointHealthScore

This protocol uses the **X-SharePointHealthScore** header to indicate the current server health status. The header **MUST** be an integer in the range from 0 to 10.

This header is defined in ABNF syntax as follows.

```
x-sharepointhealthscore-header = "X-SharePointHealthScore" ":"
"0"/"1"/"2"/"3"/"4"/"5"/"6"/"7"/"8"/"9"/"10"
```

2.2.20 X-MS-InvokeApp Header

This protocol uses the **X-MS-InvokeApp** response header to signal that the client can pass the document URL directly to a registered client application rather than downloading the document first. This header is defined in ABNF syntax as follows:

```
X-MS-InvokeApp = ("1; RequireReadOnly")
```

2.2.21 Distinguishing Clients with HTTP Headers

This protocol covers multiple application scenarios. To distinguish behaviors among protocol clients, the HTTP headers in the following sections MUST be sent by protocol clients and be recognized by front-end Web servers.

2.2.21.1 Browser Client

A protocol client MUST include "Mozilla" (case-sensitive) in the **User-Agent header** of its requests to be considered a browser client.

A protocol client MUST NOT include "FrontPage" (case-sensitive), "Office" (case-sensitive), or "non-browser" (case-sensitive) in the **User-Agent** header of its requests to be considered a browser client.

2.2.21.2 Crawler

A protocol client MUST include both "MS Search" (case-sensitive) and "Robot" (case-sensitive) in that order in the User-Agent header of its requests to be considered a **crawler**.

A crawler client MUST also be considered a browser client as specified in section [2.2.21.1](#).

2.2.21.3 Publishing Client

A protocol client MUST include "Microsoft Data Access Internet Publishing Provider" (case-sensitive) in the value of the **User-Agent** header of its requests to be considered a publishing client.

2.2.21.4 WebDAV Client

A protocol client MUST include a **Translate** header as specified in [\[MS-WDV\]](#) section [2.2.2](#), with a value beginning with "F" in its requests to be considered a WebDAV client as specified in [\[RFC2518\]](#).

2.2.21.5 SOAP Client

A protocol client MUST include the **SOAPAction** header as specified in [\[SOAP1.1\]](#) section 6.1.1 or a Content-Type header with a value beginning with "application/soap+xml" to be considered a SOAP client as specified in [\[RFC2616\]](#) section 14.17.

2.2.22 ClientFileId Header

A protocol client MAY include the **ClientFileId** header in a PUT request to specify a string value that uniquely identifies the file contents on the protocol client, and is stored by the protocol server. For more details, see [\[MS-FSSHTTP\]](#), section 2.3.3.1. [<9>](#)

2.2.23 ContentChangeUnit Header

A protocol client MAY include the **ContentChangeUnit** header in a PUT request to specify a string value that uniquely identifies the synchronization version of the file contents. For more details, see [\[MS-FSSHTTP\]](#), section 2.3.3.1. [<10>](#)

3 Protocol Details

3.1 Server Details

This protocol specifies communication between front-end Web server and protocol client applications. Protocol clients **MUST** distinguish their requests as specified in Distinguishing Clients with HTTP Headers (section [2.2.17](#)).

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with the behavior described in this document.

No abstract data models are required except those specified in [\[RFC2616\]](#).

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 Expires Header

The protocol server **SHOULD NOT** include the **Expires** [*sic*] header in any response. [<11>](#)

The protocol client **MUST** ignore the **Expires** [*sic*] header.

3.1.5.2 Handling Files that are Intended to be Downloaded by a Registered Client

The protocol server can add the **X-MS-InvokeApp** header as specified in section [2.2.20](#) to the PUT response if the file is intended to be downloaded by a registered client.

3.1.5.3 SharePointError Header

If the protocol server rejects the request because the current processing load on the server exceeds its capacity, the protocol server **SHOULD** include a **SharePointError** header set to 2 in the response. If the protocol server renders an error page to the client for any other reason, the protocol server **MUST** include a **SharePointError** header set to zero in the response.

3.1.5.4 X-RequestDigest Header

When an HTTP POST request is received, if the form digest is to be checked and the `__REQUESTDIGEST` form parameter does not exist, the server MUST check for the **X-RequestDigest** header value and use that value as the form digest.

3.1.5.5 Server Handling of the SOAPAction Header

When an HTTP POST request is received, if the protocol client is considered to be a SOAP client, the protocol server MUST check for the **UseRemoteAPIs permission** as specified in [\[MS-WSSFO2\]](#) section 2.2.2.14.

