[MS-CCRSO]:
Content Caching and Retrieval System Overview

This document provides an overview of the Content Caching and Retrieval System Overview Protocol Family. It is intended for use in conjunction with the Microsoft Protocol Technical Documents,

publicly available standard specifications, network programming art, and Microsoft Windows distributed systems concepts. It assumes that the reader is either familiar with the aforementioned material or has immediate access to it.

A Protocol Family System Document does not require the use of Microsoft programming tools or programming environments in order to implement the Protocols in the System. Developers who have access to Microsoft programming tools and environments are free to take advantage of them.

Abstract

Describes the Windows Content Caching and Retrieval System. It includes a description of the protocols, data structures, and mechanisms (such as security) required to enable a system of content caching and retrieval to interoperate with Windows systems that use the feature known as BranchCache™.

Revision Summary

Date	Revision History	Revision Class	Comments
04/23/2010	0.1	Major	First Release.
06/04/2010	0.1.1	Editorial	Revised and edited the technical content.
07/16/2010	1.0	Major	Significantly changed the technical content.
08/27/2010	1.1	Minor	Clarified the meaning of the technical content.
10/08/2010	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
03/25/2011	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
05/06/2011	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
06/17/2011	1.2	Minor	Clarified the meaning of the technical content.

Contents

1 Introduction

This Protocol Family System Document (PFSD) is primarily intended to cover the Protocol Family as a whole. In conjunction with Member Protocol Technical Documents (TDs), which are intended to cover Member Protocols, it presents the rules for information exchange relevant to those Member Protocols and the Protocol Family that are used to interoperate or communicate with a Microsoft Windows® operating system in its various environments.

The Content Caching and Retrieval System supports content retrieval scenarios such as accessing content from a file or Web server. For file access scenarios, this document can be used in conjunction with the File Access Services System Overview [MS-FSSO]. [MS-FSSO] describes the protocols required for network File Access Services interoperation with Windows systems. This document describes the additional protocols, data structures, and mechanisms, such as security, that are required to enable a system of content caching and retrieval to interoperate with Windows systems. If for any reason the content caching and retrieval of data is unavailable or fails, normal file access would continue without caching using the SMB2.1 or HTTP protocols.

The member protocols of the Content Caching and Retrieval System describe content discovery, transport, data structures used for content, and the encryption process for securing content. Within the system, caching has two distinct modes of operation. One, where the content cache is on a single predetermined computer, known in this document as a "Hosted Cache" and is a client server model.<1> The other mode is where the cached content is distributed around a number of computers; this is known in this document as "distributed cache" and is a peer-to-peer caching model.<2>

Content within the system is divided up into **segments** and **blocks**, a block being a subdivision of a segment. It is segments and blocks that are stored and retrieved by the system, rather than files.

Content caching and retrieval requires at least three computers; one computer acting as a content server (normally located on a WAN link), one acting as a client requesting content, and a third (normally a computer on the same LAN as the requesting client) that holds in cache some or all of the content that the client computer is requesting.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

Server Message Block (SMB)
handle
FileId

The following terms are defined in [MS-FSSO]:

File Client
file server
SMB File Client
SMB File Service
SMB Access Protocols

The following terms are defined in [MS-PCCRC]:

block
content server
peer

segment
segment hash of data (HoD)

The following terms are defined in [MS-PCHC]:

hosted cache
HoHoDk

The following terms are defined in [MS-PCCRTP]:

PeerDist

The following terms are defined in [MS-PCCRR]:

client (client-role peer)
download schedule session
segment retrieval session
server (server-role peer)

The following terms are defined in [MS-SMB2]:

Tree Connect

The following terms are specific to this document:

BranchCache™: A Windows Content Caching and Retrieval feature that enables content from File and Web servers on a wide area network (WAN) to be cached on computers at a local branch office. Available in two modes: Hosted Cache and distributed cache.

content: When cached, content is identified by segment and downloaded in blocks.

content block: A block of data in the content that can be retrieved from clients.

Client-Side Caching (CSC): A local cache, also known as offline files, on a computer running Windows.

distributed cache: A cache composed of blocks of data hosted on multiple peers acting in cooperation.

File Access Protocol: A protocol that enables remote access to a portion of a local Object Store and supports file system semantics. Specifically in this document, this means the SMB2.1 access protocols and HTTP/HTTPS protocols.

file handle: A general term used to refer to SMB2_FILEID, it represents an open file on the server often referred to as File ID or file id. A **file handle** is returned from an SMB2 Open or SMB2 Create operation and is unique within an SMB2 connection.

hash: A hash (such as SHA-1) on the content or content block.

hash list: A list of hashes, comprising the block hashes plus the content hash.

HMAC: Keyed-Hashing for Message Authentication. For more information, see [RFC2104].

metadata: A generic term for a **hash** or **hash list**.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. Note that in [RFC2119] terms, most of these specifications should be

imperative, to ensure interoperability. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

Any specification that does not explicitly use one of these terms is mandatory, exactly as if it used MUST.

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.

[FIPS180-2] Federal Information Processing Standards Publication, "Secure Hash Standard", FIPS PUB 180-2, August 2002, http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf

[FIPS197] National Institute of Standards and Technology, "Federal Information Processing Standards Publication 197: Advanced Encryption Standard (AES)", November 2001, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

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

[MS-ADSO] Microsoft Corporation, "Active Directory System Overview".

[MS-AUTHSO] Microsoft Corporation, "Windows Authentication Services System Overview".

[MS-CASO] Microsoft Corporation, "Certification Authority System Overview".

[MS-DISO] Microsoft Corporation, "Domain Interactions System Overview".

[MS-FSSO] Microsoft Corporation, "File Access Services System Overview".

[MS-GPSO] Microsoft Corporation, "Group Policy System Overview".

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

[MS-PCCRD] Microsoft Corporation, "Peer Content Caching and Retrieval Discovery Protocol Specification".

[MS-PCCRR] Microsoft Corporation, "Peer Content Caching and Retrieval: Retrieval Protocol Specification".

[MS-PCCRTP] Microsoft Corporation, "Peer Content Caching and Retrieval: Hypertext Transfer Protocol (HTTP) Extensions".

[MS-PCHC] Microsoft Corporation, "Peer Content Caching and Retrieval: Hosted Cache Protocol Specification".

[MS-SMB2] Microsoft Corporation, "Server Message Block (SMB) Version 2 Protocol Specification".

[MS-TLSP] Microsoft Corporation, "Transport Layer Security (TLS) Profile".

[RFC768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980, http://www.ietf.org/rfc/rfc768.txt

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

[RFC1001] Network Working Group, "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concepts and Methods", STD 19, RFC 1001, March 1987, http://www.ietf.org/rfc/rfc1001.txt

[RFC1002] Network Working Group, "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications", STD 19, RFC 1002, March 1987, http://www.ietf.org/rfc/rfc1002.txt

[RFC1034] Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987, http://www.ietf.org/rfc/rfc1034.txt

[RFC1035] Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987, http://www.ietf.org/rfc/rfc1035.txt

[RFC2104] Krawczyk, H., Bellare, M., and Canetti, R., "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997, http://www.ietf.org/rfc/rfc2104.txt

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

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

[RFC2743] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, January 2000, http://www.ietf.org/rfc/rfc2743.txt

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

[RFC4559] Jaganathan, K., Zhu, L., and Brezak, J., "SPNEGO-based Kerberos and NTLM HTTP Authentication in Microsoft Windows", RFC 4559, June 2006, http://www.ietf.org/rfc/rfc4559.txt

[SOAP/UDP] H. Combs, et al, "SOAP-over-UDP," September 2004. (See http://schemas.xmlsoap.org/ws/2004/09/soap-over-udp)

[WS-Discovery] Beatty, J., Kakivaya, G., Kemp D., et al., "Web Services Dynamic Discovery (WS-Discovery)", April 2005, http://specs.xmlsoap.org/ws/2005/04/discovery/ws-discovery.pdf

If you have any trouble finding [WS-Discovery], please check here.

1.2.2 Informative References

[MS-BPCR] Microsoft Corporation, "Background Intelligent Transfer Service (BITS) Peer-Caching: Content Retrieval Protocol Specification".

[MS-BPDP] Microsoft Corporation, "Background Intelligent Transfer Service (BITS) Peer-Caching: Peer Discovery Protocol Specification".

[MS-FSA] Microsoft Corporation, "File System Algorithms".

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

[MS-WSO] Microsoft Corporation, "Windows System Overview".

[MS-FSCC] Microsoft Corporation, "File System Control Codes".

[MSDN-PeerDISTAPI] Microsoft Corporation, "About Peer Distribution", October 2009, http://msdn.microsoft.com/en-us/library/dd407951(VS.85).aspx

[MSDN-MS-WINSRA] Microsoft Corporation, "Windows Internet Naming Service (WINS) Replication and Autodiscovery Protocol Specification" July 2007.

[MSDN-SOAPoverUDP] Combs, H., Justin, J., Kakivaya, Gl., et al., "SOAP-over-UDP", 2004, http://msdn.microsoft.com/en-us/library/ms951224.aspx

[MSFT-BranchCache] Microsoft Corporation, "BranchCache", http://technet.microsoft.com/en-us/network/dd425028.aspx

If you have any trouble finding [MSFT-BranchCache], please check here.

[MSFT-DNS] Microsoft Corporation, "DNS", updated January 2005, http://technet2.microsoft.com/windowsserver/en/library/66add8fa-0348-4cc4-94d1-6d68127290881033.mspx?mfr=true

[RFC4795] Aboba, B., Thaler, D., and Esibov, L., "Link-Local Multicast Name Resolution (LLMNR)", RFC 4795, January 2007, http://www.ietf.org/rfc/rfc4795.txt

2 Overview

Section 1, "Introduction", describes this Protocol Family System Document. This section introduces the system that is being documented.

2.1 System Summary

The Content Caching and Retrieval System consists of a set of protocols that collectively enable a content caching system. The purpose of this system is to optimize network link utilization by reducing the repeated use of slow network links in favor of faster network links. This is achieved by caching retrieved content on machines that have fast network links. The system is based on a peer-to-peer discovery and distribution model.

The member protocols each address a different aspect of the content caching system:

- Content identification as specified in [MS-PCCRC]
- Content discovery as specified in [MS-PCCRD]
- Content retrieval as specified in [MS-PCCRR]
- Content offering as specified in [MS-PCHC]

Content clients do not actively participate in the Content Caching and Retrieval System until they have obtained metadata for Content that they require. That metadata can be obtained using either SMB 2.1 or HTTP with **PeerDist** encoding. The format of the metadata is specified in [MS-PCCRC]. It should be noted that both SMB2.1 and HTTP in respect to the metadata are merely acting as a transport mechanism. These two protocols are therefore not considered core participants in the system.

2.2 List of Member Protocols

The Content Caching and Retrieval System contains the following member protocols. (See section 5.3.1, Member Protocol Roles for more detail, and section 5.3.2 for information on protocol groupings.)

Peer Content Caching and Retrieval: Content Identification, as defined in [MS-PCCRC], specifies a binary data structure used in the Content Caching and Retrieval. The primary role in the Content Caching and Retrieval System is Content Identification.

Peer Content Caching and Retrieval Discovery Protocol, as defined in [MS-PCCRD], specifies a multicast to discover and locate services based on the Web Services Dynamic Discovery (WS-Discovery) protocol [WS-Discovery]. There are two modes of operations in WS-Discovery: client-initiated probes and service-initiated announcements; both are sent through IP multicast to a predefined group. The primary role in the Content Caching and Retrieval System is Content Discovery.

Peer Content Caching and Retrieval: Retrieval Protocol, as defined in [MS-PCCRR], specifies the messages that are necessary for querying Peer-role servers or a Hosted cache server for the availability of certain content, and for retrieving the content. The primary role in the Content Caching and Retrieval System is Content Retrieval.

Peer Content Caching and Retrieval: Hosted Cache Protocol, as defined in [MS-PCHC] specifies an HTTPS-based mechanism for clients to notify a hosted cache server regarding the availability of content and for a hosted cache server to indicate interest in the content. The primary role in the Content Caching and Retrieval System is Content Notification.

Peer Content Caching and Retrieval: Hypertext Transfer Protocol (HTTP) Extensions, as defined in [MS-PCCRTP], specifies a content encoding known as PeerDist that is used by an HTTP/1.1 client and an HTTP/1.1 server to communicate content to each other. The primary role in the Content Caching and Retrieval System is metadata (hash) retrieval.

Server Message Block (SMB) Version 2.1 Protocol, as defined in [MS-SMB2]. Version 2.1 of this protocol has enhancements for the detection of content caching-enabled shares and retrieval of metadata related to content caching. The primary role in the Content Caching and Retrieval System is metadata (hash) retrieval.

2.3 Relevant Standards

Advanced Encryption Standard as specified in [FIPS197]

Hypertext Transfer Protocol - HTTP/1.1 as specified in [RFC2616]

Hypertext Transfer Protocol - HTTP/1.1 over TLS as specified in [RFC2818]

Protocol Standard for a NetBIOS Service on a TCP/UDP Transport (NetBIOS over TCP), as specified in [RFC1001] and [RFC1002].

Secure Hash Standard as specified in [FIPS180-2]

Transmission Control Protocol (TCP), as specified in [RFC793]

User Datagram Protocol (UDP), as specified in [RFC768]

Web Services Dynamic Discovery as specified in [WS-Discovery]

3 Foundation

This section describes the theoretical and practical information needed to understand this document and this system.

3.1 Background Knowledge and System-Specific Concepts

This section summarizes:

Background knowledge required to understand this document.

Concepts specific to this system.

3.1.1 File Access Services

The Content Caching and Retrieval system is supplementary to the Windows File Access Services System. Details of the requirements of the File Access Services System on domains and workgroups can be found in [MS-FSSO]. A summary of available system documents can be found in the Windows System Overview [MS-WSO]. If content caching is unavailable, then file retrieval will work as described without caching. In effect the absence of caching will not be noticeable to the user other than by its effect on performance.

3.1.2 Cache

A file cache is a collection of files duplicating original content stored elsewhere; usually the original data is expensive to fetch owing to longer access time compared to the cost of reading the cache. In other words, a cache is a temporary storage area where frequently accessed data can be stored for more rapid access. Once the data is stored in the cache, it can be used in the future by accessing the cached copy rather than re-fetching the original data.

3.1.3 Content Caching

Content Caching and Retrieval is an aid to bandwidth management for wide area networks (WAN). The goal is to increase network performance and hence response times for end users. There are many potential techniques available; however, the most efficient way to accelerate the transfer of information across a WAN is to send less data in the first place. This is the major principle employed by content caching and retrieval, namely the reduction of traffic across a WAN. Content caching enables content from file and Web servers on one end of a WAN to be cached on computers at the other end of the WAN. The content can be cached in one of two ways; distributed across multiple computers, known as Distributed Cache Mode or centrally hosted on a single server, known as Hosted Cache Mode.

Content caching enables content retrieved by using SMB2.1 or HTTP on the local end of a WAN to be cached on computers at the remote end of the WAN (often a branch office). The cached content can be distributed across client computers (a distributed cache) or centrally hosted on a single server (a hosted cache). Cached content functions on segments and blocks rather than on files.

For the purposes of the Content Caching and Retrieval System, content is considered to be divided into one or more segments. Segments are the unit of discovery. Each segment is divided in turn into blocks. Blocks are the unit of download.

The caching mechanism is designed to prevent unauthorized modification of cached content. The content is encrypted when it is transferred between distributed caches or between clients and a Hosted Cache server. Content can only be decrypted using the identifiers provided by the original

content server. That content server will only provide identifiers to authorized clients so authentication and access security is maintained.

A key mechanism behind the Content Caching and Retrieval System is the provision of metadata. Files or streams located on a content server have a list of cryptographically descriptive hashes created that uniquely identifies the data. Instead of receiving original content, a content client receives the hashed representation of the content (metadata). The content client then seeks to reassemble the content using individual hashes to identify identical segments and requests the content from one or more caches located on the local area network. Any content that cannot be discovered locally is considered "missing data" and can be retrieved from the content server over the WAN.

The Content Caching and Retrieval System has the following logical components:

Content client (content clients may act as both a **Client-role Peer** and a **Server-role Peer**).

The content client performs the following tasks:

- Decorates outgoing HTTP requests with a content caching header (PeerDist).
- Retrieves **hash lists** (in the case of HTTP).
- Retrieves content from the local cache, when available, using hash lists.
- Requests ranges of data (segments and blocks) from peers.
- Adds retrieved data to the distributed cache.
- Adds retrieved data to the local cache .
- Verifies data in the cache using hash lists.
- If content is not available (cached or direct), returns an error to the application.

Content Server:

The content server performs the following tasks:

- Receives incoming HTTP requests decorated with content caching header (PeerDist).
- Performs access checks for the user.
- Generates hash lists for content identified as being available for content caching.
- Returns hash lists (metadata) to a requesting content client using SMB2.1 or HTTP.
- Returns content to a requesting content client.
- In the case of access violations or content retrieval failures (missing original content) returns an error to the content client.

3.1.4 Windows BranchCache(tm)

BranchCache™ is the name given to the Windows Content Caching and Retrieval System [MSFT-BranchCache].

The Windows File Access Services System [MS-FSSO] uses the Content Caching and Retrieval System for two forms of transport:

- HTTP/HTTPS: With this transport the content server only calculates hashes "on demand" the first time content is sent to a client. The original content stream is sent to clients until such time as hashes have been calculated. At that time future client requests will receive content hashes and attempt peer retrieval.
- SMB Version 2.1, as specified in [MS-SMB2]: Enables an encrypted hash list of the content to be retrieved from a **file server**. Applications collocated with the **SMB File Client** can then attempt to retrieve the actual content from a peer cache or hosted cache server rather than the file server. Content clients can also publish content retrieved through the SMB File Client into their own distributed cache or to a hosted cache server.

3.1.5 Additional Network Infrastructure

The Content Caching and Retrieval System relies upon network infrastructure to provide address resolution and a variety of transports for its protocols. Address resolution from a name to an IP address can be provided by a variety of mechanisms, but the most common are DNS [RFC1034] and [RFC1035], Link-Local Multicast Name Resolution (LLNMR) [RFC4795] for DNS names, and **Windows Internet Name Service (WINS)** [MS-WINSRA] for NetBIOS names. For a general overview of Windows DNS, see [MSFT-DNS].

The System uses [RFC2616] and [RFC2818] as a transport for cached content.

3.2 System Purposes

The goal of the Content Caching and Retrieval System is to decrease WAN network use. This is accomplished by caching content that has been retrieved over a WAN link (or any high latency link) from a content server by a set of actors (computers, applications, or people) connected to a local area network (LAN) and making it available for subsequent use within the LAN environment in a secure and effective manner. The overall effect is to reduce WAN traffic and therefore increase application performance.

3.3 System Use Cases

3.3.1 Stakeholders and Interests Summary

The stakeholders and their associated interests for the Content Caching and Retrieval System are as follows:

User: The User is the principal that requires file access in order to read files on another computer. The User is referred to using the qualifiers "SMB2.1", "HTTP", or "BITS" when it is necessary to distinguish User instances. The User is external to the File Services System and the Content Caching and Retrieval System, and interacts through the Application. The Content Caching and Retrieval System only applies to the reading of existing files.

