

[MS-PCCRTP]: Peer Content Caching and Retrieval: Hypertext Transfer Protocol (HTTP) Extensions

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.
- **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
12/05/2008	0.1	Major	Initial Availability
01/16/2009	0.1.1	Editorial	Revised and edited the technical content.
02/27/2009	0.1.2	Editorial	Revised and edited the technical content.
04/10/2009	0.1.3	Editorial	Revised and edited the technical content.
05/22/2009	0.2	Minor	Updated the technical content.
07/02/2009	1.0	Major	Updated and revised the technical content.
08/14/2009	2.0	Major	Updated and revised the technical content.
09/25/2009	3.0	Major	Updated and revised the technical content.
11/06/2009	4.0	Major	Updated and revised the technical content.
12/18/2009	4.1	Minor	Updated the technical content.
01/29/2010	4.1.1	Editorial	Revised and edited the technical content.
03/12/2010	4.1.2	Editorial	Revised and edited the technical content.
04/23/2010	4.1.3	Editorial	Revised and edited the technical content.
06/04/2010	4.1.4	Editorial	Revised and edited the technical content.
07/16/2010	4.1.4	No change	No changes to the meaning, language, or formatting of the technical content.
08/27/2010	4.1.4	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	4.1.4	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	4.1.4	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	4.1.4	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	4.1.4	No change	No changes to the meaning, language, or formatting of the technical content.
03/25/2011	4.1.4	No change	No changes to the meaning, language, or formatting of the technical content.
05/06/2011	4.1.4	No change	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
06/17/2011	4.2	Minor	Clarified the meaning of the technical content.

Contents

1	Introduction	5
1.1	Glossary	5
1.2	References.....	6
1.2.1	Normative References.....	6
1.2.2	Informative References	6
1.3	Overview	6
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.....	8
1.9	Standards Assignments	8
2	Messages.....	9
2.1	Transport.....	9
2.2	Message Syntax	9
3	Protocol Details	11
3.1	HTTP/1.1 Client Details.....	11
3.1.1	Abstract Data Model	11
3.1.2	Timers	11
3.1.3	Initialization	11
3.1.4	Higher-Layer Triggered Events.....	11
3.1.5	Message Processing Events and Sequencing Rules.....	11
3.1.5.1	Receiving a Response of a PeerDist-Supporting Request.....	11
3.1.6	Timer Events	11
3.1.7	Other Local Events	11
3.2	HTTP/1.1 Server Details	12
3.2.1	Abstract Data Model	12
3.2.2	Timers	12
3.2.3	Initialization	12
3.2.4	Higher-Layer Triggered Events.....	12
3.2.5	Message Processing Events and Sequencing Rules.....	12
3.2.5.1	Receiving a PeerDist-Supporting Request.....	12
3.2.6	Timer Events	13
3.2.7	Other Local Events	13
4	Protocol Examples.....	14
5	Security.....	15
5.1	Security Considerations for Implementers.....	15
5.2	Index of Security Parameters	15
6	Appendix A: Product Behavior	16
7	Change Tracking.....	18
8	Index	20

1 Introduction

This document specifies a new content encoding, **PeerDist**, that can be used in HTTP/1.1. In particular, it specifies the mechanism used by an HTTP/1.1 client and an HTTP/1.1 server to communicate to each other using the PeerDist content encoding.

1.1 Glossary

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

HTTP client
HTTP server
Hypertext Transfer Protocol (HTTP)
Transmission Control Protocol (TCP)

The following terms are specific to this document:

Accept-Encoding: The **HTTP** header that defines the type of **content coding** that the client will accept from the server as part of the **HTTP** response. See [\[RFC2616\]](#) section 14.3 for details.

content coding: The type of encoding transformation that has been applied or can be applied to an **entity-body**. See [\[RFC2616\]](#) section 3.5 for details.