3.1.5.6 Server Handling of the DELETE Verb

When a **DELETE** request is received, the protocol server SHOULD handle the request as specified in [\[RFC2518\]](#) section 8.6 and [\[RFC2616\]](#) section 9.7.<12>

3.1.5.7 Handling Access Denied Scenarios on Different Clients

For protocol clients that do not have permission to a requested resource, front-end Web servers provide protocol clients with responses as specified in [\[RFC2616\]](#), section 10.4. The correct response varies based on the following factors:

- Whether the protocol client is a browser.
- The **authentication mode** used during the exchange.
- The authentication state of the protocol client.
- Whether the request is a SOAP request.

If a protocol client has not been authenticated, the protocol client is a browser, and the request is not a SOAP request, and the authentication mode is not Forms Authentication, the protocol server MUST return an HTTP response with status code 401 as specified in [\[RFC2616\]](#), section 10.4.2. If a protocol client has not been authenticated, the protocol client is a browser, and the request is not a SOAP request, and the authentication mode is Forms Authentication, the protocol server MUST redirect the protocol client to the logon page.

If a protocol client has not been authenticated, the protocol client is not a browser or the request is a SOAP request, and the authentication mode is not Forms Authentication, the protocol server MUST return an HTTP response with status code 401 as specified in [\[RFC2616\]](#), section 10.4.2.

If a protocol client has not been authenticated, the protocol client is not a browser or the request is a SOAP request, and the authentication mode is Forms Authentication, the protocol server MUST return an HTTP response with status code 403 as specified in [\[RFC2616\]](#), section 10.4.4.

If a protocol client has been authenticated, the protocol client is a browser, the request is not a SOAP request, and the authentication mode is not Forms Authentication, the protocol server MUST return an HTTP response with status code 401 as specified in [\[RFC2616\]](#), section 10.4.2.

If a protocol client has been authenticated, the protocol client is a browser, the request is not a SOAP request, and the authentication mode is Forms Authentication, the protocol server MUST redirect the protocol client to the login page.

If a protocol client has been authenticated, the protocol client is not a browser or the request is a SOAP request, and the authentication mode is Forms Authentication, the protocol server MUST return an HTTP response with status code 403 as specified in [\[RFC2616\]](#) section 10.4.4.

If a protocol client has been authenticated, the protocol client is not a browser or the request is a SOAP request, and the authentication mode is not Forms Authentication, the protocol server MUST return an HTTP response with status code 401 as specified in [RFC2616] section 10.4.2.

3.1.5.8 Handling Requests to Virus-Infected Resources

When an antivirus system is active and a request is received while the antivirus system is scanning incoming or outgoing documents, the front-end Web server MUST support the behavior specified in this section.

When a request is received and the requested resource is discovered to be infected by the protocol server **virus scanner**, the front-end Web server responds as follows:

- If the protocol client is a **WebDAV client** or a publishing client, as specified in section 2.2.21, and the virus status of the requested resource is not clean, the protocol server MUST return an HTTP response with status code 409 as specified in [RFC2616] section 10.4.10. The protocol server response MUST include an **x-virus-infected** header with a value that contains virus status information from the virus scanner.
- If the protocol client is not a WebDAV client or a publishing client as specified in section 2.2.21 and the document is infected, infected and cleanable, deleted by the virus scanner, or the virus scanner timed out, the protocol server responds as follows:
 - The protocol server MUST return an HTTP response with status code 409 as specified in [RFC2616] section 10.4.10.
 - The protocol server MUST include an **x-virus-infected** header whose value contains virus status information from the virus scanner with the response.
 - The protocol server MUST include a content-location header as specified in [RFC2557] section 4.2. Its value contains the URL of the **antivirus status page**.<13>
 - The body of the response MUST be an antivirus status page.
- If the protocol client is not a WebDAV client or a publishing client as specified in section 2.2.21 and the document is cleaned or clean failed, the server MUST redirect the protocol client to an antivirus status page.

3.1.5.9 Handling Requests to IRM-Protected Resources

When an IRM protector is active, the front-end Web server MUST support the behavior specified in this section.

When a request is received and the requested resource cannot be retrieved because of IRM, the protocol server responds as follows:

- The protocol server MUST return an HTTP response with status code 409 as specified in [RFC2616] section 10.4.10.
- The protocol server MUST include one of the following headers as appropriate: **x-irm-cantdecrypt**, **x-irm-rejected**, **x-irm-notowner**, **x-irm-timeout**, **x-irm-crashed**, or **x-irm-unknown-failure**.
- The value of the header MUST contain a string explaining the IRM error.