Administrator: The Administrator is the person who administers the content server and hosted cache server. The Administrator is interested in organizing content, setting access rights, and enabling content caching. The Administrator is external to the Content Caching and Retrieval System, and interacts with the System through the Administrator Tool.

Administrative Tool: The Administrator Tool is a program that offers management functionality to the Administrator by means of the Admin Client. Typical Administrator Tools are command line and graphical shells, management utilities, and graphical management programs. The Administrator Tool is external to the Content Caching and Retrieval System and makes use of the Admin Client to accomplish its work.

Administrative Client: The Admin Client is a program that allows configuration of the Content Caching and Retrieval System. The Admin Client is internal to the Content Caching and Retrieval System.

Application: The Application is a program that consumes file reading services by means of the content client. Applications (where caching applies) need to open, read, and close files. The Application is external to the File Services System and the Content Caching and Retrieval System. The Application interacts with System through the content client.

Content Client: The content client implements client-side protocol components and consumes the file services that are offered by the content server. The content client can be referred to using the qualifier "SMB2.1", "HTTP", or "BITS" when it is necessary to distinguish Client instances. The content client is internal to the Content Caching and Retrieval System. A content client may additionally act as Distributed cache peer.

Content Server: The content server's interest is to provide and maintain a secure and consistent File Access Service (see [MS-FSSO]) and to provide content metadata as part of the Content Caching and Retrieval System.

Hosted Cache Server: The hosted cache server's interest is to cache content and to receive metadata regarding the availability of content segments and blocks, and then as required to download the segments and blocks from clients that have the relevant data. Later, when another client requests the content through a secure mechanism, the content can be retrieved from the Hosted Cache rather than from a content server.

Distributed Cache Peer: A Distributed cache peer's interest is to cache and distribute data and respond to queries regarding the availability of data segments and blocks. Then when a client requests the content through a secure mechanism, the content can be obtained from the distributed cache rather than from a content server.

Object Store: The File Access Services System (and therefore the Content Caching and Retrieval System) is dependent on an external Object Store for storing files and directories. In Windows, the Object Store is provided by a local file system, usually NTFS. Some of the wire-visible behavior of File Access Services protocols is not specified by the protocols themselves, and is dependent on Object Store behavior.

3.3.2 Supporting Actors and System Interests Summary

File Access Services System [MS-FSSO]: The purpose of the File Access Services System is to allow a set of actors (people or processes) to access and share files located on a file server, using a network between them, in a secure and managed environment. The files may be distributed among a number of computers in a workgroup or domain, or centralized in one or more file server computers. The File Access Services System can optionally use the Content Caching and Retrieval System to cache content.

Group Policy System [MS-GPSO]: The Group Policy System enables an administrator to maintain standard operating environments in domains for specific groups of users and computers. As software changes and policies change over time, Group Policy can be used to update an already-deployed standard operating environment. Computers participating in the Content Caching and Retrieval System within a domain can be expected to participate in, and be influenced by, the Group Policy System.

Certification Authority System [MS-CASO]: The Certification Authority System enables an administrator to request a certificate.

Background Intelligent Transfer Service (BITS) Upload Protocol, as defined in [MC-BUP], specifies an HTTP 1.1-based upload protocol. This protocol is used to transfer large payloads from a client to a server or server to client over networks with frequent disconnections, and to send notifications about the availability of uploaded payloads. "BITS" is a PeerDist-enabled HTTP client and can therefore act as a client of the Content Caching and Retrieval System.

3.3.3 Use Case Diagrams

The following table groups use cases that span the functionality of the Content Caching and Retrieval System. Detailed descriptions for these use cases are provided in section 3.3.4.

Use case group	Use cases
Configuring Content Caching and Retrieval Components	Configure content server (SMB2.1) caching Configure content server (HTTP) caching Configure content client caching mode Configure a hosted cache server
Reading Content	Read a file using SMB2.1 with hosted cache (Cached Data Unavailable) Read a file using SMB2.1 with hosted cache (Cached Data Available) Read a file using SMB2.1 with distributed cache (Cached Data Unavailable) Read a file using SMB2.1 with distributed cache (Cached Data Available) Read a file using HTTP with Hosted Cache (Cached Data Unavailable) Read a file using HTTP with distributed cache (Cached Data Available) Read a file using BITS with Hosted Cache (Cached Data Unavailable) Read a file using BITS with distributed cache (Cached Data Available) **Note** The initial transport (SMB2.1, HTTP, or BITS) protocol has little impact on the overall caching behavior; therefore not all combinations are shown.

The following use case diagrams illustrate the separate groups of use cases described in this section.

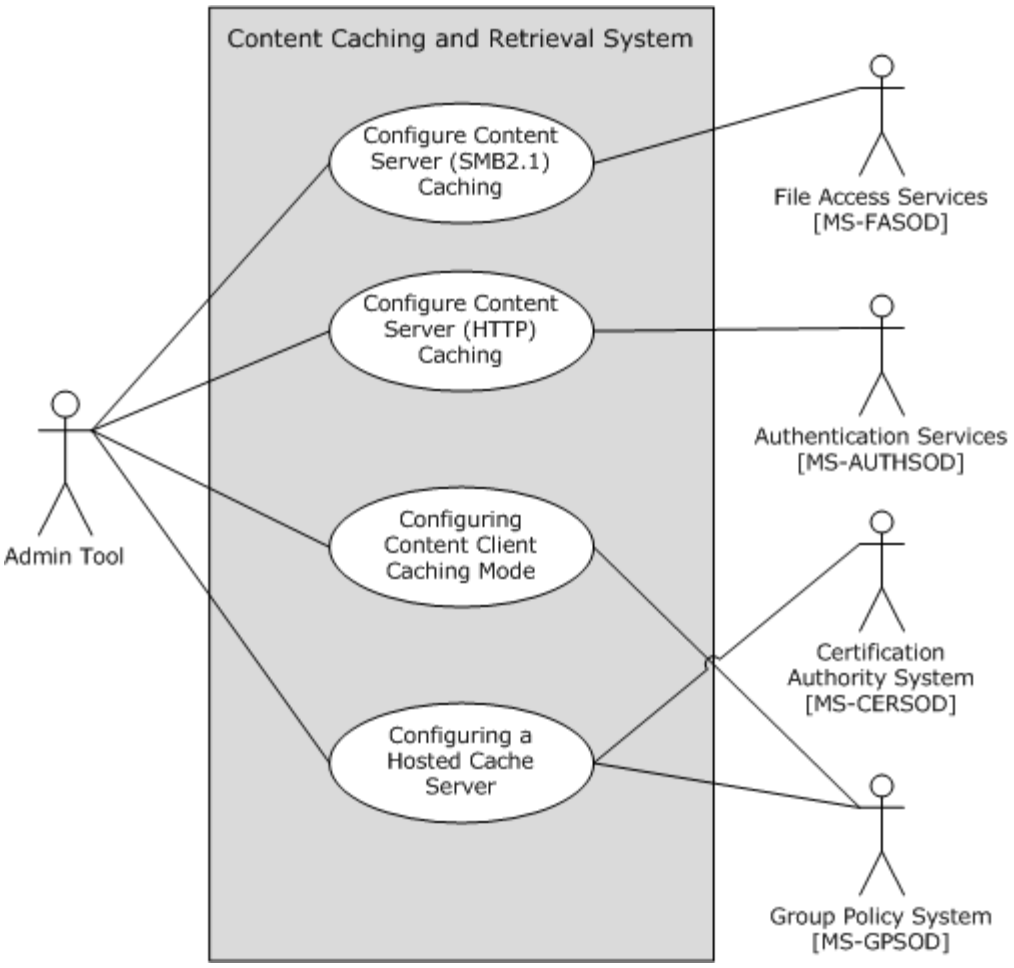

Figure 1: Use cases included in the Configuring Content Caching and Retrieval Components summary use case

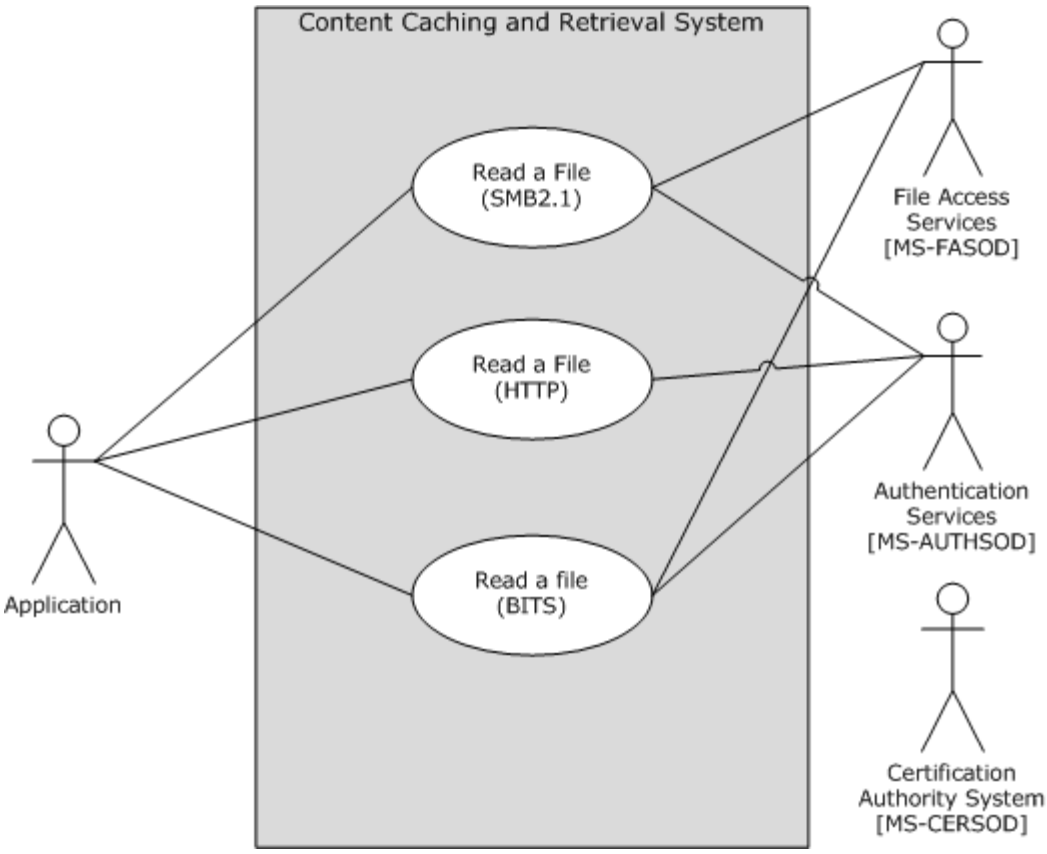

Figure 2: Use cases included in the Reading Content summary use case

3.3.4 Summary Use Case Descriptions

Note that the configuration tools on Windows are local only and therefore do not use protocols.

3.3.4.1 Configure Content Server (SMB2.1) Caching

3.3.4.1.1 Stakeholders and Interests Summary

Goal: To enable Content Caching and Retrieval on a content server share directory.

Context of Use: The Administrator is configuring a content server and is either adding a shared directory or modifying a shared directory.

Direct Actor: The direct actor is the Administrator Tool. The Administrator Tool's interest is to correctly interpret, execute, and display the results of the commands issued by the Administrator.

Primary Actor: The primary actor is the Administrator. The Administrator's interest is expressed in administrative privileges and responsibility for using the File Access Services System and the Content Caching and Retrieval System to provide File Services.

Supporting Actors: The supporting actors for this use case are as follows:

- Administrator Client: Maintains a consistent access mechanism to the SMB 2.1 File Service and Content Caching and Retrieval configuration.
- SMB File Service: Provides and maintains a secure and consistent file service by enforcing directory quotas and file screens.
- Object Store: Stores files and directories.

Stakeholders and Interests: There are no additional Stakeholders.

Preconditions: The Administrator has identified a content server and a shared directory on its Object Store, and wishes to enable Content Caching and Retrieval on that shared directory.

Minimal Guarantees: No action is taken that affects other directories or shares on the content server.

Success Guarantee: Content Caching and Retrieval is enabled on the shared directory identified on the content server.

Trigger: The Administrator Tool receives a request from the Administrator to configure Content Caching and Retrieval.

3.3.4.1.2 Main Success Scenario

1. The Administrator Tool (Share and Storage Management Console) creates or modifies an SMB2.1 share with the specified SMB2.1 share name and directory on the content server.
2. The Administrator Tool enables Content Caching and Retrieval on the SMB2.1 share.

3.3.4.2 Configure Content Server (HTTP) Caching

3.3.4.2.1 Stakeholders and Interests Summary

Goal: To enable Content Caching and Retrieval on a content server-hosted Web site.

Context of Use: The Administrator is configuring an HTTP Server and is either adding a Web site or modifying an existing site.

Direct Actor: The direct actor is the Administrator Tool (netsh). The Administrator Tool's interest is to correctly interpret, execute, and display the results of the commands issued by the Administrator.

Primary Actor: The primary actor is the Administrator. The Administrator's interest is expressed in administrative privileges and responsibility for using the Web site.

Supporting Actors: The supporting actors for this use case are as follows:

- Admin Client: Maintains a consistent access mechanism to the HTTP server and Content Caching and Retrieval configuration.
- HTTP Server: Provides and maintains a secure and consistent file service.
- Object Store: Stores files and directories.

Stakeholders and Interests: There are no additional Stakeholders.

Preconditions: The Administrator has identified a content server that is hosting a Web site on its Object Store, and wishes to enable Content Caching and Retrieval on that site.

Minimal Guarantees: No action is taken that affects other directories or shares on the content server.

Success Guarantee: Content Caching and Retrieval is enabled on the Web site identified on the content server.

Trigger: The Administrator Tool receives a request from the Administrator to configure a Web site.

3.3.4.2.2 Main Success Scenario

- The Administrator Tool creates or modifies a Web site hosted on the content server.

3.3.4.3 Configure a Content Client Caching Mode

3.3.4.3.1 Stakeholders and Interests Summary

Goal: To configure content caching mode on a client peer.

Context of Use: The Administrator is configuring a content client for either hosted cache or distributed cache mode. These are mutually exclusive roles and depend on the Abstract Data Model (ADM) element Server Role.

Direct Actor: The direct actor is the Administrator Tool (netsh). The Administrator Tool's interest is to correctly interpret, execute, and display the results of the commands issued by the Administrator.

Primary Actor: The primary actor is the Administrator. The Administrator's interest is expressed in administrative privileges and responsibility for the content client using the Content Caching and Retrieval System.

Supporting Actors: The supporting actors for this use case are as follows:

- Admin Client: Maintains a consistent access mechanism to the content client and the Content Caching and Retrieval configuration of the client.

Preconditions: The Administrator has identified a client peer (content client) and wishes to configure the caching mode. The client peer is Content Caching and Retrieval capable.

Minimal Guarantees: No action is taken that affects systems on the client peer.

Success Guarantee: The caching mode is set to one of: DISABLED, LOCAL, DISTRIBUTED, or HOSTED CLIENT.

Trigger: The Administrator Tool receives a request from the Administrator to configure the caching mode.

3.3.4.3.2 Main Success Scenario

1. The Administrator Tool establishes a connection the Content Caching and Retrieval System configuration.
2. The Administrator Tool changes context to Caching and Retrieval.
3. The Administrator Tool sets the service mode (see section 5.1) to HOSTEDCLIENT or DISTRIBUTED.
4. The firewall allows DISTRIBUTED or HOSTEDCLIENT mode protocols.

3.3.4.4 Configure a Hosted Cache Server

3.3.4.4.1 Stakeholders and Interests Summary

Goal: To configure content caching mode on a server.

Context of Use: The Administrator is configuring a hosted cache server.

Direct Actor: The direct actor is the Administrator Tool. The Administrator Tool's interest is to correctly interpret, execute, and display the results of the commands issued by the Administrator.

Primary Actor: The primary actor is the Administrator. The Administrator's interest is expressed in administrative privileges and responsibility for the hosted cache server using the Content Caching and Retrieval System.

Supporting Actors: The supporting actors for this use case are as follows:

- Admin Client: Maintains a consistent access mechanism to the hosted cache server and the Content Caching and Retrieval configuration.

Preconditions: The Administrator has identified a server and wishes to configure it as a hosted cache server. The identified server is Content Caching and Retrieval-capable. The server has a fully qualified domain name (FQDN) and if Client Authentication is to be used is a member of a domain.

Minimal Guarantees: No action is taken that affects systems on the server.

Success Guarantee: The caching mode is set to HOSTEDSERVER; Client Authentication is set to one of DOMAIN or NONE.

Trigger: The Administrator Tool receives a request from the Administrator to configure the hosted cache server.

3.3.4.4.2 Main Success Scenario

1. The Administrator Tool establishes a connection with the Content Caching and Retrieval System configuration of the Server.
2. The Administrator Tool changes context to Caching and Retrieval.
3. The Administrator Tool sets the service mode to HOSTEDSERVER; Client Authentication is set to one of DOMAIN or NONE.
4. The firewall allows HOSTEDSERVER protocols.

3.3.4.5 Reading and Caching a File from a Content Server

3.3.4.5.1 Stakeholders and Interests Summary

Goal: To read content from a content server with Caching and Retrieval in effect.

Context of Use: A number of users are attempting to read content that resides on a slow link. When some or all of the content is transferred to the LAN, then content caching and retrieval will operate to improve the performance of the File Access Service for later users.

Direct Actor: The direct actor is the Application. The Application's interest is to consume file reading services by means of the content client and to correctly execute and display the results of commands issued by a user.

Primary Actor: The primary actor is the User. The User's interest is to access content on a content server by using the File Access Services System and the Content Caching and Retrieval System to provide File Services.

Supporting Actors: The supporting actors for this use case are as follows:

- Content Client: Maintains a consistent access mechanism to the SMB 2.1 File Service and Content Caching and Retrieval System.
- SMB2.1 File Server: Provides and maintains a secure and consistent file service by enforcing directory quotas and file screens. Provide to requesting content clients using the SMB2.1 protocol as a transport, metadata for content.
- HTTP Server: Provides and maintains a secure and consistent file streaming service. Provide to requesting content clients using HTTP (PeerDist) as a transport, metadata for content.
- Object Store: Stores files and directories.

Minimal Guarantees: No action is taken that affects other directories or shares on the content server.

Success Guarantee: The Application obtains a handle to the requested file.

Trigger: The application receives a request from the user to read a file.

3.3.4.5.2 Main Success Scenario

- The Application establishes a communication channel to the content server.
- The content server authenticates the User through the mechanisms specified in [MS-AUTHSO].
- The content server authorizes the User.
- The content server provides metadata in the format of [MS-PCCRC] to the content client.
- The content server returns a **file handle** (either Block SMB2.1 or stream HTTP) to the Application.

3.3.5 Detail Use Cases - Reading and Caching a File from a Content Server

3.3.5.1 Read a file Using SMB2.1 Metadata Retrieval with Hosted Cache (Cached Data Unavailable)

Goal: Read a file from a content server with content caching support enabled using SMB2.1 for metadata retrieval, when the caching mode is hosted. (See also section 6.1.1.)

Context of use: User has located a file on a content server on a WAN link and wants to read that file. The file is located on a shared folder with content caching support enabled. The file is not initially on the LAN and needs to be retrieved from the remote end of a WAN.

Minimal Guarantees: No action is taken that affects other files exposed on the content server as a result of this operation. Files Access Services operate with no content caching.

Success Guarantee: The User reads the file.