Content-Encoding: The **HTTP** header that defines the types of **content coding** that have been applied to the **HTTP entity-body**. See [\[RFC2616\]](#) section 14.11 for details.

entity-body: The name given to the payload of an **HTTP** request or response. See [\[RFC2616\]](#) section 1.3 for details.

FIN: The **TCP** control bit that signals no more data from sender. See [\[RFC793\]](#) section 3.2 for details.

PeerDist: The name of the new content encoding specified in this document.

PeerDist Content Encoding: A way of presenting an **HTTP** entity-body (defined in [\[RFC2616\]](#)) through its metadata, in the form of a Content Information Data Structure, as defined in [\[MS-PCCRC\]](#) section 2.3, which is derived from the content using algorithms described in [\[MS-PCCRC\]](#) sections [2.1](#) and [2.2](#).

PeerDist Content Information: A Message-body (defined in [\[RFC2616\]](#)) obtained for the requested content using **PeerDist Content Encoding**. Identifies the data produced by running one of the algorithms specified in [\[MS-PCCRC\]](#) section 2.3 on a given input.

PeerDist-encoded response entity body: See **PeerDist Content Information**.

PeerDist-Supporting Request: A request sent by a client that is able to participate in peer content caching and retrieval. A PeerDist-Supporting Request will have its Accept-Encoding header value set to PeerDist.

RST: The **TCP** control bit that signals a connection reset. See [\[RFC793\]](#) section 3.2 for details.

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

1.2 References

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

1.2.1 Normative References

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

[MS-PCCRC] Microsoft Corporation, "[Peer Content Caching and Retrieval: Content Identification](#)".

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981, <http://www.ietf.org/rfc/rfc0793.txt>

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

[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>

1.2.2 Informative References

[MC-BUP] Microsoft Corporation, "[Background Intelligent Transfer Service \(BITS\) Upload Protocol Specification](#)".

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

1.3 Overview

HTTP/1.1 is a client/server-based protocol. The purpose of PeerDist content encoding is to enable peer content caching and retrieval in HTTP/1.1.

Using PeerDist content encoding, allows an **HTTP/1.1 client** to participate in the peer content caching and retrieval process. Upon detecting PeerDist encoding support from a client, a **HTTP/1.1 server** that supports peer content caching may choose to send a PeerDist-encoded response. The message body (that is, an encoded entity body) of such a response takes the form of the [Content Information Data Structure](#) as specified in [MS-PCCRC] section 2.3, constructed for the requested content using the algorithms described in [MS-PCCRC] sections [2.1](#) and [2.2](#). Receiving a PeerDist-encoded response allows an HTTP/1.1 client to use the information present in the response to discover and download actual content from peers.

A typical PeerDist-encoded response is orders of magnitude smaller than a response that is not PeerDist encoded; the actual content transfer occurs between peers. Thus, PeerDist content encoding can reduce the burden of distributing the content from the HTTP/1.1 server.

A sequence diagram describing the communication between an HTTP/1.1 client and the HTTP/1.1 server is shown here.