If the protocol client is not a WebDAV client or a publishing client, as specified in section 2.2.17, the protocol server MAY include a **Content-Location** header<14> as specified in [RFC2616] section 14.14 with a value that contains the **absolute URL** of the IRM report page. The body of the response MAY be the content of an IRM report page.

3.1.5.10 Handling Requests from Crawlers

When a request is received from a crawler, the requested resource is IRM protected, and the authenticated user has the **EnumeratePermissions** permission, as specified in [\[MS-WSSF02\]](#) section 2.2.2.14, the protocol server MUST NOT IRM encrypt the requested resource that is returned to the protocol client.

When a request is received, if the protocol client is a crawler, the authenticated user has the **EnumeratePermissions** permission, and the front-end Web server antivirus skip search crawl setting is turned on, the protocol server MUST NOT scan the requested resource for viruses before returning the resource to the protocol client.

3.1.5.11 Form Digest and Headers

When an HTTP POST request is received, if the protocol client is a SOAP client, as specified in section [2.2.21.5](#), or a FrontPage Server Extensions Web Management client, as specified in section [2.2.21.4](#), the protocol server responds as follows:

- The protocol server MUST ignore the `__REQUESTDIGEST` parameter in the POST request.
- The protocol server MUST ignore the **X-RequestDigest** header.

3.1.5.12 Handling Multipart Content Types

When an HTTP POST request containing a **MIME Encapsulation of Aggregate HTML Documents (MHTML)**, as specified in [\[RFC2557\]](#), is received and the **Content-Type** header, as specified in [\[RFC2616\]](#) section 14.17, begins with "multipart", the protocol server responds as follows:

- If the *boundary* parameter of the **Content-Type** header includes a comma (,), the protocol server MUST ignore all characters from the first comma to the end of the boundary parameter, including the comma.

When parsing an HTTP POST request to the **site-relative URL** `"/_vti_bin/shtml.dll"`, which contains a multipart content-type as specified in [\[RFC2046\]](#) section 5.1, the protocol server responds as follows:

- If the **Content-Type** header has a *charset* parameter, as specified in [\[RFC2616\]](#) section 3.4, the protocol server MUST determine the character encoding from that parameter. Otherwise, if the **Accept-Language** header exists, the protocol server MUST infer the character encoding from that **Accept-Language** header if possible. The **Accept-Language** header is specified in [\[RFC2616\]](#) section 14.4.
- If the **Content-Disposition** header, as specified in [\[RFC2183\]](#), has a *filename* parameter and the *thicket* parameter is set to 1, the request or request part is a thicket supporting file to be uploaded, and the protocol server MUST infer both the thicket folder name and the **leaf name** of the file from the *filename* parameter.
- If the **Content-Disposition** header has a *filename* parameter and the *thicket* parameter is not set to 1, the request or request part is a document to be uploaded, and the protocol server MUST infer the leaf name of the document from the *filename* parameter.
- If the **Content-Disposition** header has no *filename* parameter and starts with "form-data", the request or request part is a form field, and the protocol server MUST infer the field name from the *name* parameter of the **Content-Disposition** header.

3.1.5.13 Handling Requests that have not been Previously Authenticated

If the protocol server uses forms authentication and the protocol client sends a request without being authenticated first, the protocol server SHOULD [<15>](#) add the **X-Forms_Based_Auth_Required** header (see section [2.2.12](#)) to the response.

If the protocol server uses forms authentication and the protocol client sends a request without being authenticated first, the server SHOULD [<16>](#) add the **X-Forms_Based_Auth_Return_Url** header (see section [2.2.13](#)) to the response.

3.1.5.14 Handling Files that have been Checked Out

The protocol server SHOULD [<17>](#) add the **X-MS-File-Checked-Out** header (see section [2.2.14](#)) to the PUT response if the file is checked out after being uploaded to the protocol server.