Trigger: User action in the Application.

3.3.5.1.1 Stakeholders and Interests Summary

The stakeholders and their associated interests are as follows:

User - The User's interest is to use the Application to access files on a content server.

Application - The Application's interest is to consume file reading services by means of the content client.

Content Client - The content client's interest is to use client-side protocol components and consume the file services that are offered by the content server and to offer received content to the hosted cache server.

Content Server - The content server's interest in this Use Case is to provide and maintain a secure and consistent File Service [MS-FSSO] and provide content metadata.

Hosted Cache Server - The hosted cache server's interest is in providing two mechanisms, one for querying for the availability of certain content, and the other for retrieving content from a content client.

3.3.5.1.2 Main Success Scenario

The User requests the Application to read a file.

The Application uses the content client to open a file handle to the required existing file on the content server using the mechanisms of SMB2.1. This operation completes successfully.

The Application directs the content client to read data from the file represented by the file handle. As part of that operation, the content client requests content information (metadata) (block hashes, segment hashes (**HoD**, and private segment keys Kp) for the data [MS-SMB2].

The content server sends content identifiers on the same channel to the content client, in this case SMB Version 2.1 [MS-SMB2].

The content client computes segment identifiers (**HoHoDk**) for the data [MS-PCCRC].

The content client queries the hosted cache server for the availability of blocks from the target segments using [MS-PCCRR].

The hosted cache server indicates that it does not have the required blocks [MS-PCCRR].

The content client retrieves the content from the content server using the mechanism of [MS-FSSO].

The retrieved data is placed in the local cache of the content client computer.

The content client retrieves the data from the local cache and returns it to the Application.

The Application delivers the file contents to the User.

The content client offers the metadata content to the Hosted Cache server [MS-PCHC].

The hosted cache server optionally authenticates the content client.

The retrieved data is placed in the distributed cache of the content client computer.

The hosted cache server requests from the content client any desired segments and blocks [MS-PCCRR].

The content client sends the requested segments and blocks to the hosted cache server [MS-PCCRR].

3.3.5.1.3 System Assumptions and Preconditions

The following preconditions must be satisfied for distributed cache mode to operate successfully:

Preconditions: The client computers are configured to use content caching and retrieval. A shared folder with the desired file has been created on the content server with content caching support enabled. The user has located the URL of the file on the content server. The client computers are configured with the location of the hosted cache server.

Note that the specific URL (that is, \\server\share\file and \\192.168.0.3\share\file) will be considered different for local caching but not for content caching and retrieval.

System Availability: Appropriate content caching components must be installed and enabled on all the computers involved.

Domain Configuration: In a domain configuration, Clients and Servers have access to directory services provided by the domain. (Note that domain configuration is not a requirement but can be used).

Authentication Services: Authentication services as described in [MS-AUTHSO] are available to all Clients and Servers.

Network Configuration: In order for system components running on different computers to communicate, the network services and infrastructure must be functional and configured such that required protocols, ports, and so on are remotely accessible.

3.3.5.2 SMB2.1 Metadata Retrieval with Hosted Cache (Cached Data Available)

Goal: Read a file from a content server with content caching support enabled using SMB2.1, when the caching mode is hosted. (See also section 6.1.1, Reading a File Using SMB2.1 as a Metadata Channel.)

Context of Use: The User has located a file on a content server on a WAN link and wants to read that file. The file is located on a shared folder with content caching support enabled. The file is has previously been retrieved across the WAN link and is cached on the LAN.

Minimal Guarantees: No action is taken that affects other files exposed on the content server as a result of this operation. Files Access Services operate with no content caching.

Success Guarantee: The User reads the file.

Trigger: User action in the Application.

3.3.5.2.1 Stakeholders and Interests Summary

The stakeholders and their associated interests are as follows:

User - The User's interest is to use the Application to access files on a content server.

Application - The Application's interest is to consume file reading services by means of the content client.

Content Client - The content client's interest is to use client-side protocol components and consume the file services that are offered by the content server.

Content Server - The content server's interest in this Use Case is to provide and maintain a secure and consistent File Service [MS-FSSO] and provide content metadata using the SMB2.1 protocol as transport.

The Hosted Cache Server - The hosted cache server's interest is in providing two mechanisms, one for querying for the availability of certain content, and the other for retrieving content from a server.

3.3.5.2.2 Main Success Scenario

The User requests the Application to read a file.

The Application uses the content client to open a file handle to the required existing file on the content server using the mechanisms of SMB2.1. This operation completes successfully.

The Application directs the content client to read data from the file represented by the file handle. As part of that operation, the content client requests content information metadata (block hashes, segment hashes, and private segment keys) for the data [MS-SMB2].

The content server sends content identifiers on the same channel, in this case SMB Version 2.1, to the content client [MS-SMB2].

The content client computes segment identifiers (HoHoDk) for the data [MS-PCCRC].

The content client queries the hosted cache server for the availability of the target data [MS-PCCRR].

The hosted cache server returns the requested data [MS-PCCRR].

The retrieved data is placed in the distributed cache of the content client computer.

The retrieved data is placed in the local cache of the content client computer

The content client retrieves the data from the local cache and supplies that data to the Application.

The Application delivers the file contents to the User.

3.3.5.2.3 System Assumptions and Preconditions

The following preconditions must be satisfied for distributed cache mode to operate successfully:

Preconditions - The Client computers are configured to use content caching and retrieval. A shared folder with the desired file has been created on the content server with content caching support enabled. The user has located the URL of the file on the content server. A prior client within the local network has retrieved the data, enabling some or all of the content to be stored in a Distributed cache peer.

Note that the specific URL (that is, \\server\share\file and \\192.168.0.3\share\file) will be considered different for the local file cache but not for content caching and retrieval.

System Availability - Appropriate content caching components must be installed and enabled on all the computers involved.

Domain Configuration: In a domain configuration, Clients and Servers have access to directory services provided by the domain. (Note that domain configuration is not a requirement but can be used).

Authentication Services: Appropriate Authentication services as described in [MS-AUTHSO] are available to all Clients and Servers. Domain membership is not a requirement for clients and servers.

Network Configuration: In order for system components running on different computers to communicate, the network services and infrastructure must be functional and configured such that required protocols, ports, and so on are remotely accessible.

3.3.5.3 SMB2.1 Metadata Retrieval with Distributed Cache (Cached Data Unavailable)

Goal - Read a file from a content server with content caching support enabled using SMB2.1, when the caching mode is distributed. (See also section 6.1.1, Reading a File Using SMB2 as a Metadata Channel.)

Context of Use - The User has located a file on a content server on a WAN link and wants to read that file. The file is located on a shared folder with content caching support enabled. The file is not initially on the LAN and needs to be retrieved from the remote end of a WAN.

Minimal Guarantees - No action is taken that affects other files exposed on the content server as a result of this operation.

Success Guarantee - The User reads the file.

Trigger - User action in the Application.

3.3.5.3.1 Stakeholders and Interests Summary

The stakeholders and their associated interests are as follows:

User -The User's interest is to use the Application to access files on a content server.

Application - The Application's interest is to consume file reading services by means of the content client.

Content Client - The content client's interest is to use client-side protocol components and consume the file services that are offered by the content server.

Content Server - The content server's interest in this Use Case is to provide and maintain a secure and consistent File Service [MS-FSSO] and provide content metadata using the SMB2.1 protocol as transport.

Distributed Cache Peers - A distributed cache peer's interest is in providing two mechanisms, one responding to broadcasts for cached content, and the other for delivering the cached content.

3.3.5.3.2 Main Success Scenario

The User requests the Application to read a file.

The Application uses the content client to open a **file handle** [MS-SMB2] to the required existing file on the content server using the mechanisms of SMB2.1. This operation completes successfully.

The Application directs the content client to read data from the file represented by the file handle. As part of that operation, the content client requests content information metadata (block hashes, segment hashes, and private segment keys) for the data [MS-SMB2].

The content server sends metadata on the same channel to the content client [MS-SMB2].

The content client computes a segment discovery key [MS-PCCRC].

The client broadcasts a probe message [MS-PCCRD] to discover if any distributed cache peers have the required content.

No distributed cache peer responds with an indication of content. (Note that successful discovery would be via [MS-PCCRD].)

The content client retrieves the data from the content server [MS-SMB2].

The retrieved data is placed in the local cache of the content client computer.

The content client retrieves the data from the local cache and supplies that data to the Application.

The Application delivers the file contents to the User.

The retrieved data is placed in the distributed cache of the content client computer.

3.3.5.3.3 System Assumptions and Preconditions

The following preconditions must be satisfied for distributed cache mode to operate successfully:

Preconditions - The client computers are configured to use content caching and retrieval. A shared folder with the desired file has been created on the content server with content caching support enabled. The user has located the URL of the file on the content server.

Note that the specific URL (that is, \\server\share\file and \\192.168.0.3\share\file) will be considered different for local caching but not for content caching and retrieval.

System Availability - Appropriate content caching components must be installed and enabled on all the computers involved.

Domain Configuration: In a domain configuration, clients and servers have access to directory services provided by the domain. (Note that domain configuration is not a requirement but can be used.)

Authentication Services: Authentication services as described in [MS-AUTHSO] are available to all Clients and Servers.

Network Configuration: In order for system components running on different computers to communicate, the network services and infrastructure must be functional and configured such that required protocols, ports, and so on are remotely accessible.

3.3.5.4 SMB2.1 Metadata Retrieval with Distributed Cache (Cached Data Available)

Goal - Read a file from a content server with content caching support enabled using SMB2.1, when the caching mode is distributed.

Context of Use - The User has located a file on a content server on a WAN link and wants to read the file. The file is located on a shared folder with content caching support enabled. The file has previously been retrieved across the WAN link and is cached on the LAN.

Minimal Guarantees - No action is taken that affects other files exposed on the content server as a result of this operation.

Success Guarantee - The User reads the file.

Trigger - User action in the Application.

3.3.5.4.1 Stakeholders and Interests Summary

The stakeholders and their associated interests are as follows:

User - The User's interest is to use the Application to read files on a content server.

Application - The Application's interest is to consume file reading services by means of the content client.

Content Client - The content client's interest is to use client-side protocol components and consume the file services that are offered by the content server.

Content Server - The content server's interest is to provide and maintain a secure and consistent File Service [MS-FSSO] and provide content metadata using the SMB2.1 protocol as transport.

Distributed Cache Peers - A distributed cache peer's interest is to respond to broadcasts for cached content and to deliver the cached content.

3.3.5.4.2 Main Success Scenario

The User requests the Application to read a file.

The Application uses the content client to open a file handle [MS-SMB2] to the required existing file on the content server using the mechanisms of SMB2.1. This operation completes successfully.

The Application directs the content client to read data from the file represented by the file handle. As part of that operation, the content client requests content information (metadata) (block hashes, segment **hashes** HoD, and private segment keys Kp) for the data [MS-SMB2].

The content server sends metadata on same channel to client [MS-SMB2].

The content client computes a segment discovery key [MS-PCCRC].

The content client broadcasts a Probe Message [MS-PCCRD].

A distributed cache peer with the required content responds with a Probe Match message [MS-PCCRD].

The content client requests the content from the distributed cache peer [MS-PCCRR].

The content client receives the data and decrypts the data [MS-PCCRR].

The content client validates the block data against the block hash [MS-PCCRC].

The retrieved data is placed in the distributed cache of the content client computer.

The retrieved data is placed in the local cache of the content client computer.

The content client retrieves the data from the local cache and supplies that data to the Application.

The Application delivers the file contents to the User.

3.3.5.4.3 System Assumptions and Preconditions

The following preconditions must be satisfied for distributed cache mode to operate successfully:

Preconditions - The Client computers are configured to use content caching and retrieval. A shared folder with the desired file has been created on the content server with content caching support enabled. The user has located the URL of the file on the content server. Note that the specific URL (that is, \\server\share\file and \\192.168.0.3\share\file) will be considered different for local caching but not for content caching and retrieval.

A prior client within the LAN has retrieved the data, enabling some or all of the content to be stored in a distributed cache peer.

System Availability - Appropriate content caching components must be installed and enabled on all the computers involved.

Domain Configuration: In a domain configuration, clients and servers have access to directory services provided by the domain. (Note that domain configuration is not a requirement but can be used.)

Authentication Services: Authentication services as described in [MS-AUTHSO] are available to all clients and servers.

Network Configuration: In order for system components running on different computers to communicate, the network services and infrastructure must be functional and configured such that required protocols, ports, and so on are remotely accessible.

3.3.5.5 HTTP Metadata Retrieval with Hosted Cache (Cached Data Unavailable)

Goal - Read a file from a content server Web site with content caching support enabled using HTTP, when the caching mode is hosted. (See also section 6.1.2, Reading a File Using HTTP with Content Caching as the Metadata Channel.)

Context of Use - The User has located a file on a content server (a Web server on a WAN link) and wants to read the file. The file is located on a Web site with caching support enabled. The file is not initially on the LAN and needs to be retrieved across the WAN link.

Minimal Guarantees - No action is taken that affects other files exposed on the content server as a result of this operation. HTTP access operates with no content caching.

Success Guarantee - The User reads the file.

Trigger - User action in the Application.

3.3.5.5.1 Stakeholders and Interests Summary

User - The User's interest is to use the Application to read files on a (Web) content server.

Application - The Application's interest is to consume file reading services by means of the content client.

Content Client - The content client's interest is to use client-side protocol components and consume the file services that are offered by the content server.

Content Server - The content server's interest is to provide and maintain a secure and consistent HTTP File Access Service and provide content metadata using the HTTP protocol as transport.

Hosted Cache Server - The hosted cache server's interest is to cache data and to receive metadata from clients regarding the availability of data segments and then download the blocks within the segment that it actually requires and to supply blocks of data to clients when requested. Then later,

when another client requests the content through a secure mechanism the content can be retrieved from the hosted cache rather than from the content server.

3.3.5.5.2 Main Success Scenario

The User requests the Application to read a file.

The Application establishes an HTTP connection to the content server.

The content server authenticates the User through the mechanisms of [MS-DISO] if required.

The Application performs an HTTP GET request as specified in [RFC2616] decorated with PeerDist encoding [MS-PCCRTP].

The content server checks the authorization of the User to perform the action if required [MS-AUTHSO].

The content server retrieves metadata (block hashes, segment hashes, and private segment key) for the data [MS-PCCRC].

The content server sends metadata on the same application channel to the content client, in this case HTTP [MS-PCCRTP].

The content client computes a segment discovery key [MS-PCCRC].

The content client queries the hosted cache server for the availability of the target data [MS-PCCRR].

The hosted cache server does not have the data.

The content client retrieves the data from the content server using the HTTP protocol.

The retrieved data is placed in the distributed cache of the content client computer.

The retrieved data is placed in the local cache of the content client computer.

The content client offers the data to the hosted cache server [MS-PCHC].

The hosted cache server shows an interest in the data [MS-PCHC].

The hosted cache server requests from the content client any desired segments and blocks [MS-PCCRR].

The content client sends the requested segments and blocks to the hosted cache server [MS-PCCRR].

The Application delivers the file contents to the User.

3.3.5.5.3 System Assumptions and Preconditions

The following preconditions must be satisfied for hosted cache mode to operate successfully:

Preconditions: The content client and hosted cache server computers are configured to use content caching. The Client is configured with the network address of the server maintaining the hosted cache. A Web site with a file has been created on the content server with caching support enabled. The user has located the URL of the file on the content server.

System Availability: Content (BranchCache) caching must be installed and enabled on all the computers involved.

Domain Configuration: In a domain configuration, client and servers have access to directory services provided by the domain. (Note that domain configuration is not a requirement.)

Authentication Services: Authentication services as described in [MS-AUTHSO] are available to all clients and servers.

Network Configuration: In order for system components running on different computers to communicate, the network services and infrastructure must be functional and configured such that required protocols, ports, and so on are remotely accessible.

3.3.5.6 HTTP Metadata Retrieval with Distributed Cache (Cached Data Available)

Goal - Read a file from a content server Web site with content caching support enabled using HTTP, when the caching mode is distributed. (See also 6.1.2, Reading a File Using HTTP with Content Caching as the Metadata Channel.)

Context of Use - The User has located a file on a content server (a Web server on a WAN link) and wants to read that file. The file is located on a Web site with content caching support enabled. The file has previously been retrieved across the WAN link and is cached on the LAN.

Minimal Guarantees - No action is taken that affects other files exposed on the content server as a result of this operation. HTTP access operates with no content caching.

Success Guarantee - The User reads the file.

Trigger - User action in the Application.

3.3.5.6.1 Stakeholders and Interests Summary

User - The User's interest is to use the Application to read files on a (Web) content server.

Application - The Application's interest is to consume file reading services by means of the content client.

Content Client - The content client's interest is to utilize client-side protocol components and consume the file services that are offered by the content server.

Content Server - The content server's interest in this Use Case is to provide and maintain a secure and consistent HTTP File Access Service and provide content metadata using the HTTP protocol as transport.

Distributed Cache Peers - A distributed cache peer's interest is in providing two mechanisms, one responding to broadcasts for cached content, and the other for delivering cached content.

3.3.5.6.2 Main Success Scenario

The User requests the Application to read a file.

The Application establishes an HTTP connection to the content server.

The content server authenticates the User through the mechanisms of [MS-DISO] if required.

The Application performs an HTTP GET request as specified in [RFC2616] decorated with [MS-PCCRTP] extensions.

The content server checks the authorization of the User to perform the action [MS-AUTHSO] if required.

The content server retrieves metadata (block hashes, segment hashes, and private segment key) for the data [MS-PCCRC].

The content server sends metadata on the same application channel to the content client [MS-PCCRTP].

The content client computes a segment discovery key [MS-PCCRC].

The client broadcasts a probe message [MS-PCCRD] to discover if any distributed cache peers have the required content.

A distributed cache peer with the required content responds with a probe match message [MS-PCCRD].

The content client requests the segment from the distributed cache peer [MS-PCCRR].

The content client receives the data and decrypts the data [MS-PCCRR].

The content client validates the block data against the block hash [MS-PCCRC].

The retrieved data is placed in the distributed cache of the content client computer.

The retrieved data is placed in the local cache of the content client computer.

The content client retrieves the data from the local cache and supplies that data to the Application.

The Application delivers the file contents to the User.

3.3.5.6.3 System Assumptions and Preconditions

The following preconditions must be satisfied for distributed cache mode to operate successfully:

Preconditions: The client's computers are configured to use content caching and retrieval. A Web site with a file has been created on the content server with caching support enabled. The user has located the URL of the file on the content server. A prior client within the LAN has retrieved the data enabling some or all of the content to be stored in a distributed cache peer.

System Availability: Content (BranchCache) caching must be installed and enabled on all the computers involved.

Domain Configuration: In a domain configuration, clients and servers have access to directory services provided by the domain. (Note that domain configuration is not a requirement.)

Authentication Services: Authentication services as described in [MS-AUTHSO] are available to all clients and servers.

Network Configuration: In order for system components running on different computers to communicate, the network services and infrastructure must be functional and configured such that required protocols, ports, and so on are remotely accessible.

3.3.5.7 BITS (HTTP Metadata Retrieval) with Distributed Cache - Application (Cached Data Unavailable)

Goal - Read a file from a content server Web site with content caching support enabled using BITS, when the caching mode is distributed. (See also section 6.1.5, Reading a File Using BITS with Content Caching.)