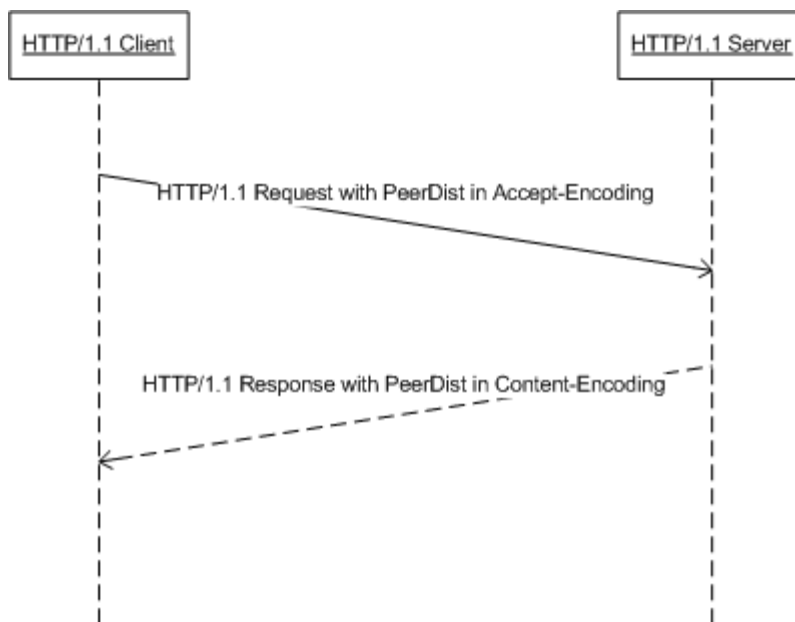


Figure 1: Sequence diagram describing the communication between the HTTP/1.1 client and the HTTP/1.1 server

1.4 Relationship to Other Protocols

The PeerDist content encoding defined in this document is intended to be used for HTTP/1.1.

The PeerDist content encoding is used by clients and servers that are capable of participating in peer content caching and retrieval.

The PeerDist content encoding uses the [Content Information](#) data structure defined in [\[MS-PCCRC\]](#) section 2.3.

1.5 Prerequisites/Preconditions

None.

1.6 Applicability Statement

Advertising PeerDist content encoding capability is applicable for an HTTP/1.0 client or HTTP/1.1 client (only) if it is able to participate in peer content caching and retrieval. [<1>](#)

Using PeerDist content encoding is applicable for an HTTP/1.1 server (only) when communicating to an HTTP/1.1 client that has advertised its capability to participate in peer content caching and retrieval.

1.7 Versioning and Capability Negotiation

The PeerDist content encoding defined in this document uses a version parameter that the HTTP/1.1 client sets to specify the maximum version of PeerDist content encoding that the client supports. [<2>](#)

The PeerDist content encoding defined in this document uses a version parameter that the HTTP/1.1 server sets to specify the version of PeerDist content encoding that is used for the HTTP response. [<3>](#)

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

This document defines PeerDist, a new content encoding that can be used in HTTP/1.1. HTTP/1.1 is the primary transport for all messages used by the PeerDist content encoding.

2.2 Message Syntax

HTTP/1.1 [RFC2616](#) defines the syntax of HTTP/1.1 messages.

This document defines a new content encoding value, namely PeerDist. The PeerDist content-encoding value can be specified in the **Accept-Encoding** and **Content-Encoding** header fields, as shown in the following examples.

```
Accept-Encoding: gzip, deflate, peerdist
Content-Encoding: peerdist
```

In addition, this document also defines a new extension-header field value. The syntax of this header field value is described as follows.

```
extension-header = X-P2P-PeerDist
X-P2P-PeerDist = "X-P2P-PeerDist" ":" peerdist-params
```

The extension-header field can appear in both requests and responses. The purpose of this header field is to carry additional parameters when the PeerDist content encoding is used.

```
peerdist-params = 1#( version | [content-len] | [missing-data-request] )
version = "Version" "=" major-version "." minor-version
major-version = 1*DIGIT
minor-version = 1*DIGIT
```

Note that there can be no spaces between major-version and "." as well as "." and minor-version. The major and minor versions MUST be considered as separate multidigit numbers. Thus, version 1.23 is higher than version 1.3.

The *version* parameter is used by HTTP/1.1 clients to specify the maximum version of PeerDist content encoding that the client supports. The *version* parameter is used by the HTTP/1.1 server to specify the version of PeerDist content encoding that was used for the response.

```
content-len = "ContentLength" "=" 1*DIGIT
```

The *content-len* parameter contains the length of the **entity-body**, defined in [RFC2616](#) section 1.3, in octets, before the PeerDist content encoding is applied to it.