3.1.5.15 Enabling Clients or Applications to Make Requests on Behalf of the User

The protocol server can send an **X-RequestToken** value (see section [2.2.15](#)) to the protocol client or a third-party application to enable the protocol client or a third-party application to make requests on behalf of the current user. The **X-RequestToken** [<18>](#) value is generated by the protocol server and it is an opaque string to the protocol client. The content of the **X-RequestToken** value is protocol server implementation-specific. The protocol server MAY [<19>](#) include information, such as the protocol client application identifier, the current user identifier, time stamp, and their hash values in the **X-RequestToken** value and any other information necessary for the protocol server to execute request. When the protocol client or a third-party application needs to make requests on behalf of the current user, the protocol client or a third-party application can include the **X-RequestToken** value in the **X-RequestToken** header in a new request to the protocol server. The protocol server can read the user identifier from this header and execute the request on behalf of the user specified by the user identifier. The protocol server can further use the hash value to ensure that the value of this token has not been tampered with. The protocol server could also define an expiration period and ensure that the token has not expired by checking the time stamp.

3.1.5.16 Providing Diagnostic Information

The protocol server can send back a GUID in a **SPRequestGuid** header (see section [2.2.16](#)) in an HTTP response to facilitate diagnosing protocol server problems. The protocol server MAY [<20>](#) log information pertinent to the request, including this GUID, in an implementation-specific way. The protocol server MUST NOT have any other dependencies on the value of this GUID. The user of this protocol could search for the GUID obtained from the HTTP response in any logs on the protocol server for more information about the request.

3.1.5.17 Handling Cultural Information

The protocol server MUST store primary culture information with every site. The culture information is used to determine the language to be used when the protocol server returns culture sensitive information to the protocol client, such as error strings. How such culture information is defined, stored, and used is implementation-specific.

If the **X-UseWebLanguage** header (see section [2.2.17](#)) is sent to the protocol server in a request whose request URI ends with `"/_vti_bin/client.svc"`, the protocol server MUST process the request, using the primary culture information to retrieve any culture sensitive information.

If the protocol client does not specify this header, the protocol server can use a language other than the primary culture defined on the site to return information to the protocol client.

3.1.5.18 Handling Authentication Requests

If the client sends the **X-RequestForceAuthentication** header (see section [2.2.18](#)) to the server in a request whose request URI ends with `"/_vti_bin/client.svc"`, or `"/_vti_bin/sites.asmx"`, and if protocol client has not been authenticated, and the anonymous user does not have permission access to the requested resource, the protocol server MUST return an HTTP response with status code 401 as specified in [\[RFC2616\]](#), section 10.4.2.

If this header is sent in a request whose request URI ends with `"/_vti_bin/client.svc"`, or `"/_vti_bin/sites.asmx"`, and if protocol client has not been authenticated, and the anonymous user has permission access to the requested resource, the protocol server MAY return an HTTP response with status code 401 as specified in [\[RFC2616\]](#), section 10.4.2.

3.1.5.19 X-SharePointHealthScore Header

The server can monitor the current load and its ability to process requests. If it monitors the load, it can return the load information to the client in a **X-SharePointHealthScore** header. This header specifies a value between 0 and 10, where 0 represents a low load and a high ability to process requests and 10 represents a high load and that the server is throttling requests to maintain adequate throughput.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

4 Protocol Examples

4.1 Request Using the Content-Disposition Header for a Thicket Supporting File

A protocol client uploads a thicket supporting file to the protocol server using the **Content-Disposition** header as follows.

```
PUT /rootsite/subsite/Shared%20Documents/eggplant_files/eggplant.jpg HTTP/1.1
Host: www.contoso.com
Accept: */*
Content-Type: image/jpeg
Content-Length: 32768
Content-Disposition: inline; filename=/eggplant_files/eggplant.jpg; thicket=1; name=Example
Connection: Keep-Alive
```

The body contains image data.

4.2 x-virus-infected Header

The eggplant.aspx file has been infected by a virus.

4.2.1 Client Request

A WebDAV client requests eggplant.aspx as follows.

```
GET /rootsite/subsite/Shared%20Documents/eggplant.aspx HTTP/1.1
Host: www.contoso.com
Accept: */*
User-Agent: Microsoft-WebDAV-MiniRedir
Translate: F
Connection: Keep-Alive
```

4.2.2 Server Response

The resource is infected by a virus, so the protocol server responds with the following.

```
HTTP/1.1 409 CONFLICT
Date: Thu, 13 Mar 2008 16:09:43 GMT
Server: Microsoft-IIS/6.0
Cache-Control: private
Content-Length: 0
Public-Extension: http://schemas.microsoft.com/repl-2
x-virus-infected: The requested document has been infected by a virus.
```

4.3 IRM Headers

The eggplant.aspx file is protected by IRM.