Context of Use - The User has located a file on a content server and wants to read that file. The file is located on a Web site with content caching support enabled. The file is has previously been retrieved across the WAN link and is cached on the LAN.

Minimal Guarantees - No action is taken that affects other files exposed on the content server as a result of this operation. HTTP access operates with no content caching.

Success Guarantee - The User reads the file.

Trigger - User action in the Application.

3.3.5.7.1 Stakeholders and Interests Summary

User - The User's interest is to use the Application to read files on a content server.

Application - The Application's interest is to consume file reading services by means of the content client.

Content Client - The content client's interest is to use client-side protocol components and consume the file services that are offered by the content server.

Content Server - The content server's interest in this use case is to provide and maintain a secure and consistent HTTP File Access Service and provide content metadata using the HTTP protocol as transport.

Distributed Cache Peers - A distributed cache peer's interest is in providing two mechanisms, one responding to broadcasts for cached content, and the other for delivering cached content.

3.3.5.7.2 Main Success Scenario

The User requests the Application to read a file.

The Application establishes an HTTP connection to the content server.

The content server authenticates the User through the mechanisms of [MS-DISO] if required.

The Application performs an HTTP GET request as specified in [RFC2616] decorated with [MS-PCCRTP] extensions.

The content server checks the authorization of the User to perform the action [MS-AUTHSO] if required.

The content server retrieves metadata (block hashes, segment hashes, and private segment key) for the data [MS-PCCRC].

The content server sends metadata on the same application channel to the content client [MS-PCCRTP].

The content client computes a segment discovery key [MS-PCCRC].

The client broadcasts a Probe Message [MS-PCCRD] to discover if any of the distributed cache peers have the required content.

No distributed cache peer responds with an indication of possessing the required content (note that successful discovery would proceed using [MS-PCCRD]).

The content client retrieves the data from the content server using the HTTP protocol.

The retrieved data is placed in the local cache of the content client computer.

The content client retrieves the data from the local cache and supplies that data to the Application.

The Application delivers the file contents to the User.

The retrieved data is placed in the distributed cache of the content client computer.

3.3.5.7.3 System Assumptions and Preconditions

The following preconditions must be satisfied for hosted cache mode to operate successfully:

Preconditions: The content client and hosted cache server computers are configured to use content caching. The client is configured with the network address of the server maintaining the hosted cache. A Web site with a file has been created on the content server with caching support enabled. The user has located the URL of the file on the content server.

System Availability: Content (BranchCache) caching must be installed and enabled on all the computers involved.

Domain Configuration: In a domain configuration, clients and servers have access to directory services provided by the domain. (Note that domain configuration is not a requirement.)

Authentication Services: Authentication services as described in [MS-AUTHSO] are available to all clients and servers.

Network Configuration: In order for system components running on different computers to communicate, the network services and infrastructure must be functional and configured such that required protocols, ports, and so on are remotely accessible

3.3.5.8 BITS (HTTP Metadata Retrieval) with Hosted Cache - Application (Cached Data Available)

Goal - Retrieve a file from a content server Web site with content caching support enabled using BITS. The caching mode is hosted. (See also section 6.1.4.1.)

Context of Use - The User has located a file on a content server (a Web server on a WAN link) and wants to read that file. The file is located on a Web site with caching support enabled. The file is not initially on the LAN and needs to be retrieved across a WAN link.

Minimal Guarantees - No action is taken that affects other files exposed on the content server as a result of this operation. HTTP access operates with no content caching.

Success Guarantee - The User retrieves the file.

Trigger - User action in the Application.

3.3.5.8.1 Stakeholders and Interests Summary

User - The User's interest is to use the Application to read files on a content server.

Application - The Application's interest is to consume file reading services by means of the content client.

Content Client - The content client's interest is to use client-side protocol components and consume the file services that are offered by the content server and to offer any available content received to the hosted cache server.

Content Server - The content server's interest in this use case is to provide and maintain a secure and consistent HTTP File Access Service and provide content metadata using the HTTP protocol as transport.

Hosted Cache Server - The hosted cache server's interest is to cache data and to receive metadata from clients regarding the availability of data segments and then download the blocks within the segment that it actually requires and to supply blocks of data to clients when requested. Then later, when another client requests the content through a secure mechanism, the content can be retrieved from the hosted cache rather than from the content server.

3.3.5.8.2 Main Success Scenario

The User requests the Application to read a file.

The Application establishes an HTTP connection to the content server.

The content server authenticates the User through the mechanisms of [MS-DISO] if required.

The Application performs an HTTP GET request as specified in [RFC2616] decorated with [MS-PCCRTP] extensions.

The content server checks the authorization of the User to perform the action [MS-AUTHSO] if required.

The content server retrieves metadata (block hashes, segment hashes, and private segment key) for the data [MS-PCCRC].

The content server sends metadata on the same application channel to the content client [MS-PCCRTP].

The content client computes a segment discovery key [MS-PCCRC].

The content client queries the hosted cache server for the availability of the target data [MS-PCCRR].

The hosted cache server has some or all of the required data.

The content client retrieves the available data from the hosted cache server using [MS-PCCRR]. Any unavailable data is retrieved direct from the content server using the HTTP protocol.

The content client offers any data retrieved directly from the content server to the Hosted Cache server using [MS-PCHC].

The hosted cache server requests from the content client any desired segments and blocks [MS-PCCRR].

The content client sends the requested segments and blocks to the hosted cache server using [MS-PCCRR].

The retrieved data is placed in the distributed cache of the content client computer.

The retrieved data is placed in the local cache of the content client computer.

The content client retrieves the data from the local cache and supplies that data to the Application.

The Application delivers the file contents to the User.

3.3.5.8.3 System Assumptions and Preconditions

The following preconditions must be satisfied for hosted cache mode to operate successfully:

Preconditions: The content client and hosted cache server computers are configured to use content caching. The client is configured with the network address of the server maintaining the hosted cache. A Web site with a file has been created on the content server with caching support enabled. The user has located the URL of the file on the content server. A prior client within the LAN has retrieved the data, enabling some or all of the content to be stored on the hosted cache server.

System Availability: Content (BranchCache) caching must be installed and enabled on all the computers involved.

Domain Configuration: In a domain configuration, clients and servers have access to directory services provided by the domain. (Note that domain configuration is not a requirement.)

Authentication Services: Authentication services as described in [MS-AUTHSO] are available to all clients and servers.

Network Configuration: In order for system components running on different computers to communicate, the network services and infrastructure must be functional and configured such that required protocols, ports, and so on are remotely accessible.

4 System Context

This section describes the relationship between this system and its environment.

4.1 System Environment

Network Infrastructure: This system requires access to network services that support:

TCP/IP (IPv4 or IPv6)

UDP/IP (IPv4 or IPv6)

Domain Naming System (DNS) name resolution.

Cache mode	Protocol	Activity	Port
Distributed and Hosted	HTTP	Content Retrieval (uses HTTP)	TCP Port 80
Hosted	HTTPS	Content Offering (HTTPS)	TCP Port 443
Distributed	WS-Discovery	Peer Discovery (Uses WSD)	UDP port 3702

Object Store: Both SMB2 and HTTP File Services must have access to a hierarchical object store for persistence of files and namespace. The Object Store would typically be built on file system(s) available in the host operating system, but the Object Store may also need to include functionality in addition to that provided by the file system. A File Service may need to implement additional logic to convert between the semantics of a particular **File Access Protocol** and semantics of the available Object Store. Some semantics implemented in Windows file systems are directly visible when using the SMB Access Protocols, and such wire-visible behaviors are documented in [MS-FSA].

Case Sensitivity: The Object Store must support case-insensitive operation for the SMB File Service, and should support case-sensitive operation for the HTTP File Service.

Accounts: If computers hosting File Clients and File Services are not joined to a domain, then user accounts must be configured across computers by some external mechanism.

Hosted Cache: If a hosted cache is to be used, the location of the Hosted Cache (DNS Name or IP Address) and the Hosted Cache Listen and Connections Ports MUST be configured on the client computers.

4.2 System Assumptions and Preconditions

The following assumptions and preconditions must be satisfied for the Content Caching and Retrieval System to operate successfully:

System Availability:

The File Access Services System must be installed on all the computers involved in content caching.

Internet Information Services (IIS) must be installed and configured on the content server for HTTP caching.

Internet Explorer or BITS client must be installed on client computers for HTTP caching.

The Content Caching and Retrieval System components (BranchCache) must be installed on the computers involved.

Authentication Services: Authentication services as described in [MS-AUTHSO] are available to all File Clients and File Services.

Network Configuration: In order for system components running on different computers to communicate, the network services and infrastructure must be functional and configured such that required protocols, ports, and so on are remotely accessible.

Domain Configuration: In a domain configuration, File Client and File Service have access to directory services provided by the domain.

Domain Functionality: For system functionality requiring a domain (as defined in [MS-DISO]) and directory services (as defined in [MS-ADSO]), at least one domain controller must be configured and accessible. Domain functionality is not a requirement of the Content Caching and Retrieval System.

4.3 System Relationships

This section defines the Content Caching and Retrieval System as a black box, and examines the relationships of the black box to components outside of the black box, in terms of system dependencies and system influences.

4.3.1 Black Box Relationship Diagram

The Content Caching and Retrieval System is shown in the following figure as a "black box", showing a high-level abstraction of the Content Caching and Retrieval System's major components, and the external systems and components with which the system interacts.

The abstract components of the Content Caching and Retrieval System are as follows.

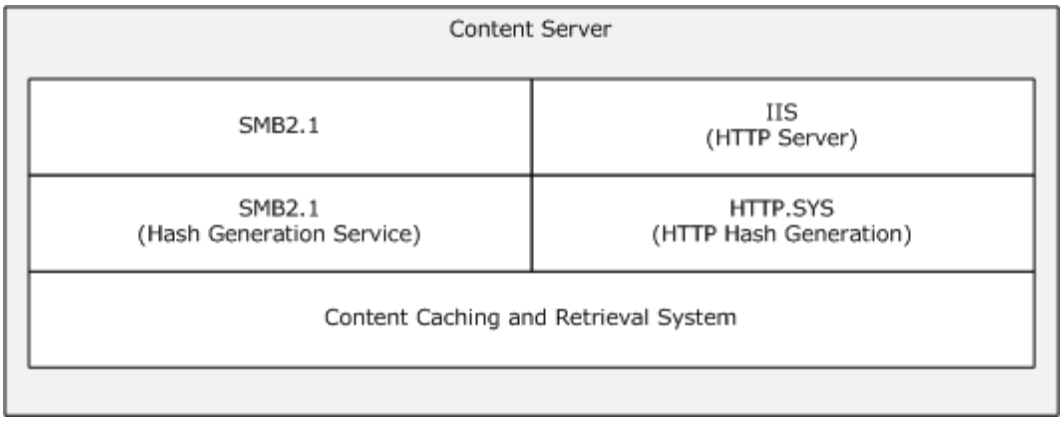

Figure 3: Abstraction view of a content server

A content server File is an abstraction of the software running on a file and Web server that provides remote file access to one or more Users. The abstraction can be considered too be made up of two subsystems: Block File Services (SMB2.1) and Streaming File Services (HTTP). A content server can be managed by one or more Administrators. Internally, the content server contains an object store that contains the static content made available through the SMB2.1 and HTTP protocols. The object store is an implementation-dependent local file system.

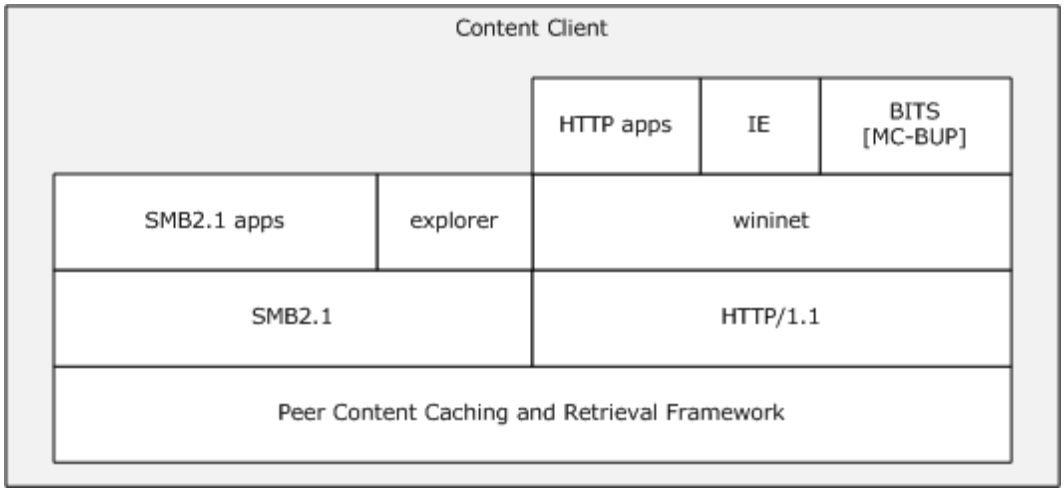

Figure 4: Abstraction view of a content client

Content client and Admin Client are abstractions of low-level protocol state and operating software that runs in application and administrative nodes, respectively. These Clients are considered part of the Content Caching and Retrieval System. They are used by Application and Administrator Tool software to effect communication with components of the System.

The Application and Administrator Tool represent programs with which a User and an Administrator typically interact, respectively. It should be understood that this is a somewhat arbitrary distinction, and that many programs have both Application and Administrator Tool features. For example, a graphical shell is frequently used directly by a user to perform personal file management tasks. This type of program is classified as an Application, but it certainly has some aspects of an Administrator Tool.

The dotted lines model dependency and influences relationships between the Content Caching and Retrieval System and external systems and components.

The Content Caching and Retrieval System has two real dependencies. It depends on the File Access Services System for SMB2.1, and it depends on a Web Server Service for HTTP. Other external relationships are influences, because the system may, depending on the configuration, operate with reduced functionality if the external service is not present.

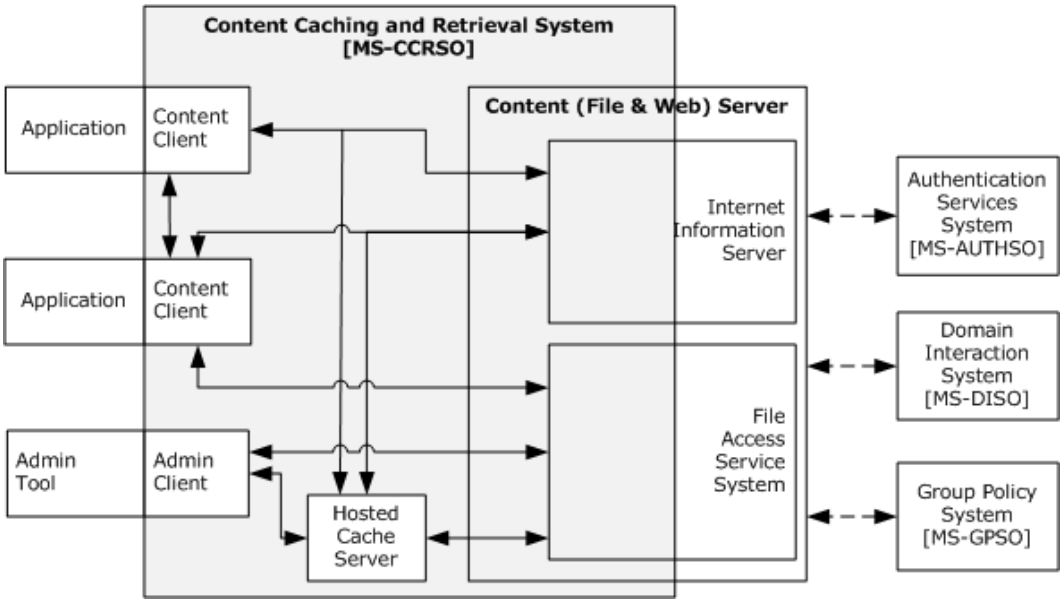

Figure 5: Content Caching and Retrieval System black box diagram

4.3.2 System Dependencies

The Content Caching and Retrieval System depends on the following external systems and components:

- The File Access Services System and dependents
- The HTTP and HTTPS protocols.

4.3.3 System Influences

The Content Caching and Retrieval System can be influenced by the external systems and components shown in the following table.

External entity	Content Caching and Retrieval System depends on external entity for…	Consequences if absent
Authentication Services System	Authenticating client and server principals.	Centralized identity management does not work if the Domain Interactions System is not available.
Group Policy System	Configuration of individual capabilities within the Content Caching and Retrieval System.	Cannot centrally configure some functionality of the system.
Domain Interaction System	Content Caching and Retrieval can operate either in a Domain or in a Workgroup. Domain services are required for Hosted Cache client authentication.	Hosted Cache client authentication cannot function.
File Access System	File Access services to which Content Caching and Retrieval is an adjunct.	No file access and therefore no content caching and retrieval.

External entity	Content Caching and Retrieval System depends on external entity for…	Consequences if absent
Web Server	HTTP Content Caching is dependent on the availability of HTTP served content.	No HTTP content caching can be performed.
Certification Authority System	Hosted Cache mode makes use of SSL; this requires a X509 Certificate.	Hosted Cache communication will fail.

Specific-system influences are as follows:

Group Policy enables a client to interact with the Content Cache Service, which is off by default. This enables a computer to act as both a client-role peer and client-role server, thereby publishing and retrieving content and metadata obtained from a content server.

The Background Intelligent Transfer Service (BITS) Upload Protocol, as defined in [MC-BUP], specifies an HTTP 1.1-based upload protocol. This protocol is used to transfer large payloads from a client to a server or server to client over networks with frequent disconnections and to send notifications about the availability of uploaded payloads. This protocol is a client of the Content Caching and Retrieval System.

4.4 System Applicability

The Content Caching and Retrieval System is supplementary to the File Access Services; it is a form of wide area network (WAN) link acceleration. The goal is to increase network utilization. The major principle employed by Content Caching and Retrieval is the reduction of traffic across a WAN. Content caching enables content from file and Web servers on one end of a WAN to be cached on computers at the other end of the WAN.

4.5 System Versioning and Capability Negotiation

The Content Caching and Retrieval System does not define any versioning or capability negotiation beyond what is described in the member protocol specifications, as listed in section 2.2. [MC-BUP], [MS-PCCRTP], and [MS-PCCRR] have version-specific capability.

SMB Version 2.1 offers functionality relevant to the Peer Content Caching and Retrieval System. Content Caching was first introduced in the Windows® 7 operating system platform and can be made available on Windows Vista® operating system and Windows Server® 2008 operating system systems by the addition of a Windows Management Framework. This makes the Peer Content Caching and Retrieval System available only to BITS clients.

4.6 System Vendor-Extensible Fields

The Content Caching and Retrieval System does not define any vendor-extensible field; neither at the system level nor at the individual protocol level.

5 System Architecture

This section describes the basic structure of the system and the interrelationships among its parts, consumers, and dependencies.

5.1 Abstract Data Model

This section describes the conceptual data organization the system maintains to provide its functionality. The data model described in this section can be implemented in a variety of ways. This specification does not prescribe any specific implementation technique.

The purpose of this section is to document shared state that is maintained between the Content Caching and Retrieval System member protocols and/or external systems in order to support operations involving more than one protocol.