The *missing-data-request* parameter is used by HTTP/1.1 clients to indicate to the server that the client is sending the request because it was unable to retrieve data from its peers. This parameter MUST NOT be specified when the PeerDist content encoding is specified in the Accept-Encoding header field value.

```
missing-data-request = "MissingDataRequest" "=" ( "true" | "false" )
```

3 Protocol Details

3.1 HTTP/1.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

An HTTP/1.1 client MAY [<4>](#) include the PeerDist content encoding in its Accept-Encoding header field value for every HTTP request it sends, as shown in the following example.

```
Accept-Encoding: gzip, deflate, peerdist
```

If the client chooses to use the PeerDist content encoding for an HTTP request, the client MUST also include the PeerDist parameters header field in the same HTTP request. As shown in the following example, the PeerDist parameters header field MUST contain the *Version* parameter containing the highest version of the PeerDist content encoding that the client supports.

```
X-P2P-PeerDist: Version=1.0
```

An HTTP/1.0 client MAY [<5>](#) include the PeerDist content encoding in the Accept-Encoding header field value of its HTTP requests.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.5.1 Receiving a Response of a PeerDist-Supporting Request

When an HTTP/1.1 client sends an HTTP request with the PeerDist content encoding listed in its Accept-Encoding header, the HTTP/1.1 server MAY send an HTTP response with a "Connection" header field with a value of "close". When an HTTP/1.1 client receives such a response, it SHOULD close the underlying **TCP** connection gracefully by sending a **FIN** instead of an **RST**. [<6>](#)

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 HTTP/1.1 Server Details

The HTTP/1.1 server MAY [\[7\]](#) use the PeerDist content encoding if the HTTP/1.1 request indicates that the client supports it. [\[RFC2616\]](#) section 14.11 defines content encoding usage.

The HTTP/1.1 server MAY [\[8\]](#) use the PeerDist content encoding in its response to an HTTP/1.0 request if the HTTP/1.0 request includes an Accept-Encoding header field containing PeerDist.

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 constructs, for the requested content, a [Content Information Data Structure](#) defined in [\[MS-PCCRC\]](#) section 2.3 using the algorithms described in [\[MS-PCCRC\]](#) sections [2.1](#) and [2.2](#) and places such a structure in the response message as an encoded entity body.

3.2.5.1 Receiving a PeerDist-Supporting Request

The HTTP/1.1 server MAY choose the PeerDist content encoding or any other client-supported encoding, as defined in [\[RFC2616\]](#), for its HTTP response. If the HTTP/1.1 server uses the PeerDist content encoding for its response, then the server MUST construct for the requested content, a [Content Information Data Structure](#) as specified in [\[MS-PCCRC\]](#) section 2.3, using the algorithms described in [\[MS-PCCRC\]](#) sections [2.1](#) and [2.2](#), and place such a structure in the response message as an encoded entity-body. It MUST also include the PeerDist parameters header field in the response. The PeerDist parameters header field MUST contain the *Version* parameter containing the version of the PeerDist content encoding used in the response. As shown in the following example, the PeerDist parameters header field MUST also contain the *ContentLength* parameter specifying the content length of the response entity-body before the PeerDist content encoding has been applied to it.

```
Content-Encoding: PeerDist
X-P2P-PeerDist: Version=1.0, ContentLength=102400
```

If the HTTP/1.1 server sends a PeerDist-encoded response entity-body, it MUST encode the entity-body into segments and blocks as specified in [\[MS-PCCRC\]](#) section 2, and then use that encoding to construct a Content Information Data Structure, as specified in [\[MS-PCCRC\]](#) section 2.3. It MUST then use this latter data structure as the PeerDist-encoded response entity-body.

The HTTP/1.1 server MAY [<9>](#) choose to use the algorithms and data structures defined in [MS-PCCRC] on the response entity-body before sending it to the HTTP/1.1 client. Furthermore, it MAY [<10>](#) send the "Connection" header field with a value of "close" to require the HTTP/1.1 client not to use the same connection for future HTTP requests. The HTTP/1.1 server SHOULD NOT [<11>](#) send the "Connection" header field in its response if the HTTP/1.1 client is known to be unable to handle the "Connection" header field gracefully, as specified in section [3.1.5.1](#).