4.3.1 Client Request

A WebDAV client requests eggplant.aspx as follows.

```
GET /rootsite/subsite/Shared%20Documents/eggplant.aspx HTTP/1.1
```

```
Host: www.contoso.com
Accept: */*
User-Agent: Microsoft-WebDAV-MiniRedir
Translate: F
Connection: Keep-Alive
```

4.3.2 Response Using the x-irm-cantdecrypt Header

The IRM system is unable to decrypt the requested resource, so the protocol server responds with the following.

```
HTTP/1.1 409 CONFLICT
Date: Thu, 13 Mar 2008 16:09:43 GMT
Server: Microsoft-IIS/6.0
Cache-Control: private
Content-Length: 0
Public-Extension: http://schemas.microsoft.com/repl-2
x-irm-cantdecrypt: Cannot unprotect
```

4.3.3 Response Using the x-irm-rejected Header

The IRM system does not accept files of the requested type, so the protocol server responds with the following.

```
HTTP/1.1 409 CONFLICT
Date: Thu, 13 Mar 2008 16:09:43 GMT
Server: Microsoft-IIS/6.0
Cache-Control: private
Content-Length: 0
Public-Extension: http://schemas.microsoft.com/repl-2
x-irm-rejected: This library does not accept files of the given type.
```

4.3.4 Response Using the x-irm-notowner Header

The IRM system does not own the requested resource, so the protocol server responds with the following.

```
HTTP/1.1 409 CONFLICT
Date: Thu, 13 Mar 2008 16:09:43 GMT
Server: Microsoft-IIS/6.0
Cache-Control: private
Content-Length: 0
Public-Extension: http://schemas.microsoft.com/repl-2
x-irm-notowner: Wss is not the owner of this file.
```

4.3.5 Response Using the x-irm-timeout Header

The IRM system times out, so the protocol server responds with the following.

```
HTTP/1.1 409 CONFLICT
Date: Thu, 13 Mar 2008 16:09:43 GMT
Server: Microsoft-IIS/6.0
Cache-Control: private
Content-Length: 0
Public-Extension: http://schemas.microsoft.com/repl-2
x-irm-timeout: Protector timed out.
```


4.3.6 Response Using the x-irm-crashed Header

The IRM system crashed, so the protocol server responds with the following.

```
HTTP/1.1 409 CONFLICT
Date: Thu, 13 Mar 2008 16:09:43 GMT
Server: Microsoft-IIS/6.0
Cache-Control: private
Content-Length: 0
Public-Extension: http://schemas.microsoft.com/repl-2
x-irm-crashed: Protector crashed.
```

4.3.7 Response Using the x-irm-unknown-failure Header

The IRM system experienced an unknown failure, so the protocol server responds with the following.

```
HTTP/1.1 409 CONFLICT
Date: Thu, 13 Mar 2008 16:09:43 GMT
Server: Microsoft-IIS/6.0
Cache-Control: private
Content-Length: 0
Public-Extension: http://schemas.microsoft.com/repl-2
x-irm-unknown-failure: Error in IRM
```

4.4 Response Using the SharePointError Header

An error occurred while processing the protocol client request, so the protocol server responds with the error page as follows.

```
HTTP/1.1 307 TEMPORARY REDIRECT
Date: Thu, 13 Mar 2008 16:09:43 GMT
Server: Microsoft-IIS/6.0
Location: http://www.contoso.com/ layouts/error.html
Last-Modified: Thu, 13 Mar 2008, 16:09:42 GMT
Content-Type: text/html
Cache-Control: private
Content-Length: 249
Public-Extension: http://schemas.microsoft.com/repl-2
SharePointError: 0

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
  <head>
    <title>An error has occurred!</title>
  </head>
  <body>
    An error has occurred!
  </body>
</html>
```

4.5 Deleting a Resource in a Document Library Forms Folder

The AllItems.aspx file is in the Forms folder of the Shared Documents **document library**.