The sharing of state happens internal to the server implementation. Compatible implementations can use any mechanism of their choice to perform the sharing, as long as it satisfies the requirements of this section and the protocol specifications. For example, an implementation could use a single store that is shared across all the protocols, or it could use multiple stores (one per protocol) and build a synchronization mechanism to maintain consistency across all the stores, as long as that synchronization mechanism ensures that all atomicity requirements are observed and that two protocols that share state always see an identical view of that shared state.

The following table lists the local ADM elements common to content caching peers.

ADM Element	Possible Values	Notes
System Mode	Either online or offline.	
Server Role	Either Disabled, Local, Distributed, Hosted Client or Hosted Server.	Hosted Server only applies to Servers. Local mode disables the use of remote caches.
hosted cache server Location	IPv4 or IPv6 address, NetBIOS name or FQDN.	Only applicable if role is Hosted Client.
Client Authentication	Domain or None.	Only applicable if role is Hosted Server.<3>
Hosted Cache Listen Port	IP Port	
Hosted Cache Connection Port	IP Port	
Hosted Client broadcast listen port	IP Port	
BITS_DisableBranchCache	True or False.	If True, BITS client does not use Content Caching and Retrieval.
BITS_BC_EnabledOnURL	True or False.	If True, the BITS client does not use Content Caching and Retrieval.
BITS_ReceivedBC_DataOnURL	True or False.	True if the previous content block from URL was received from CCR System.
BITS_RetryCountOnURL	Unsigned Integer	Retry block download, up to 6 times if BITS_ReceivedBC_DataOnURL is True for the

ADM Element	Possible Values	Notes
		given URL.<4>
HashPublicationForPeerCaching	Share, Disable on All Shares, Enable on All Shares.	Impacts SMB2.1 shares only.
Network Latency	Unsigned Integer	Value, in milliseconds, above which network files may be cached.
Size of Cache	Unsigned Integer	Value in bytes.

The following is a summary of ADM elements from the member protocols:

[MS-PCCRTP] - None.

[MS-PCCRD] - The following ADM items are described in detail in section 3.1.1:

Client Peer - HoHoDk List, Outstanding Probe List.

Server Peer - Available Segment List.

[MS-PCHC] - The following ADM items are described in detail in section 3.1.1:

Content information for an offered segment.

[MS-PCCRR] - The following ADM items are described in detail in sections 3.1.1 and 3.1.2:

Client Side:

Outstanding Request List: A list of request messages sent for which responses have not yet been received, along with the addresses of the peers to which they were sent.

Server Side:

Content Cache: This is the local content cache on the server. It consists of a list of segment IDs and associated block ranges, along with their Content Information (see [MS-PCCRC] section 2) and corresponding content blocks that the client or server has previously obtained either from other peers or from the content server.

Active Client Count: This counter keeps track of the number of active clients the server is currently serving.

5.1.1 Content Identifiers

For the purposes of the Content Caching and Retrieval System, content is considered to be divided into one or more segments. Segments are the unit of discovery.

Each segment is a binary string of a standard size (32 MB), except possibly the last segment which may be smaller if the content size is not a multiple of the standard segment size. Each segment is identified on the network by its Segment Identifier, also known as HoHoDk. Different content items can share the same segment if they happen to contain an identical part that coincides with a complete segment.

Each segment is divided in turn into blocks. Each block is a binary string of a fixed size (64 KB), except for the last block in the last segment, which again may be shorter. Unlike segments, blocks in different segments are always considered distinct objects, even if they are identical. Blocks within

a segment are identified by their progressive index within the segment (Block 0 is the first block in the Segment, Block 1 the second, and so on). Because of the fixed block size, a block's index can also be used to compute its actual byte offset in the segment. Given the standard block size of 64 KB, Block 0 is located at offset 0 in the segment, Block 1 at offset 65536, block 2 at offset 131072, and so on. Blocks are the unit of download.

The following figure depicts how the content is divided into segments and blocks.

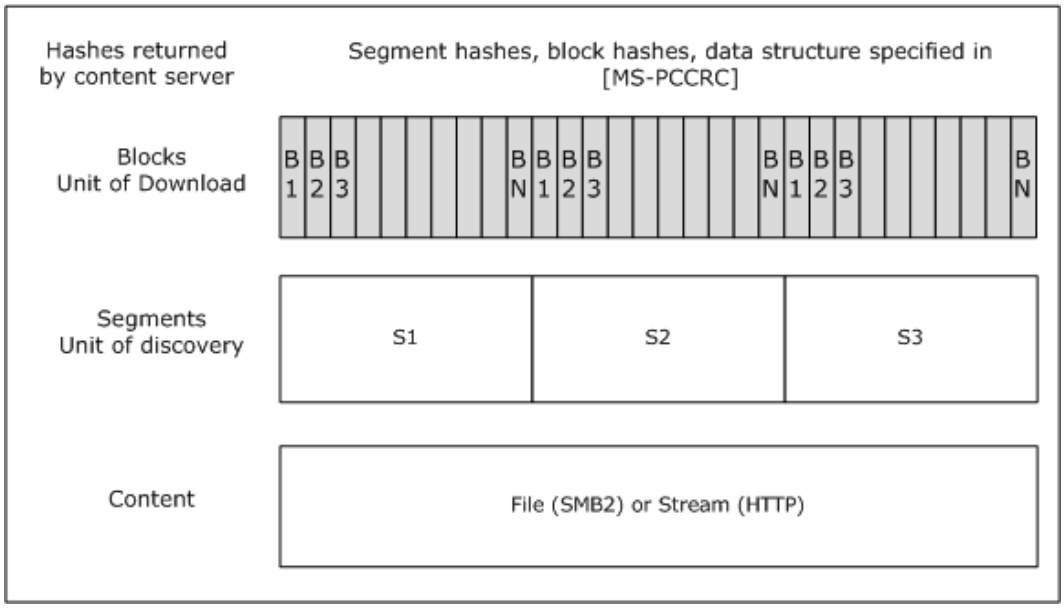

Figure 6: Content identifiers

Note that given the entire set of blocks for a segment, each identified by index, one can reconstruct the original segment simply by concatenating the blocks in order by index. Similarly, given the entire sequence of HoHoDk values for the successive segments in a content item, and a set of segments with matching associated HoHoDk values, one can reconstruct the original content simply by concatenating the segments in order based on HoHoDk value. Details of calculating the identifiers and keys can be found in [MS-PCCRC] section 2.2.

5.1.1.1 Security

The following figure illustrates how the content is encrypted by the system.

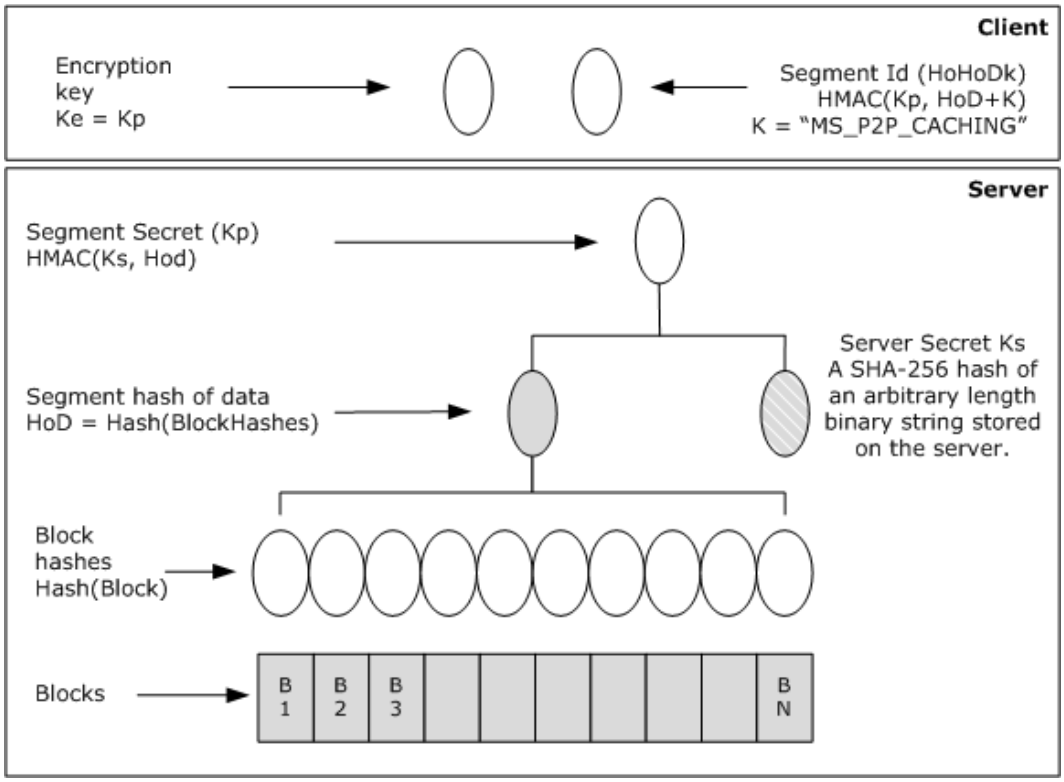

Figure 7: Content security

5.1.1.2 Client-Side Content Security

In the figure titled "Content security" in the preceding section, Ke represents an encryption key derived from the segment secret (Kp). A sending peer will encrypt data with Ke, but Ke is never disclosed between peers. The receiving client must already have obtained enough information to compute the value of Ke from a content server in order to decrypt the peer-supplied data. Ke = Kp.

5.1.1.3 Server-Side Content Security

The hash Algorithm used is the SHA-256 hash as specified in [FIPS180-2]. Further details can be found in [MS-PCCRC] section 2.2. The **HMAC** function is defined in [RFC2104].

5.1.2 Client-Role Peer

This section specifies how the Content Caching and Retrieval System, running on multiple peers, uses the Discovery Protocol [MS-PCCRD] and the Retrieval Protocol [MS-PCCRR] to allow the client-role peer to retrieve content blocks of a target segment from one or more server-role peers. Requests come from higher-layer applications on the client-role peer to retrieve the whole or parts of a content item, which may span multiple segments. For each target segment, the client-role peer uses the Discovery Protocol to find server-role peers (or directly contacts a hosted cache server) that has (the whole or parts) of the target segment. The client-role peer then initiates Retrieval Protocol exchanges to each server-role peer to query the block ranges each server-role peer has, and downloads the blocks.

The Discovery Protocol and the Retrieval Protocol both operate on or within a single segment. The operations specified in this section allow a client-role peer to find and retrieve blocks (parts or all) of a single target segment. This process is referred to as a segment retrieval session. If the content spans multiple segments, then multiple segment retrieval sessions are required to retrieve all the content's segments and reassemble them into the complete content item.

5.1.2.1 Hosted Cache Mode

A hosted cache Content Client-role peer maintains the following:

- **Content Cache**: Stores the content information supplied by the higher-layer applications (as defined in [MS-PCCRC] section 2.3).

5.1.2.2 Distributed Cache Mode

A Content Client-role peer maintains the following data:

- **Content Cache**: Stores the content information supplied by the higher-layer applications (as defined in [MS-PCCRC] section 2.3), as well as the corresponding segments and blocks of content retrieved from server-role peers or content servers. The same content cache is used by both the client role and the server role of the same peer, so that content retrieved from peers is available for retrieval by other peers. This same content cache is also used by the peer when instantiating a server in the Retrieval Protocol.

- **Server Information Lists**: For each segment requested (identified by segment ID), the client MUST maintain a (possibly empty) <5> list of servers discovered by the Discovery Protocol, and their corresponding block ranges returned by the Retrieval Protocol GetBlockList (MSG_GETBLKLIST) exchanges. Also, for each list, and each entry in each list, its time of most recent use MUST be stored, so that the least recently used item or list may be deleted to make room for a new one when the maximum number already exists. The list of servers MUST persist across multiple segment retrieval sessions, subject to an expiration timer for the list (**Server List Timer**) <6> and for each server (**Server Entry Timer**) <7> if not used. The block range information for each server MUST be deleted at the end of each segment retrieval session. By default, the maximum number of server lists MUST be 256, <8> and the maximum number of server entries in a list MUST be 16. <9>

- **Discovery Frequency Counter**: Every peer maintains a global counter measuring the rate at which the Discovery Protocol is being instantiated. This is used to throttle the rate of discovery probes sent by the peer. When this counter reaches a threshold Fmax=100Hz, then further instantiations of the Discovery Protocol MUST be blocked until the frequency comes back under 100 Hz. <10>

- **Discovered Server Counter**: For each instantiation of the Discovery Protocol, the peer maintains a distinct counter counting the number of peers discovered so far by that particular instantiation. When this counter reaches the threshold Dmax, the framework will not use any subsequent replies from additional peers for Retrieval Protocol operations, but will still add them to the **Server Information List** for use in subsequent segment retrieval sessions. The default value of Dmax MUST be set to 16. <11>

- **Download Initiated Flag**: For each segment ID requested, a flag is stored to indicate whether the Retrieval Protocol has been initiated yet for that segment ID.

- **Block Download Status**: For each block in the requested block ranges of the segment, a flag is stored to indicate the download status (idle, downloading, downloaded, or error).

5.2 White Box Relationships

The following figure shows the protocol layering relationships for the Content Caching and Retrieval System member protocols. The default relationship, indicated by a solid arrow, is "is transported by". The "includes" stereotype means that a protocol document includes a second document by reference (for example, [MS-SMB2] includes [MS-FSCC]). Member protocols are shown in shaded boxes.

Content Caching and Retrieval can be initiated both by SMB2 and HTTP. A BITS client can use the system and acts as an HTTP client.

A content client uses HTTPS for communication with a hosted cache server specifically while offering content. The content is then retrieved using [MS-PCCRR].

A content client uses [MS-PCCRD] an implementation of [WS-Discovery] to locate peer computers with cached content.

The majority of traffic in the system is [MS-PCCRR]; this protocol is used to transfer actual content regardless of the protocol that retrieved the content metadata.

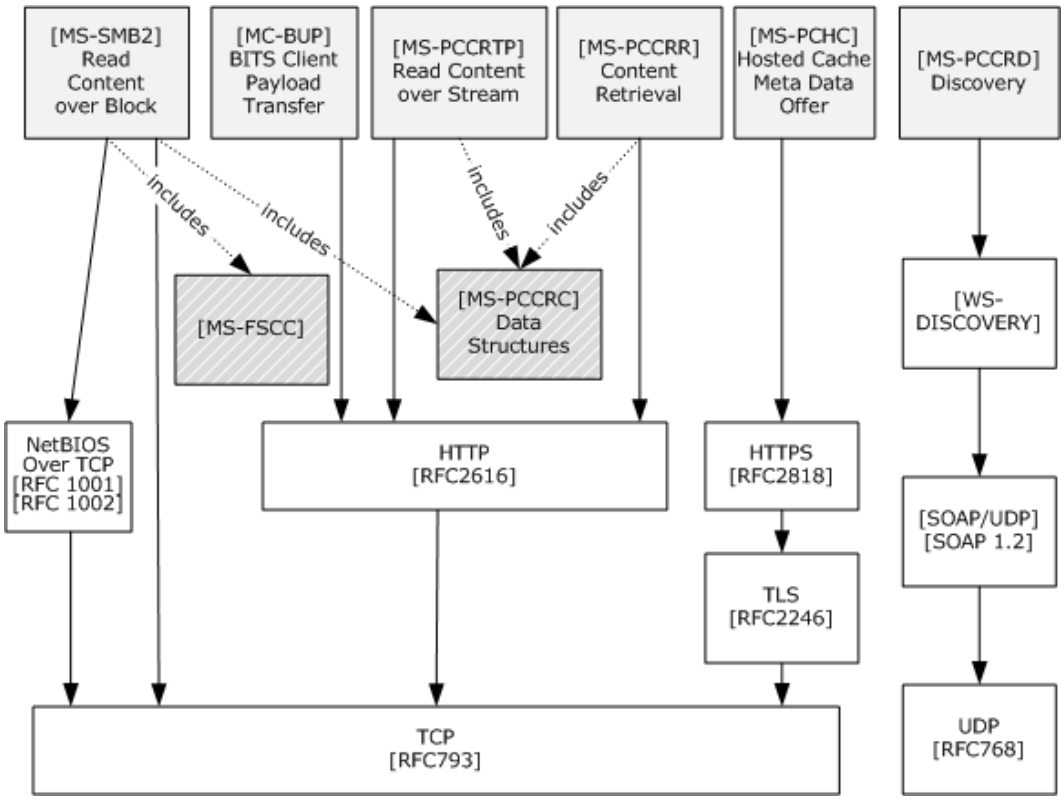

Figure 8: Protocol relationships for the Content Caching and Retrieval System

5.2.1 HTTP Metadata Retrieval Integration

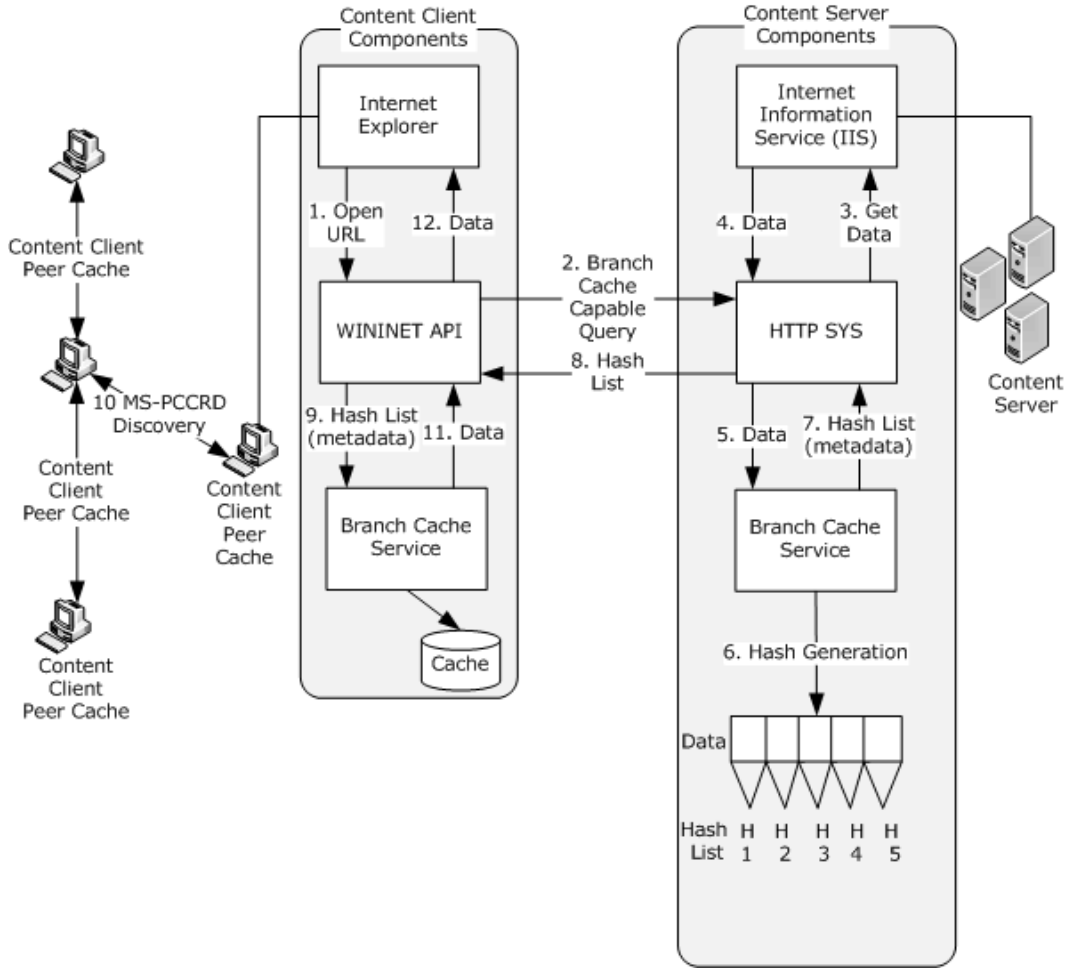

Figure 9: HTTP metadata retrieval integration

The sequence of messages for an HTTP request with content caching and retrieval are shown in the figure in this section and are as follows:

1. The Client Application opens a URL.
2. The WINHTTP on the client side decorates outgoing HTTP request with a PeerDist header.
3. IIS on the server side performs any required access checks, then if valid, requests the data from HTTP.SYS. If hashes are available, these are returned to the client.
4. If hashes are not available, a response is returned immediately to http.sys to pass the requested data back to the client.
5. If the requested data and hashes are available, the data is placed in the distributed cache.
6. Metadata in the form of hash lists is generated making the metadata available for any subsequent requests on the same data.