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

When the HTTP client uses the PeerDist content encoding, it specifies PeerDist in the Accept-Encoding header field, as shown in the following example.

```
GET /index.html HTTP/1.1
Host: www.hostname.com
Accept: */*
Accept-Language: en-US
Accept-Encoding: gzip, deflate, peerdist
X-P2P-PeerDist: Version=1.0
User-Agent: Mozilla/4.0
```

In this example, the HTTP client announces that it is ready to accept the response entity-body that is encoded using the PeerDist content encoding. It also declares the version of the PeerDist content encoding for which it is configured.

If the server sends the HTTP response entity-body encoded with peerdist **content coding**, then it will set the Content-Encoding header field value to peerdist as shown in the following example.

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Encoding: peerdist
Content-Length: 198
Last-Modified: Fri, 01 Aug 2008 01:02:03 GMT
Accept-Ranges: bytes
ETag: "8d2babfc81f3c81"
Server: Microsoft-IIS/7.0
X-P2P-PeerDist: Version=1.0, ContentLength=184946
Date: Fri, 01 Aug 2008 10:20:30 GMT
...198 bytes of PeerDist Content Information...
```

In this response, the server indicates that the content is encoded using the PeerDist content encoding. The server used version 1.0 of the PeerDist content encoding. The server also includes the content length of the entity-body when it is encoded using the identity content coding. In other words, the **Content-Length** header field would have had the value 184946 if the Content-Encoding header was either missing or specified "identity" as defined in [\[RFC2616\]](#).

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:

- Windows Vista® operating system
- Windows Server® 2008 operating system
- Windows® 7 operating system
- Windows Server® 2008 R2 operating system

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 1.6:](#) For Windows Vista and Windows Server 2008, support for the client-side elements of this protocol is available only via the optional installation of the Background Intelligent Transfer Service (see [\[MC-BUP\]](#)) via Windows Management Framework. Support for the server-side elements of this protocol is not available for Windows Vista or Windows Server 2008.

[<2> Section 1.7:](#) HTTP/1.1 clients in Windows Vista, Windows Server 2008, Windows 7, and Windows Server 2008 R2 set the PeerDist version parameter to 1.0.

[<3> Section 1.7:](#) HTTP/1.1 servers in Windows Server 2008 R2 set the PeerDist version parameter to 1.0.

[<4> Section 3.1.4:](#) HTTP/1.1 clients in Windows Vista, Windows Server 2008, Windows 7, and Windows Server 2008 R2 use the PeerDist content encoding for GET requests only.

[<5> Section 3.1.4:](#) HTTP/1.0 clients in Windows Vista, Windows Server 2008, Windows 7, and Windows Server 2008 R2 use the PeerDist content encoding for GET requests only.

[<6> Section 3.1.5.1:](#) HTTP/1.1 clients in Windows Vista, Windows Server 2008, Windows 7, and Windows Server 2008 R2 disconnects gracefully by sending a FIN when they receive an HTTP response with "Connection" header field with a value of "close".

[<7> Section 3.2:](#) The HTTP/1.1 server in Windows Server 2008 R2 sends a PeerDist-encoded response only when the response contains an **ETag** header field or a **Last-Modified** header field or both header fields.

[<8> Section 3.2:](#) The HTTP/1.1 server in Windows Server 2008 R2 does not send a PeerDist-encoded response to an HTTP/1.0 request.

[<9> Section 3.2.5.1:](#) The HTTP/1.1 server in Windows Server 2008 R2 uses the algorithms and data structures defined in [\[MS-PCCRC\]](#) to generate the **PeerDist Content Information** only when it receives an HTTP/1.1 request. The server runs the algorithms asynchronously, and therefore it does not use the PeerDist encoding for the response to the request that triggered the execution of the algorithms. Similarly, the server does not use the PeerDist encoding for any HTTP/1.1 requests for the same content that are received during the execution of the algorithms on that content.