4.5.1 Client Request

A WebDAV client attempts to delete AllItems.aspx as follows.

```
DELETE /root/site/subsite/Shared%20Documents/Forms/AllItems.aspx HTTP/1.1
Host: www.contoso.com
Accept: */*
User-Agent: Microsoft-WebDAV-MiniRedir
Connection: Keep-Alive
```

4.5.2 Server Response

The protocol server does not delete AllItems.aspx, but responds with the following.

```
HTTP/1.1 200 OK
Date: Thu, 13 Mar 2008 16:09:43 GMT
Server: Microsoft-IIS/6.0
Cache-Control: private
Content-Length: 0
Public-Extension: http://schemas.microsoft.com/repl-2
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

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

- The 2007 Microsoft Office system
- Microsoft Office 2010 suites
- Microsoft Office 2013
- Windows SharePoint Services 2.0
- Windows SharePoint Services 3.0
- Microsoft SharePoint Foundation 2010
- Microsoft SharePoint Foundation 2013
- Windows 8.1 Update
- Microsoft Office 2016
- Windows 10 operating system
- 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 2.1](#): Client support for SSL/TLS is available only in Windows Vista operating system and Windows Server 2008 operating system with Service Pack 2 (SP2). WebDAV servers that run on Internet Information Services (IIS), or on Windows SharePoint Services 2.0, or on Windows SharePoint Services 3.0, or on SharePoint Foundation 2010 support SSL/TLS.

[<2> Section 2.1](#): Windows XP operating system and Windows Server 2003 operating system WebDAV clients support only port 80. Support for other ports is available only in Windows Vista and Windows Server 2008. The WebDAV client in Windows Vista and Windows Server 2008 uses port 80 by default for HTTP, and port 443 for HTTP over SSL/TLS. WebDAV servers that run on Internet Information Services (IIS), on Windows SharePoint Services 2.0, on Windows SharePoint Services 3.0, or on SharePoint Foundation 2010 support SSL/TLS.

[<3> Section 2.2.1](#): This header is not used by SharePoint Foundation 2010.

[<4> Section 2.2.1](#): In all versions of Windows SharePoint Services 3.0, the server sends an **Expires** [*sic*] header instead of an **Expires** header.

[<5> Section 2.2.4](#): This header is not used by Windows SharePoint Services 3.0, or by SharePoint Foundation 2010.

[<6> Section 2.2.6](#): This header is not returned by Windows SharePoint Services 2.0, Windows SharePoint Services 3.0, or SharePoint Foundation 2010.

<7> [Section 2.2.17](#): Windows SharePoint Services 2.0 and Windows SharePoint Services 3.0 do not implement the **X-UseWebLanguage** header.

<8> [Section 2.2.18](#): Windows SharePoint Services 2.0 and Windows SharePoint Services 3.0 do not implement the **X-RequestForceAuthentication** header.

<9> [Section 2.2.22](#): This header is ignored by Windows SharePoint Services 2.0, Windows SharePoint Services 3.0, SharePoint Foundation 2010, and SharePoint Foundation 2013.

<10> [Section 2.2.23](#): This header is ignored by Windows SharePoint Services 2.0, Windows SharePoint Services 3.0, SharePoint Foundation 2010, and SharePoint Foundation 2013.

<11> [Section 3.1.5.1](#): In all versions of Windows SharePoint Services 3.0, the protocol server sends an **Expires** [*sic*] header instead of an **Expires** header.

<12> [Section 3.1.5.6](#): In all versions of Windows SharePoint Services 3.0, when handling **DELETE** requests, if the protocol client includes a **User-Agent** header with a value that begins with "Microsoft-WebDAV-MiniRedir" (case-sensitive) and the requested URL either specifies a folder resource that is named "Forms" or if the resource is in a folder named "Forms", the protocol server sends a response with status code 200, and the resource is not deleted.

<13> [Section 3.1.5.8](#): Windows SharePoint Services 2.0, Windows SharePoint Services 3.0 and SharePoint Foundation 2010 do not include a new line before the **content-location** header.

<14> [Section 3.1.5.9](#): Windows SharePoint Services 2.0, Windows SharePoint Services 3.0 and SharePoint Foundation 2010 do not include a new line before the content-location header.

<15> [Section 3.1.5.13](#): Windows SharePoint Services 2.0 and Windows SharePoint Services 3.0 do not implement the **X-Forms_Based_Auth_Required** header.