7. Any subsequent request for the same data identified by the URL results in metadata being returned to HTTP.SYS.

8. Metadata is returned to the content client using [MS-PCCRTP].

9. WINHTTP gets a response comprising of a hash list. The hash list is passed to the BranchCache Service to look up the data.

10. For unsuccessful cache lookups (missing block), a range request, including the byte range, is broadcast using [MS-PCCRD]. If data is found in a peer client, the data is retrieved using [MS-PCCRR].

11. If data is retrieved, it is added to the cache and the hash list made available for discovery. If WINHTTP does not get the associated data (that is, the original content is unavailable for some reason), it returns a suitable (HTTP) error to the application.

12. The requested data is returned to the requesting content client.

5.2.2 BITS Integration

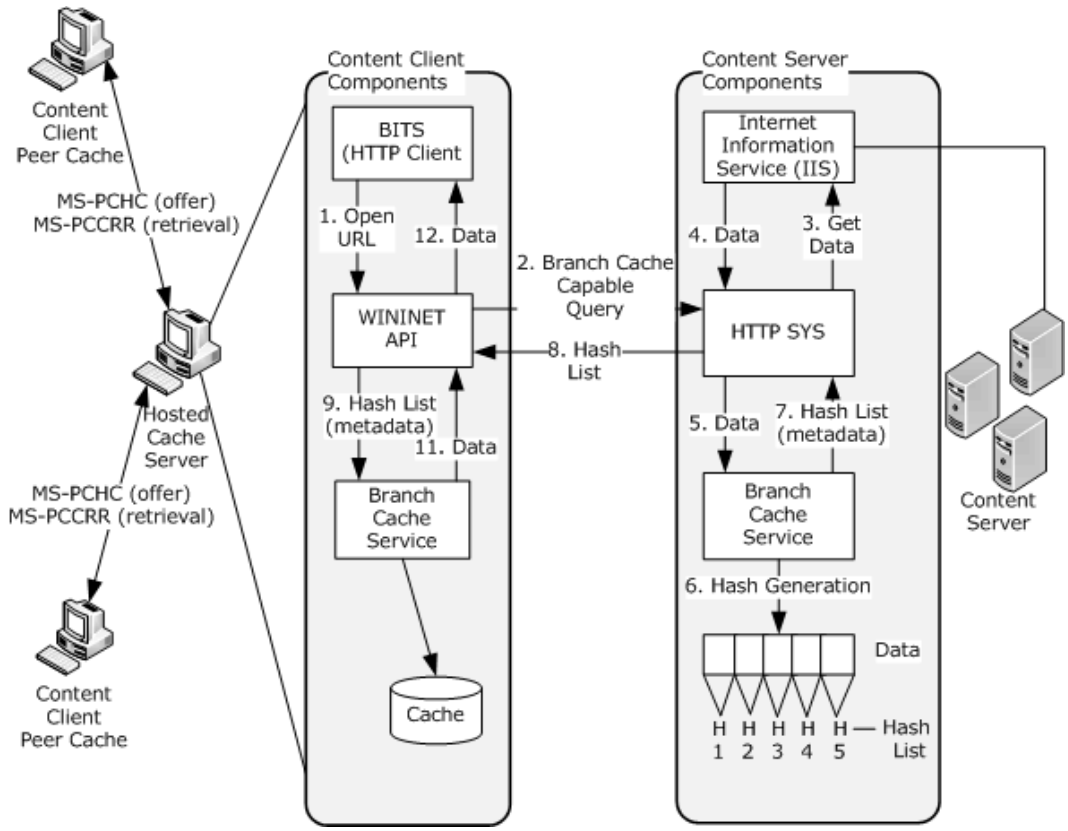

Figure 10: BITS integration

In the context of Content Caching and Retrieval, BITS (the Background Intelligent Transfer Service) is a client of the HTTP service. The figure in this section shows the Hosted Cache configuration, but Distributed is equally valid and would be the same as the figure in the previous section titled "HTTP integration", with Internet Explorer replaced by a BITS client.

The sequence of messages for a BITS request with content caching and retrieval are in fact HTTP requests. The main difference being that the BITS client makes extensive use of HTTP range requests (see [RFC2616] section 3.12) as BITS is mainly used for file transfer and may be subject to interruption, pause and continuation actions. The higher-layer protocol provides BITS with the desired range(s) of the server URL; BITS then issues single- or multi-range HTTP requests representing a subset of the desired ranges.

The sequence below is for a Hosted Cache configuration:

1. The Client Application opens a URL. **BITS_BC_EnabledOnURL** is initially set to false. **BITS_ReceivedBC_DataOnURL** is initially set to false.

2. BITS on the client side decorates the outgoing HTTP requests with a PeerDist header. BITS chooses a single- or multi-range HTTP request based on **BITS_BC_EnabledOnURL** and the size of the desired range(s), as follows:

   - When **BITS_BC_EnabledOnURL** is TRUE, then BITS sends a multi-range request only if there is a sequence of application ranges that are each smaller than 2 KB in size. Otherwise, it sends a single-range request.

   - When **BITS_BC_EnabledOnURL** is FALSE, then BITS may send a multi-range request if there is a sequence of application ranges, regardless of size. Otherwise, it sends a single-range request.

3. IIS on the server side performs any required access checks; then if valid, requests the data from HTTP.SYS. If hashes are available, these are returned to the client. If the following conditions are true, then the client sets the ADM element **BITS_BC_EnabledOnURL** to false to indicate that ranged requests cannot be used with the content caching and retrieval system:

   - The client request was a single-range request.

   - The range was part (or all) of a desired range that is larger than 2 KB.

   - The server returned content, not hashes.

4. If hashes are not available, a response is returned immediately to http.sys to pass the requested data back to the client.

5. If the requested data and the hashes are available, the data is placed in the distributed cache.

6. Metadata in the form of hash lists is generated, making the metadata available for any subsequent requests on the same data.

7. Any subsequent request for the same data identified by the URL results in metadata being returned to HTTP.SYS.

8. Metadata (hash list) is returned to the content client using [MS-PCCRTP].

9. BITS gets a response comprising a hash list. The hash list is passed to the BranchCache Service to look up the data.

10. For unsuccessful local cache lookups (missing block), a direct request for the data from the Hosted Cache is made using [MS-PCCRR]. If a block is successfully retrieved for a URL, then **BITS_ReceivedBC_DataOnURL** is set to true. Otherwise, the client checks the value of **BITS_ReceivedBC_DataOnURL**: if true, then the client will make multiple attempts for the same block using [MS-PCCRR], up to **BITS_RetryCountOnURL**. If one of the attempts succeeds, then **BITS_ReceivedBC_DataOnURL** remains true and **BITS_RetryCountOnURL** is

set to zero; otherwise, it is set to false so that only a single attempt will be made for subsequent blocks.

11. If BITS does not get the associated data (because the original content is unavailable), it returns a suitable (HTTP) error to the application. If the original data is available it is obtained from the content server.

12. The requested data is returned to the requesting content client.

Note Prior to the introduction of the Content Caching and Retrieval System, BITS clients used [MS-BPDP] for discovery and [MS-BPCR] for retrieval.<12> [MS-BPDP] and [MS-BPCR] protocols cannot coexist with the Content Caching and Retrieval System.

5.2.3 SMB2.1 Metadata Retrieval Integration

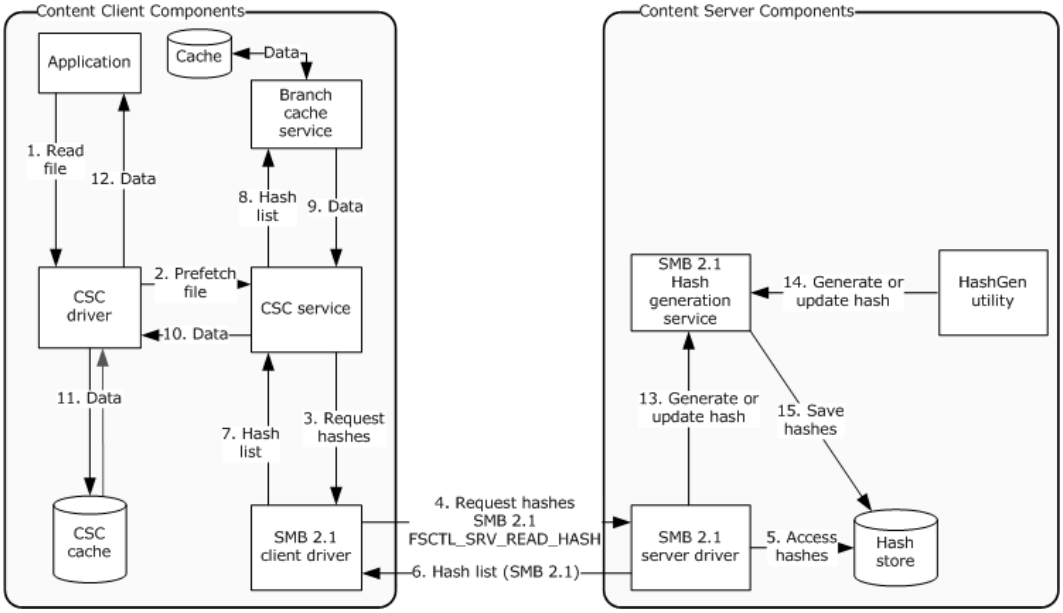

Figure 11: SMB2.1 integration

The sequence of messages for an SMB 2.1 request with content caching and retrieval are shown in the figure in this section and are as follows:

1. The Client Application opens a remote file. During this stage an SMB2.1 Tree Connect allows the content client to determine whether the remote share supports hash lists via the SHI1005_FLAGS_ENABLE_HASH flag.

2. An attempt is first made to retrieve the file from the local cache (CSC).

3. If the file is unavailable and the share supports hash lists, the CSC Service requests the hash list.

4. The SMB2.1 Client Driver performs an IOCTL FSCTL_SRV_READ_HASH.<13>

5. The SMB2.1 Server Driver accesses the hash lists.

6. The SMB2.1 Server Driver returns the hash list to the SMB2.1 client.

7. The SMB2 .1 Client Driver returns the hash list to the CSC Service.

8. The CSC Service supplies the hash list to the BranchCache Service to retrieve the data.

9. The data, if available, is returned to the CSC Service. For unsuccessful local cache lookups (missing blocks) a request is broadcast using [MS-PCCRD]. If data is found in a peer client, the data is retrieved using [MS-PCCRR].

10. The data is placed in the Branch Cache and delivered to the CSC driver.

11. The data is placed in the local CSC cache.

12. The data is returned to the Client Application.

13. If no hash list was available and the data was directly returned from the content server, then hash list generation is initiated on the content server.

14. The hash list may be independently created by running the hashgen utility on the content server.

15. The generated hash list is stored, making it available for future client requests.

5.2.4 PCCRD and WS-Discovery

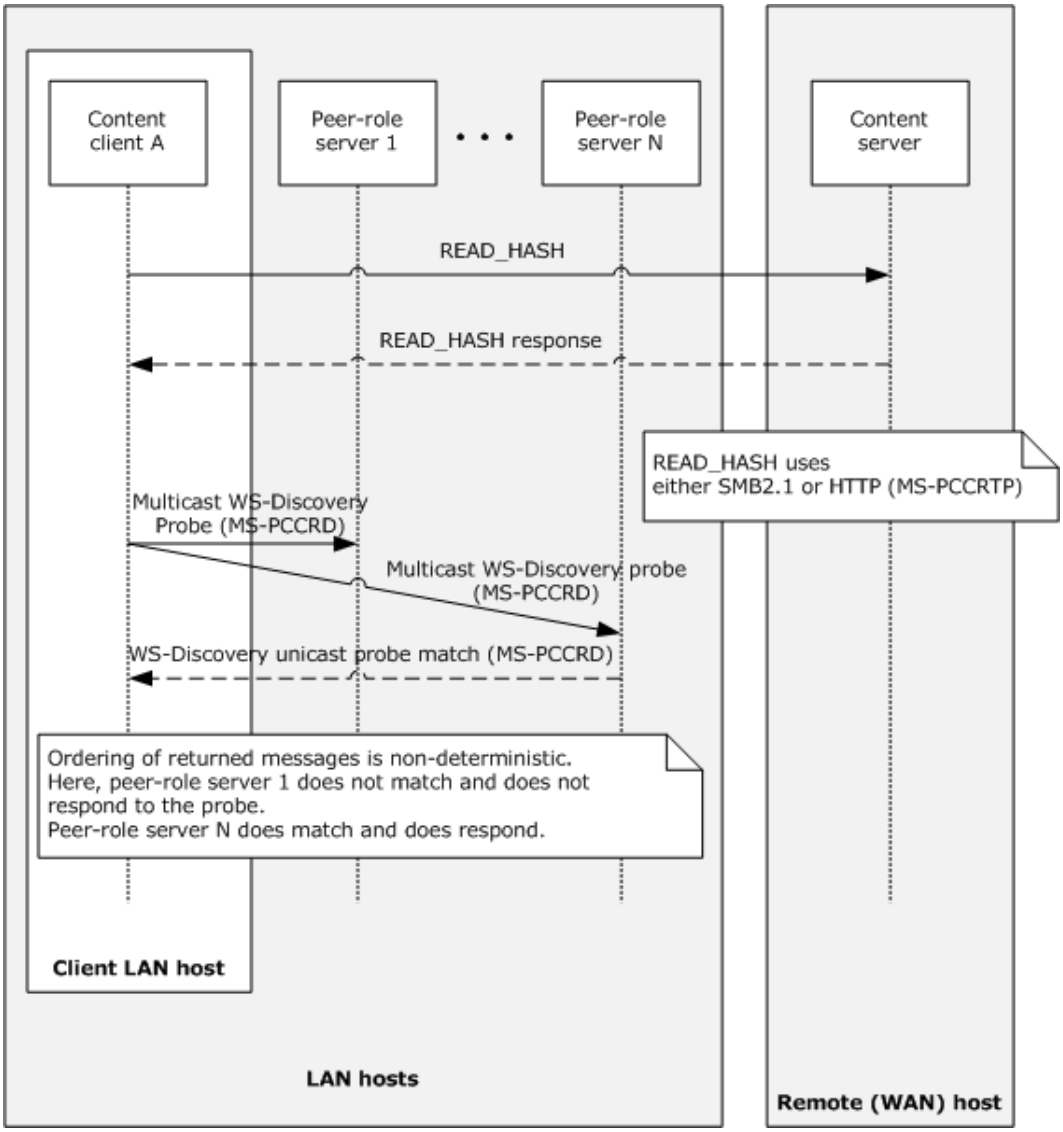

Figure 12: PCCRD content discovery

The figure in this section shows the ordering of messages when the service is operating in distributed cache mode. The initial request (READ_HASH) for the content takes place in either SMB2.1 or HTTP [MS-PCCRTP] (step 1). The response with the hashes comes back in the same protocol that makes the request (step 2). Once the content hashes have been received by the content client, it can determine whether any of the content exists within the LAN where it resides. The content client does this by checking any local machine cache, then if the data is not available locally performing a Multicast WSD/MS-PCCRD Probe message with the hashes of the content (step 3).

Any server-role peers on the LAN that receive the Multicast Probe message and have the matching content [MS-PCCRD] respond with a Unicast Probe-Match message (step 4).

After the content client receives a match for content, it will initiate one or more [MS-PCCRR] sessions (not shown in this figure) to retrieve the content from the server-role peer.

5.3 Member Protocol Functional Relationships

5.3.1 Member Protocol Roles

This section describes all member protocol roles.

5.3.1.1 Metadata (Hash) Retrieval

Windows currently has two protocols enabled for content metadata retrieval; the format for the metadata that is transferred is specified in [MS-PCCRC].

The 2.1 version of the SMB2 protocol has enhancements for the detection of content caching enabled shares and retrieval of metadata related to content caching. The role of SMB2.1 within the Content Caching and Retrieval System is primarily metadata retrieval.

[MS-PCCRTP] specifies a new content encoding, PeerDist, that can be used in HTTP/1.1. Like SMB2.1 above, the primary role of this protocol in the Content Caching and Retrieval System is content metadata retrieval. 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.

5.3.1.2 Content Identification

[MS-PCCRC] specifies the binary data structure used to uniquely identify content for discovery and retrieval purposes.

5.3.1.3 Content Discovery

[MS-PCCRD] specifies a protocol for use by a peer adopting the client role to discover content among other peers. It is based on the Web Service Dynamic Discovery Protocol (also called WSD) [WS-Discovery]. In the case of a BITS content client, it obsoletes [MS-BPDP] as a discovery mechanism.

5.3.1.4 Content Retrieval

[MS-PCCRR] specifies two protocol message exchanges - one for querying a server for the availability of certain content, and the other for retrieving content from a server. In the case of a BITS content client, this protocol renders [MS-BPCR] obsolete as a retrieval mechanism.

5.3.1.5 Content Offering

[MS-PCHC] specifies two protocol messages for exchange of metadata - one where a client informs a hosted cache that it has segments it can offer, the other where the client gives further information regarding the range of blocks within a segment.

5.3.2 Member Protocol Groups

This section provides additional information regarding the roles of each of the protocol groups. Member protocol groups are described in the following sections.

5.3.2.1 File Retrieval

SMB2 version 2.1 and HTTP ([MS-PCCRTP] specifies content encoding over HTTP) are the two protocols used for file retrieval that are content caching aware.

5.3.2.2 Content Discovery

[MS-PCCRD], an implementation of WS-Discovery, is used for content discovery when the system is acting in distributed mode.

When the system is operating in Hosted Cache mode, the content client will attempt to retrieve content for which it has obtained hashes directly from the hosted cache server. If this fails, then the content is retrieved from the content server with no further attempts to find cached content.

5.3.2.3 Content Retrieval

Content in the form of blocks is transferred using HTTP, with the message format specified in [MS-PCCRR].

5.3.2.4 Hosted Cache Meta Data Transfer

[MS-PCHC] over HTTPS is used by clients to offer metadata to a Hosted Cache server.

5.3.2.5 Authentication

Hosted Server Authentication uses HTTPS for secure transport of [MS-PCHC]. [MS-TLSP] describes the Windows implementation of TLS.