However, after the algorithms have completed and the PeerDist Content Information has been generated for that content, the server will respond to requests for the same content with the PeerDist Content Information for that content.

[<10> Section 3.2.5.1:](#) The HTTP/1.1 server in Windows Server 2008 R2 sends the "Connection" header field with a value of "close" if the HTTP request is a range retrieval request, and the total length of the full entity-body is greater than 1 megabyte.

[<11> Section 3.2.5.1:](#) The HTTP/1.1 server in Windows Server 2008 R2 does not send the "Connection" header field with a value of "close" if the HTTP/1.1 client is "Microsoft BITS".

7 Change Tracking

This section identifies changes that were made to the [MS-PCCRTP] protocol document between the May 2011 and June 2011 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

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

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

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

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

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

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

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

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

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

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.2 References	Added explanatory statement regarding the removal of the publishing year from Microsoft Open Specification document references.	N	Content updated.

8 Index

A

Abstract data model
 [client](#) 11
 [server](#) 12
[Applicability](#) 7

C

[Capability negotiation](#) 7
[Change tracking](#) 18
Client
 [abstract data model](#) 11
 [higher-layer triggered events](#) 11
 [initialization](#) 11
 [local events](#) 11
 message processing
 [overview](#) 11
 [PeerDist-Supporting request - receiving a response](#) 11
 sequencing rules
 [overview](#) 11
 [PeerDist-Supporting request - receiving a response](#) 11
 [timer events](#) 11
 [timers](#) 11

D

Data model - abstract
 [client](#) 11
 [server](#) 12

E

[Examples - overview](#) 14

F

[Fields - vendor-extensible](#) 8

G

[Glossary](#) 5

H

Higher-layer triggered events
 [client](#) 11
 [server](#) 12

I

[Implementer - security considerations](#) 15
[Index of security parameters](#) 15
Initialization
 [client](#) 11
 [server](#) 12

[Introduction](#) 5

L

Local events
 [client](#) 11
 [server](#) 13

M

Message processing
 client
 [overview](#) 11
 [PeerDist-Supporting request - receiving a response](#) 11
 server
 [overview](#) 12
 [PeerDist-supporting request - receiving](#) 12
Messages
 [syntax](#) 9
 [transport](#) 9

N

[Normative references](#) 6

O

[Overview \(synopsis\)](#) 6

P

[Parameters - security index](#) 15
[Preconditions](#) 7
[Prerequisites](#) 7
[Product behavior](#) 16

R

[References - normative](#) 6
[Relationship to other protocols](#) 7

S

Security
 [implementer considerations](#) 15
 [parameter index](#) 15
Sequencing rules
 client
 [overview](#) 11
 [PeerDist-Supporting request - receiving a response](#) 11
 server
 [overview](#) 12
 [PeerDist-supporting request - receiving](#) 12
Server
 [abstract data model](#) 12
 [higher-layer triggered events](#) 12

- [initialization](#) 12
- [local events](#) 13
- message processing
 - [overview](#) 12
 - [PeerDist-supporting request - receiving](#) 12
- [overview](#) 12
- sequencing rules
 - [overview](#) 12
 - [PeerDist-supporting request - receiving](#) 12
- [timer events](#) 13
- [timers](#) 12
- [Standards assignments](#) 8
- [Syntax](#) 9

T

- Timer events
 - [client](#) 11
 - [server](#) 13
- Timers
 - [client](#) 11
 - [server](#) 12
- [Tracking changes](#) 18
- [Transport](#) 9
- Triggered events - higher-layer
 - [client](#) 11
 - [server](#) 12

V

- [Vendor-extensible fields](#) 8
- [Versioning](#) 7