<16> [Section 3.1.5.13](#): Windows SharePoint Services 2.0 and Windows SharePoint Services 3.0 do not implement the **X-Forms_Based_Auth_Return_Url** header.

<17> [Section 3.1.5.14](#): Windows SharePoint Services 2.0 and Windows SharePoint Services 3.0 do not implement the **X-MS-File-Checked-Out** header.

<18> [Section 3.1.5.15](#): Windows SharePoint Services 2.0 and Windows SharePoint Services 3.0 do not implement the **X-RequestToken** header.

<19> [Section 3.1.5.15](#): SharePoint Foundation 2010 server includes the protocol client application identifier, the current user identifier, time stamp and their hash values in the **X-RequestToken** header.

<20> [Section 3.1.5.16](#): Windows SharePoint Services 2.0 and Windows SharePoint Services 3.0 do not implement this header. SharePoint Foundation 2010 saves this GUID in several different types of logs on the server.

7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

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

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

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

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

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

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

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

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

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
6 Appendix A: Product Behavior	Updated product behavior section to accurately reflect supported products.	Y	Product behavior note updated.

8 Index

A

[Abstract data model](#) 17
 [server](#) 17
[Applicability](#) 10

B

[Browser client header](#) 16

C

[Capability negotiation](#) 10
[Change tracking](#) 30
[ClientFileId Header message](#) 16
[ContentChangeUnit Header message](#) 16
[Content-Disposition Header message](#) 11
[Crawler header](#) 16

D

[Data model - abstract](#) 17
 [server](#) 17
[Delete a resource from a document library example](#)
 25
 [client request](#) 25
 [server response](#) 26
[Distinguishing Clients with HTTP Headers message](#)
 16

E

Examples

[delete a resource from a document library](#) 25
[delete a resource from a document library - client request](#) 25
[delete a resource from a document library - server response](#) 26
[IRM header - client request](#) 23
[IRM header - overview](#) 23
[IRM header - x-irm-cantdecrypt response](#) 24
[IRM header - x-irm-crashed response](#) 24
[IRM header - x-irm-notowner response](#) 24
[IRM header - x-irm-rejected response](#) 24
[IRM header - x-irm-timeout response](#) 24
[IRM header - x-irm-unknown-failure response](#) 25
[SharePointError header](#) 25
[using the Content-Disposition header for a thicket supporting file](#) 23
[x-virus-infected header](#) 23
[x-virus-infected header - client request](#) 23
[x-virus-infected header - server response](#) 23
[Expires Header message](#) 11

F

[Fields - vendor-extensible](#) 10

G

[Glossary](#) 6

H

Headers

[browser client](#) 16
[Content-Disposition](#) 11
[crawler](#) 16
[Expires](#) 11
[publishing client](#) 16
[SharePointError](#) 13
[SOAP client](#) 16
[SPRequestGuid](#) 15
[WebDAV client](#) 16
[X-Forms Based Auth Required](#) 14
[X-Forms Based Auth Return Uri](#) 14
[x-irm-cantdecrypt](#) 12
[x-irm-crashed](#) 13
[x-irm-notowner](#) 12
[x-irm-rejected](#) 12
[x-irm-timeout](#) 12
[x-irm-unknown-failure](#) 13
[X-MS-File-Checked-Out](#) 14
[X-RequestDigest](#) 13
[X-RequestToken](#) 14
[x-virus-infected](#) 11
[Higher-layer triggered events](#) 17
 [server](#) 17

I

[Implementer - security considerations](#) 27
[Index of security parameters](#) 27
[Informative references](#) 8
[Initialization](#) 17
 [server](#) 17
[Introduction](#) 6
IRM header example
 [client request](#) 23
 [overview](#) 23
 [x-irm-cantdecrypt response](#) 24
 [x-irm-crashed response](#) 24
 [x-irm-notowner response](#) 24
 [x-irm-rejected response](#) 24
 [x-irm-timeout response](#) 24
 [x-irm-unknown-failure response](#) 25

L

[Local events](#) 22

M

Message processing

[access denied scenario](#) 18
[authentication request](#) 22
[checked out files](#) 21
[crawler request](#) 20
[cultural information](#) 21
[DELETE request](#) 18
[Expires header](#) 17
[form digest](#) 20