Client Authentication [RFC4559]. When a Hosted Cache server is used, client authentication can be enabled.

5.4 System Internal Architecture

5.4.1 Peer Content Caching and Retrieval Framework

A client participating in the Content Caching and Retrieval System acts as a peer to other clients in the system. A peer can be considered to be part of the Peer Content Caching and Retrieval Framework.

For the purposes of describing the Framework, a Hosted Cache server can be considered a peer.

A programmers API giving access to the Peer Distribution API is also available ([MSDN-PeerDISTAPI], [MSFT-BranchCache]).

5.4.1.1 Peer Details - Hosted Cache Mode

A Content Client-role peer in the Peer Content Caching and Retrieval Framework can work in a hosted cache mode. In this mode, the Client-role peer, do not use the Discovery Protocol to discover peer servers to retrieve content blocks. Instead, the content clients (peers) are configured with the address of a hosted cache server and query that server directly for the block ranges of the requested segments. The hosted cache server, in turn, is offered content by peers and retrieves the offered content blocks directly from the peer that offered them.

5.4.1.2 Peer Details - Cooperative Mode

A Content Client-role peer in the Peer Content Caching and Retrieval Framework can work in a distributed cache mode. In this mode, the Peer Content Caching and Retrieval Framework (the framework) is running on multiple peers. The peers use the Discovery Protocol [MS-PCCRD] and the Retrieval Protocol [MS-PCCRR] to allow the client-role peer to retrieve content blocks of a target segment from one or more server-role peers.

The framework accepts requests from the higher-layer applications on the client-role peer to retrieve the whole or parts of a content item, which may span multiple segments. For each target segment, the framework's client-role peer uses the Discovery Protocol to find server-role peers that have (the whole or parts) of the target segment. The client-role peer then initiates Retrieval Protocol [MS-PCCRR] exchanges to each server-role peer to query the block ranges each server-role peer has, and downloads the blocks.

The Discovery Protocol and the Retrieval Protocol both operate on or within a single segment. The operations specified in this section allow a client-role peer to find and retrieve blocks (parts or all) of a single target segment. This process is referred to as a segment retrieval session. If the content spans multiple segments, then the framework can use multiple segment retrieval sessions to retrieve all the content's segment and reassemble them into the complete content item.

5.4.1.3 Server Details

In addition to potentially acting as a client-role peer, the server side of the framework responds to client-role peer requests [MS-PCCRR], client-role probes for content [MS-PCCRD], and in the case of a hosted cache server, offers of content from client-role peers using [MS-PCHC].

The server maintains an Active Client Count as a mechanism for managing resources. See [MS-PCCRR] section 3.1.2.1. Any given server has finite resources and therefore will need to limit the number of simultaneous Upload Sessions that are possible.

If the service mode is distributed, then the server maintains a list of all HoHoDks for all available segments, and their corresponding block counts. See [MS-PCCRD] section 3.2.

If the service mode is hosted cache server, then the server will maintain content information for the offered segments and a block cache. See [MS-PCHC] section 3.1.1.

5.4.2 Content Caching Manager View

In the following diagram, solid lines indicate internal communication, and external protocols are indicated by annotations.

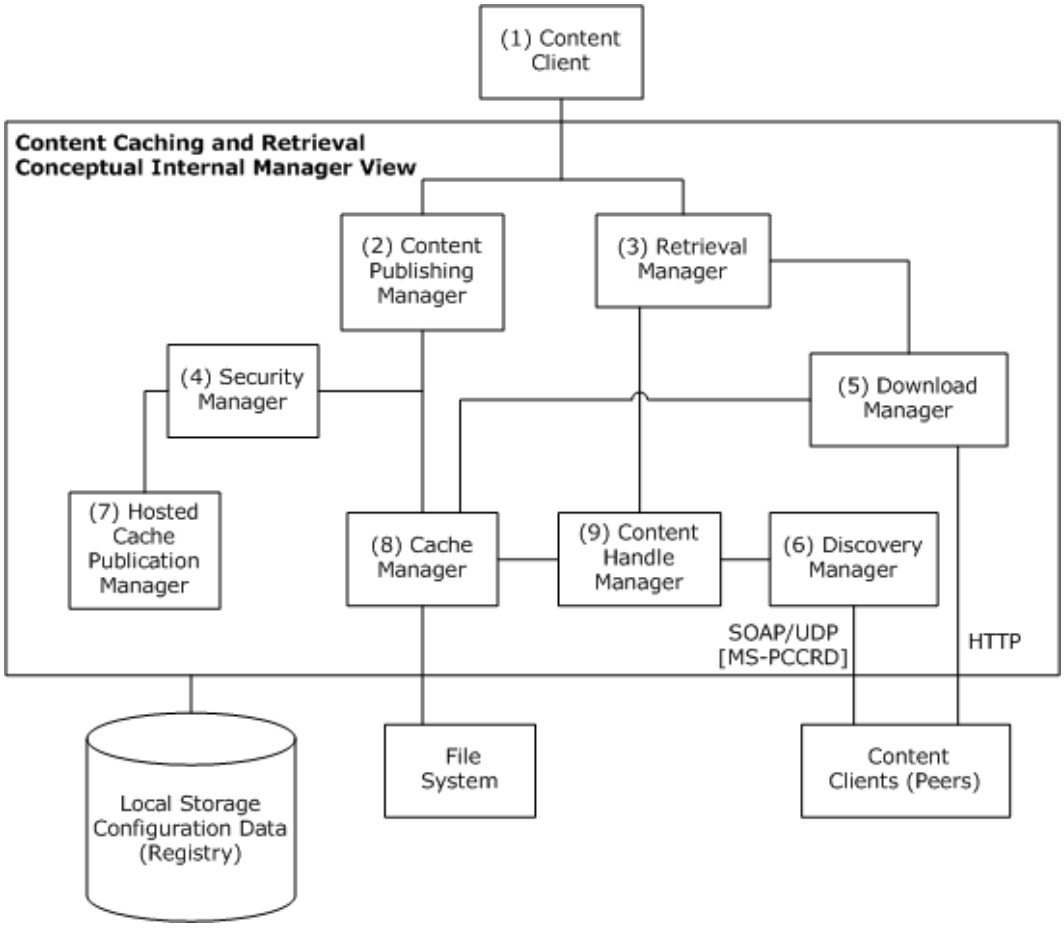

Figure 13: Content caching manager view

This section describes a conceptual management model where a series of managers participate in the Content Caching and Retrieval System.

5.4.2.1 Content Client (1)

The content client implements client-side protocol components and consumes the file services that are offered by a Content Server. The previous diagram shows a conceptual relationship and indicates internal communication rather than protocols.

5.4.2.2 Content Publishing Manager (2)

The Content Publishing Manager is invoked by a server application and performs the following tasks:

- Calls the Cache Manager to store the content in the publication cache.
- Calls the Security Manager to start computing hashes on the data.
- Retrieves sets of segment hashes associated with the content ID, by calling the Content Handle Manager.

5.4.2.3 Retrieval Manager (3)

The Retrieval Manager is invoked by a content client with a hashlist and a range of data and performs the following tasks:

- Checks if hashes are available in the re-publication cache.
- Calls the Content Handle Manager to see if any of the hashes have associated data in the cache. Note data is only available if it has been authorized by the security manager.
- When data is in cache, calls the cache manager to retrieve the requested byte range.
- When data is not in cache, calls the download manager to retrieve the data.
- For blocks not found with peers or the cache, informs content client that the hashes are not available.
- Receives blocks of data from a content client and writes them to the Cache Manager.

5.4.2.4 Security Manager (4)

The security manager performs the following tasks:

- Hash Generation and Validation.
- Proving the HoHoDk used as the key in WSD discovery messages. See [MS-PCCRC] section 2.2.
- Securing connections among peers by encrypting content data. See [MS-PCCRC] section 2.2.

5.4.2.5 Download Manager (5)

The Download Manager performs the following tasks when supplied with a hashlist and byte range from the retrieval manager:

- Calls the Discovery Manager to locate peers who have data corresponding to the segment hashes.
- If peers (which could be the Hosted Cache) publishing hash availability are found, then the Download Manager establishes connections with the various peers.
- As blocks are downloaded and placed in the re-publication cache, it calls the security manager to validate the data received against the hashes received from the Content Client; if valid, the blocks are marked as valid (available for use or to share with other Peers) within the cache. If data is not valid, the blocks are marked as missing within the cache. If not retrieved within a certain timeout, the data block is reported as missing, and the content client will need to fetch the block directly from the server.

5.4.2.6 Discovery Manager (6)

The Discovery Manager is responsible for providing a list of peers that can serve data identified by hashes. It does this by performing the following tasks:

- Uses MS-PCCRD to discover peers that can serve data segments.
- If Hosted Cache mode is in effect, the Discovery Manager returns the location of the hosted cache to the retrieval manager instead of the list of peers.

5.4.2.7 Content Handle Manager (9)

The Content Handle Manager performs the following tasks:

- When supplied a content ID, returns an object that can be used to get segment hashes associated with the content ID.
- When supplied a segment hash, returns an object that can be used to access the content in the local cache.
- When the Cache Manager provides a notification that a particular handle is no longer available (for example, it has expired due to Least Recently Used policy), the Content Handle Manager maps this into a segment hash and notifies the Discovery Manager that the segment hash should no longer be published.

5.4.2.8 Cache Manager (8)

The Cache Manager performs the following tasks:

- Manages the publication and re-publication cache. The publication cache holds actual content, and the re-publication cache holds hashes. The re-publication cache uses an algorithm to periodically remove cache entries based on Least Recently Used policy to make room for new content. The publication cache releases content only when the publication client explicitly un-publishes the content or disconnects ungracefully.
- Notifies the Content Handle Manager when a complete content block has been downloaded for a given segment hash, when a segment within the content is fully downloaded, when a segment has been removed from the cache, and when the cache becomes completely empty.

5.4.2.9 Hosted Cache Publication Manager (7)

The Hosted Cache Publication Manager (HCPM) peer client side performs the following tasks:

- Acts as a broker for content to be stored on a system that has been configured to run as a hosted cache server.
- As blocks of content are validated by the security manager, the HCPM may choose to offer content data, associated hashes, and other segment-level information to the hosted cache server. The location of a hosted cache server can be configured on the client using Group Policy. The offer of content data, associated hashes, and other segment-level information to the hosted cache server, as well as the actual transfer of the information, is done using a TLS-secured network connection.

The Hosted Cache Publication Manager (HCPM) server side performs the following tasks:

- Authenticating (using domain credentials or server authentication per configuration settings) offerings made by peer clients. If the offered content is needed (not already present) by the Hosted Cache, the HCPM receives, by secure transfer, the content description information for the segment to be uploaded.
- Once this information is present on the Hosted Cache, the HCPM interacts with the client offering the content just as a normal peer Client would interact with another peer and transfer the content data so that it may later be served to requesting clients.

5.5 Failure Scenarios

This section describes the common failure scenarios and specifies the system behavior in such conditions.

5.5.1 Connection Disconnected

A common failure scenario is an unexpected connection breakdown between the system and external entities. A disconnection can be caused by the network not being available, or by one of the communicating participants becoming unavailable. In the case where the network is not available, both participants remain active and expect the other party to continue the communication pattern specified by the protocol being executed at the time of the failure. Similarly, in the case where one of the participants is not available, the active participant expects the communication to proceed as specified by the protocol being executed.

Generally, a protocol detects a connection breakdown failure through either of the following methods:

- By using a timer object that generates an event if the corresponding participant has not responded within a reasonable time span.
- By being notified by the underlying protocol that the connection is disconnected.

When a connection disconnected event is detected, it causes the protocol to tear down all related communications and update any necessary data structures to maintain the system state.

Details about how each protocol detects a connection disconnected event, and how it behaves under this scenario, are provided in the specifications of the member protocols, as listed in section 2.2.

A content client failing to find content on a hosted cache server is not considered a common failure, but rather a normal operation. The first time any content client attempts to retrieve data from a hosted cache server, it will fail to find the content, and the Client will simply request the full content from the Content Server.

5.5.2 Internal Failures

The Content Caching and Retrieval System is dependent on the File Access Service System. That system is not defensive against internal failures of its state, other than that described in [MS-FSSO] and the specifications of its member protocols.

5.5.3 System Configuration Corruption or Unavailability

The Content Caching and Retrieval System relies on the availability and consistency of its configuration data. Configuration consists of the data that determines the behavior of the system under specific conditions or for specific functionality. In the event that the Content Caching and Retrieval System fails in some manner and the File Access Service System is still functioning, the system will operate without content caching.

During content retrieval malformed messages received by the content client and messages of unknown type are quietly discarded; see [MS-PCCRR] section 3.1.1.5.4.

6 System Details

This section contains the details that complete the descriptions in earlier sections of the document. These details are needed to understand and implement this system.

6.1 Architectural Details

Content caching enables content from file and Web servers on the local end of a wide area network (WAN) to be cached on computers at the remote end of the WAN. The cached content can be distributed across client computers in one of two modes (distributed cache mode) or centrally hosted on a server (hosted cache mode).

When caching is enabled (identified per share for SMB2.1 and per site for HTTP) and hashes are available, a copy of the content that is retrieved from the content server (using HTTP/HTTPS or SMB2.1) is cached at the remote end of the WAN. Usually this means storing the content on one or more computers on a LAN. If another client at the remote end of the WAN requests the same content, that client can download it directly from a local cached location without needing to retrieve the content by using the WAN.

6.1.1 Reading a file using SMB2.1 as Metadata Channel

This example illustrates use cases (3.3.5.1 Read a file using SMB2.1 Metadata Retrieval with Hosted Cache). The SMB2.1 protocol has 2 distinct roles to play. The first role is to act as a member of the File Access System providing the usual file sharing resources between machines. The second role is to act as a transport for metadata from a content server to a Content Client, thereby allowing the content client to participate in the Content Caching and Retrieval System.

The initial stages of the read operation from an SMB2.1.1 protocol operation perspective are the same regardless of the caching mode (Hosted, Distributed or none). After hashes have been identified as available (SMB2.1 is required for this) then, the retrieval mechanism will vary depending on the mode of caching in effect. The caching mode is configurable, either manually or through the use of group policy. The following description has an SMB2.1 stage and then a specific stage for Hosted Cache Mode and distributed cache Mode.

The basic scenario is that two users within a LAN will obtain content by reading in turn a file from a content server that is reached over a network link with a latency value greater than the value of the ADM element "Network Latency". When the first user reads the file, there will be no cached content available within the LAN, and therefore the file will need to be retrieved using an SMB2.1 read operation as specified in [MS-FSSO] and [MS-SMB2]. The Content Caching and Retrieval System does not change the requirements related to authentication and access.

After at least 32 Mb (1MB (one whole Segment) of content has been retrieved for the first time and is available on the LAN, then there is potential for the content to be cached and made available to other users (peers) on the LAN.

At this stage the mode of caching becomes pertinent.

If Hosted Cache Mode is in effect, content will be offered by the content client to the hosted cache server using [MS-PCHC]. Depending on the current state of the hosted cache server with regards to the content, one or more [MS-PCCRR] sessions may be initiated by the hosted cache server to transfer any required content from the offering Content Client.

If distributed cache Mode is in effect, no immediate action is taken.

6.1.1.1 SMB2.1 Initial Stages of Content Caching and Retrieval

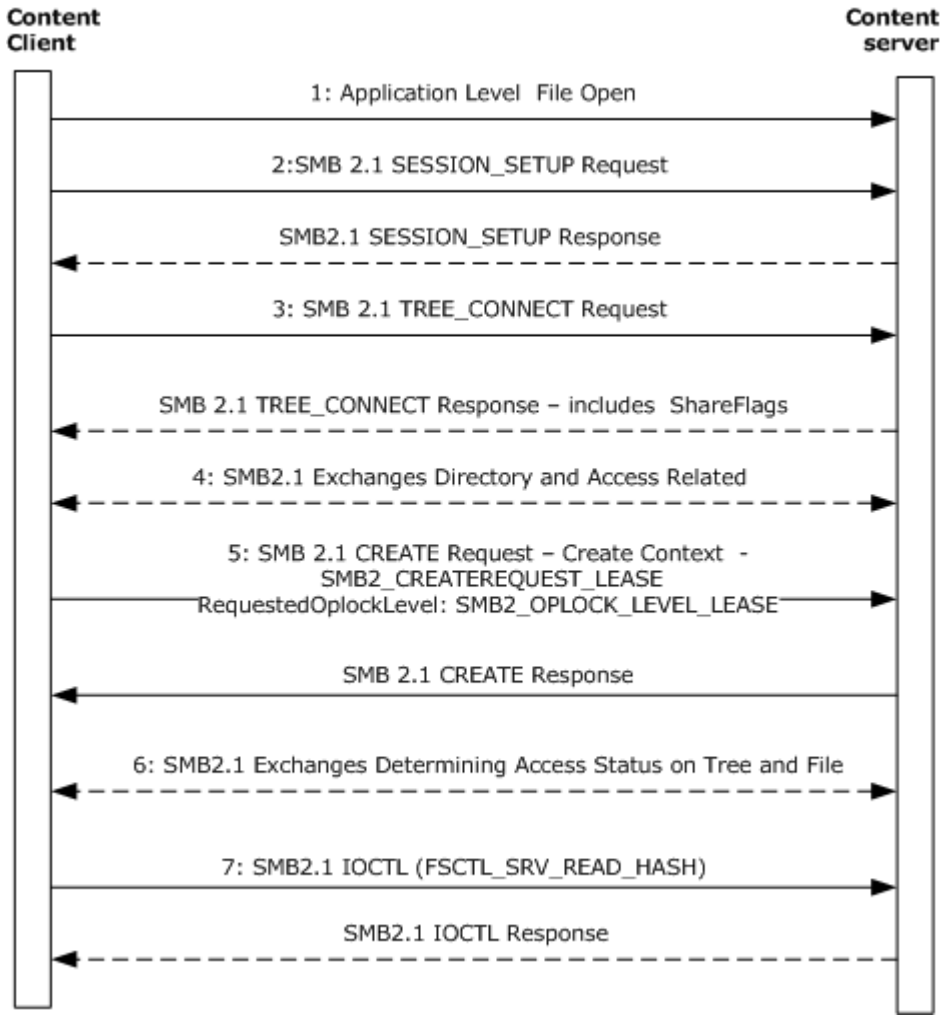

Figure 14: SMB2.1 Stages - Read Operation Summary sequence of Events

The example begins with the normal sequence of events that would occur when reading a file using the SMB2.1 protocol. The specification for the SMB2.1 protocol can be found in [MS-SMB2], and a description of that protocol's interaction with other protocols can be found in [MS-FSSO], the Protocol Family System Overview for File Access Services System. A full elaboration of opening a file using SMB2.1 can be found in [MS-FSSO] section 6.1.5.13.

The initial sequence is an open operation followed by a SESSION_SETUP and **TREE_CONNECT**. An SMB2.1 TREE_CONNECT operation is required to access any share on a SMB2.1 server. The activities up to this point are independent of the Content Caching and Retrieval System and are shown in the protocol exchange in steps D1 to D3 of the SMB2.1 read file common stages figure, and in steps 1-3 of the Read Operation Summary sequence of Events figure.

It is only after a response has been obtained to the TREE_CONNECT does that a content client gets an indication that the share has the capability to participate in the Content Caching and Retrieval System.