- [IRM-protected resource](#) 19
- [multipart content type](#) 20
- [overview](#) 17
- [providing diagnostic information](#) 21
- [request not previously authenticated](#) 21
- [request on behalf of the user](#) 21
- [SharePointError header](#) 17
- [SOAPAction header](#) 18
- [virus-infected resource](#) 19
- [X-RequestDigest header](#) 18
- [X-SharePointHealthScore header](#) 22
- Messages**
 - [ClientFileId Header](#) 16
 - [ContentChangeUnit Header](#) 16
 - [Content-Disposition Header](#) 11
 - [Distinguishing Clients with HTTP Headers](#) 16
 - [Expires Header](#) 11
 - [SharePointError Header](#) 13
 - [SPRequestGuid Header](#) 15
 - [syntax](#) 11
 - [transport](#) 11
 - [X-Forms Based Auth Required Header](#) 14
 - [X-Forms Based Auth Return Url Header](#) 14
 - [x-irm-cantdecrypt Header](#) 12
 - [x-irm-crashed Header](#) 13
 - [x-irm-notowner Header](#) 12
 - [x-irm-rejected Header](#) 12
 - [x-irm-timeout Header](#) 12
 - [x-irm-unknown-failure Header](#) 13
 - [X-MS-File-Checked-Out Header](#) 14
 - [X-MS-InvokeApp Header](#) 15
 - [X-RequestDigest Header](#) 13
 - [X-RequestForceAuthentication Header](#) 15
 - [X-RequestToken Header](#) 14
 - [X-SharePointHealthScore](#) 15
 - [X-UseWebLanguage Header](#) 15
 - [x-virus-infected Header](#) 11
- N**
 - [Normative references](#) 8
- O**
 - Other local events
 - [server](#) 22
 - [Overview \(synopsis\)](#) 8
- P**
 - [Parameters - security index](#) 27
 - [Preconditions](#) 10
 - [Prerequisites](#) 10
 - [Product behavior](#) 28
 - [Publishing client header](#) 16
- R**
 - [References](#) 7
 - [informative](#) 8
 - [normative](#) 8
 - [Relationship to other protocols](#) 10
 - [Request using the Content-Disposition header for a thicket supporting file example](#) 23

- S**
 - Security
 - [implementer considerations](#) 27
 - [parameter index](#) 27
 - Sequencing rules
 - [access denied scenario](#) 18
 - [authentication request](#) 22
 - [checked out files](#) 21
 - [crawler request](#) 20
 - [cultural information](#) 21
 - [DELETE request](#) 18
 - [Expires header](#) 17
 - [form digest](#) 20
 - [IRM-protected resource](#) 19
 - [multipart content type](#) 20
 - [overview](#) 17
 - [providing diagnostic information](#) 21
 - [request not previously authenticated](#) 21
 - [request on behalf of the user](#) 21
 - [SharePointError header](#) 17
 - [SOAPAction header](#) 18
 - [virus-infected resource](#) 19
 - [X-RequestDigest header](#) 18
 - [X-SharePointHealthScore header](#) 22
 - Server
 - [abstract data model](#) 17
 - [higher-layer triggered events](#) 17
 - [initialization](#) 17
 - [other local events](#) 22
 - [overview](#) 17
 - [timer events](#) 22
 - [timers](#) 17
 - [Server - overview](#) 17
 - [SharePointError header example](#) 25
 - [SharePointError Header message](#) 13
 - [SOAP client header](#) 16
 - [SPRequestGuid Header message](#) 15
 - [Standards assignments](#) 10
 - [Syntax](#) 11
 - T**
 - [Timer events](#) 22
 - [server](#) 22
 - [Timers](#) 17
 - [server](#) 17
 - [Tracking changes](#) 30
 - [Transport](#) 11
 - [Triggered events](#) 17
 - Triggered events - higher-layer
 - [server](#) 17
 - V**
 - [Vendor-extensible fields](#) 10
 - [Versioning](#) 10
 - W**
 - [WebDAV client header](#) 16
 - X**

[X-Forms Based Auth Required Header message](#) 14
[X-Forms Based Auth Return Url Header message](#)
14
[x-irm-cantdecrypt header example](#) 24
[x-irm-cantdecrypt Header message](#) 12
[x-irm-crashed header example](#) 24
[x-irm-crashed Header message](#) 13
[x-irm-notowner header example](#) 24
[x-irm-notowner Header message](#) 12
[x-irm-rejected header example](#) 24
[x-irm-rejected Header message](#) 12
[x-irm-timeout header example](#) 24
[x-irm-timeout Header message](#) 12
[x-irm-unknown-failure header example](#) 25
[x-irm-unknown-failure Header message](#) 13
[X-MS-File-Checked-Out Header message](#) 14
[X-MS-InvokeApp Header message](#) 15
[X-RequestDigest Header message](#) 13
[X-RequestForceAuthentication Header message](#) 15
[X-RequestToken Header message](#) 14
[X-SharePointHealthScore message](#) 15
[X-UseWebLanguage Header message](#) 15
[x-virus-infected header example](#) 23
 [client request](#) 23
 [server response](#) 23
[x-virus-infected Header message](#) 11