The SMB2.1 TREE_CONNECT Response ([MS-SMB2] section 2.2.10) indicates in the returned ShareFlags field whether the share supports hash generation for content cache retrieval of data. The value SHI1005_FLAGS_ENABLE_HASH 0x00002000 indicates that caching is enabled on the server share and that the server can respond to requests for metadata (subject to the service being installed and configured). The flag informs the content client that it can make requests for metadata, but does not guarantee that it will be available.

If content cache retrieval is enabled on the share and the content client wishes to use cache data, the content client MUST attempt to obtain a lease that allows it to cache reads and writes to the data (see [MS-SMB2] section 3.2.4.3.8,"Requesting a Lease on a File").

A lease is obtained as part of the SMB2.1 CREATE request. The lease is requested as part of the CREATE operation with a RequestedOplockLevel of SMB2_OPLOCK_LEVEL_LEASE. Only if this lease is granted by the server is it safe for the content client to use cached data and therefore attempt to look up the data in the cache.

The protocol exchange is shown in steps D3 to D4 of the SMB2.1 read file common stages figure, and in step 5 of the Read Operation Summary sequence of Events figure.

If content caching retrieval is enabled on the server share and a lease has been obtained, then the SMB2.1 content client performs a SRV_READ_HASH Request (see [MS-SMB2] section 2.2.31.2) on the file and tries to obtain the hash (**metadata**) for the target file. If the hash is out-of-date, an error is returned to the client; if no hash exists for the specified file, an error is returned to the client.<14> A server can choose to update the hash for either of these situations.

The protocol exchange is shown in steps D5 to D6 of the SMB2.1 read file common stages figure, and in step 7 of the Read Operation Summary sequence of Events figure.

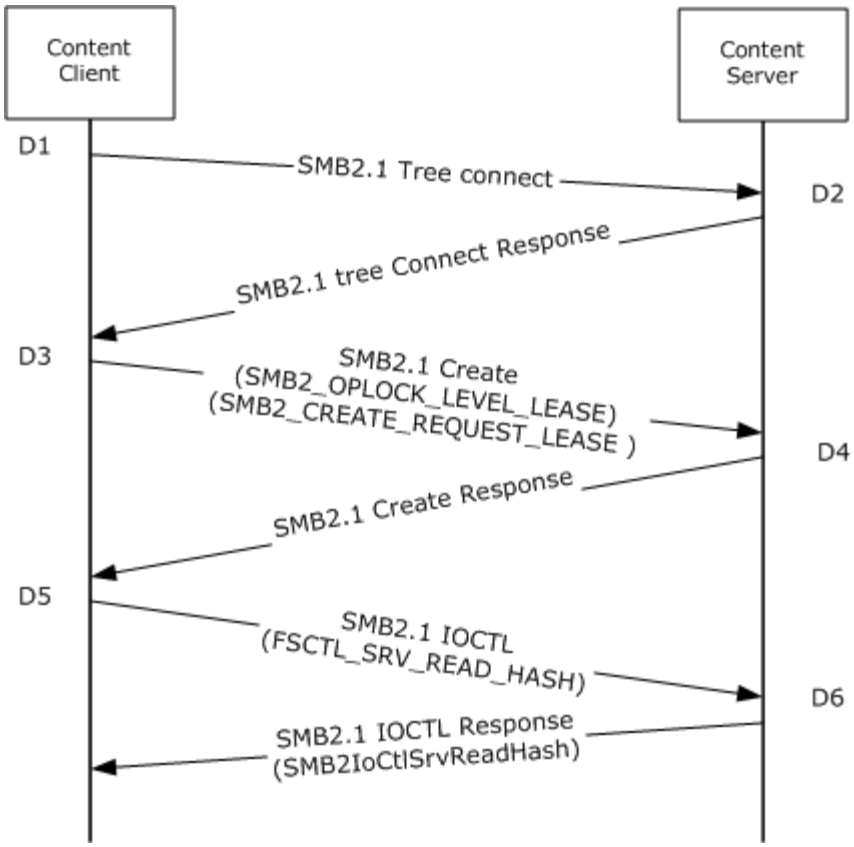

Figure 15: SMB2.1 read file common stages

After the initial SMB2.1-specific exchanges have been performed, the sequence of activity becomes dependent on the caching mode that is in effect. If the caching mode is Hosted Cache, see section 6.1.3 for subsequent details; if the caching mode is distributed cache, see section 6.1.4 for subsequent details.

6.1.2 Reading a File Using HTTP as the Metadata Channel

The sequence described in this example illustrates detailed use case 3.3.5.5 The details show how the Application uses the content client to open a file on the content server using the HTTP protocol. The HTTP protocol has two distinct roles to play. The first role is to act in the usual manner as a file retrieval protocol. The second role is through the use of PeerDist [MS-PCCRTP] to act as a transport for metadata from a content server to a Content Client, thereby allowing the content client to participate in the Initial Stages of Content Caching and Retrieval System.

6.1.2.1 HTTP Metadata Retrieval

The content client sends an HTTP GET request to the Content Server. The client indicates in the HTTP request header that it is PeerDist-enabled by listing PeerDist as an accepted encoding as specified in [MS-PCCRTP] (see M1 in the Hosted Cache – content retrieval figure).

In the (M2) HTTP 200 response, the content server indicates that the content is encoded using the PeerDist content encoding. It also includes the content length of the entity-body, and the content length indicates the length of the metadata. All malformed messages received by the server and

messages of unknown types sent to the Retrieval Protocol MUST be silently discarded (see [MS-PCCRR] section 3.1.2.5.4).

6.1.3 Reading a File Hosted Cache Stage

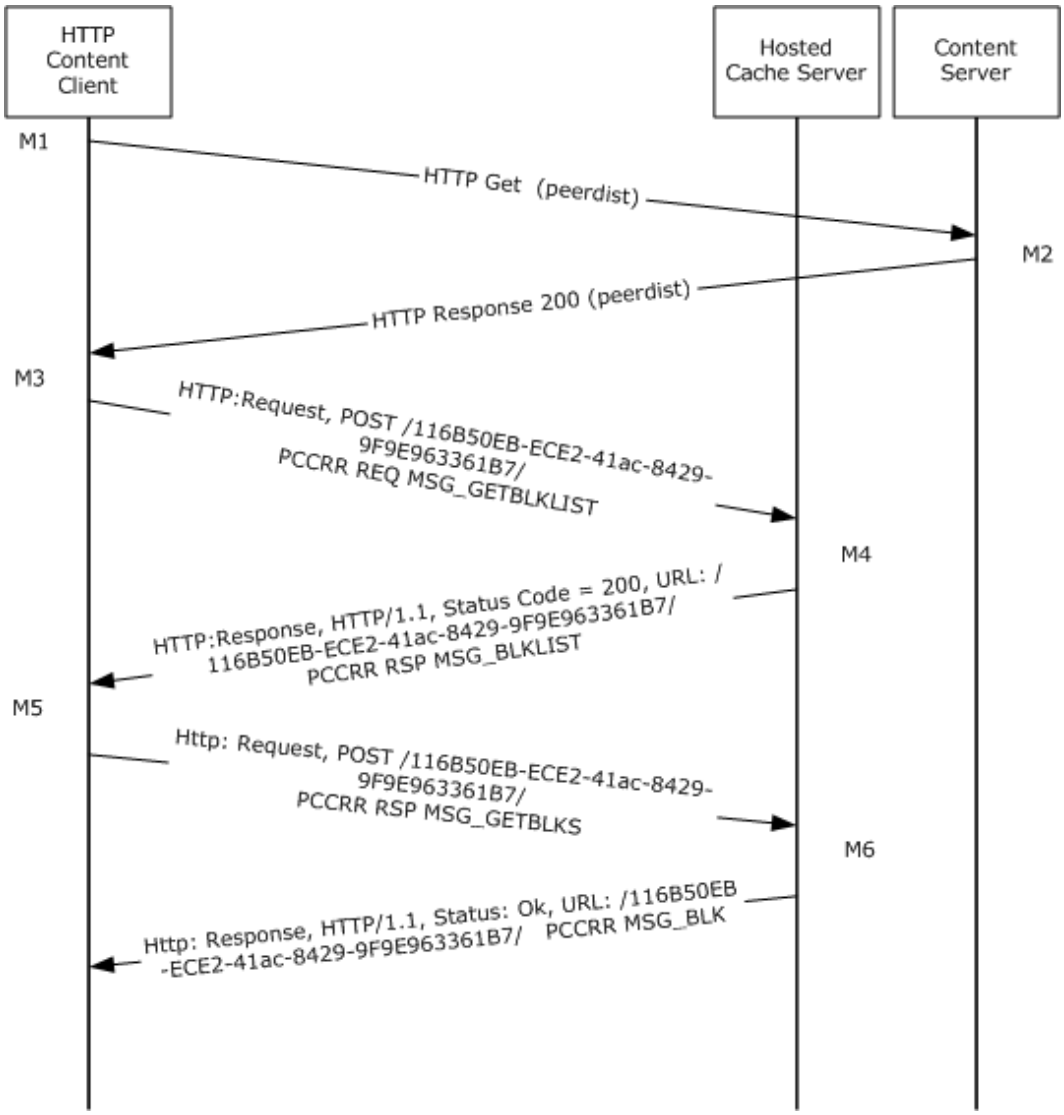

Figure 16: Hosted Cache – content retrieval

6.1.3.1 Content Retrieval

When Hosted Cache mode is configured, no discovery of peers occurs, and the content client is preconfigured with the FQDN of the Hosted Cache server. The content client initiates the transport with the hosted cache server (M3 in the figure in section 6.1.3) by sending an HTTP POST request to the root path of {116B50EB-ECE2-41ac-8429-9F9E963361B7/} of the Server. That request also carries a [MS-PCCRR] REQUEST-MESSAGE (MSG_GETBLKLIST) (carried as an entity body of the HTTP POST) to the hosted cache server requesting the block IDs of the blocks within the target segment that the content client is interested in.

The hosted cache server responds (M4 in the figure in section 6.1.3) with an HTTP status code of 200 (OK) for the URL /116B50EB-ECE2-41ac-8429-9F9E963361B7/ and a [MS-PCCRR] RESPONSE-MESSAGE (MSG_BLKLIST) indicating the blocks currently available for download from the hosted cache server.

The content client sends (M5 in the figure in section 6.1.3) a [MS-PCCRR]) REQUEST-MESSAGE (MSG_GETBLKS) to the hosted cache server.

The hosted cache server responds with [MS-PCCRR] RESPONSE-MESSAGE (MSG_BLK) (M6 in the figure in section 6.1.3).

The steps M3, M4 followed by M5, and M6 are repeated until the required content is transferred.

6.1.3.2 Content Offering, No Client Authentication

Figure 17: Content offering to hosted cache server, no client authentication

This section presents an example of a hosted cache server that has no block hashes associated with a segment being offered by a Content Client. The hosted cache server has been configured to not require client authentication (see section 6.1.3.3 for details about Client Authentication).

In the sequence shown, on availability of new blocks for a segment, the content client uses the [MS-PCHC] protocol to offer the associated segment to the hosted cache server. The hosted cache server determines that it has no block hashes, and therefore requests that the content client send it complete information on the segment, so that the hosted cache server can then use [MS-PCCRR] to retrieve the desired blocks from the Content Client.

The communication between the hosted cache server and content client is secured by using the HTTPS protocol. The initial SSL handshake is not discussed in this section, only the traffic that is

pertinent to Content Caching and Retrieval. The URL on which the hosted cache server listens is normally "https://:<port number>/C574AC30-5794-4AEE-B1BB-6651C5315029/". The port number is configurable, but would normally be 443 for HTTPS. If the hosted cache server is configured to something other than port 443, then Content Clients also need to be configured to use that port.

Content offering <15>begins with the content client sending a request message as the payload of an HTTP POST request to the Hosted Server. The message is a PCHC initial offer of content to the Hosted Cache; see messages M1 in the preceding figure ([MS-PCHC] section 2.2).

The hosted cache server sends the response message as a payload of the HTTP response. The transfer encoding is of typed chunked (see [RFC2616] sections 3.6.1 and 19.4.6). The hosted cache server indicates in the response whether it is interested in the content or not (M2). Interested means that the Hosted Cache server needs the range of block hashes from the content client before it can download content.

In this case, the hosted cache server is interested because it has no block hashes, and indicates this to the content client in the PCHC message. The Content Client responds (M3) by sending a SEGMENT_INFO_MESSAGE message. This message contains the segment hash of data (HoD) for the previously offered segment, as well as the range of block hashes in the segment. An example of **hash data** is shown below.

After the hosted cache server obtains the SEGMENT_INFO_MESSAGE, it responds with an HTTP 200 OK response (M4). The actual data is then retrieved from the offering peer by [MS-PCCRR].

The M5 message shows the initial sequence of [MS-PCCRR], with the hosted cache server requesting the data from the Content Client.

```
+ Ethernet: Etype = IPv6,DestinationAddress:[00-15-5D-D4-8A-B2],SourceAddress:[00/15-5D-
D4-8A-B4]
+ Ipv6: Next protocol = TCP, Payload Length = 1274
+ Tcp: Flags=...AP..., SrcPort=60633, DstPort=7000, PayloadLen=1254, Seq=2059702016 -
2059703270, Ack=982804
+ Http: HTTP Payload, URL: /C574AC30-5794-4AEE-B1BB-6651C5315-029/
- Pchc: Request:The message is a Segment Info Message.
  - PCHCRequest: The message is a Segment Info Message.
    - SegmentInfoMessage:
      + MessageHeader: Version: 1.0, Type: 0x2:The message is a Segment Info Message.
      + ConnectionInformation: Port: 80
        PEERDIST_CONTENT_TAG: 0x35db045d14234553a0510dc2e15e6c4c
(0x0x35db045d14234553a0510dc2e15e6c4c)
      - ContentInformation: Contains 1 Segments
        + Version: 1.0
          dwHashAlgo: SHA-256 hash algorithm
          dwOffsetInFirstSegment: 0 (0x0)
          dwReadBytesInLastSegment: 2230784 (0x220A00)
          cSegments: 1 (0x1).
        - PCCRCSegmentDescription: The 1 Segment
           ullOffsetInContent: 0 (0x0)
           cbSegment: 2230784 (0x220a00)
           cbBlockSize: 65536 (0x10000)
         Hash_SegmentHashOfData:
0xabf7be4ae77fc990b85040ae5fda50a6421d8d918fd7ed034de66c08894b9637
           Hash_SegmentSecret:
0x3384f7b6c7c970a9763f8ed89722bd7fe56a604e8c6de5b03b194a2729172aaa
        - PCCRCSegmentContestBlocks: Contains 35 Blocks
           cBlocks: 35 (0x23)
           HashData: 0x96176beb1b32f47ecff28fba91c0911db99f192b02110239793c7a06be0367c3
           HashData: 0x5a0579ccbd50806841519f237427deb71d1573d63e5099834fad31c93740abd6
           HashData: 0x28ed31da1b55fce18800dc48e4093f4d4f253c4b0e81df4ad82cb49aedead189
           HashData: 0x05e8e50b7e89ac50a4d87465d662dd4552e2202e96f29f33f2fb0fd8a1b111ff
           HashData: 0x416e7db6f15135d0377381d8742b8e82d765e5374b1e66b4f9b5faface6059af
           HashData: 0xf8adf491f03629247a51cbc5335622d5aec04333bbdca446210a9ec0a7b3356d
```

Figure 18: Example of a PCHC segment info message

6.1.3.3 Content Offering with Client Authentication

A hosted cache server can be configured to require client authentication. The following figure shows the sequence of messages with this configuration in effect, and the details are described as follows.

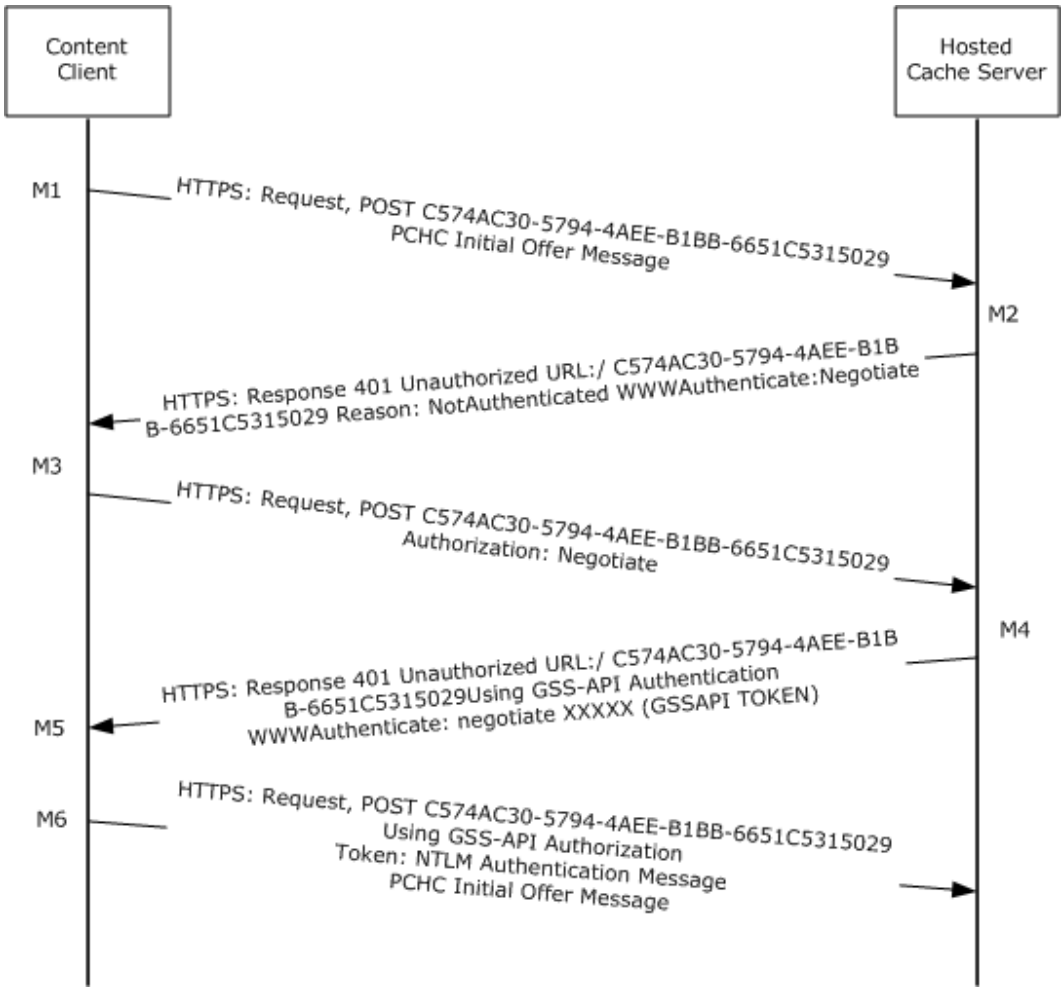

Figure 19: Hosted cache content offering with client authentication

The communication between the hosted cache server and content client is secured by using the HTTPS protocol. The initial SSL handshake is not discussed,; the only the traffic shown is that pertinent to Content Caching and Retrieval. The URL on which the hosted cache server listens is normally https://:<port number>/C574AC30-5794-4AEE-B1BB-6651C5315029/. The port number is configurable but would normally be 443 for HTTPS. If the hosted cache server is configured to something other than port 443, Content Clients also need to be configured to use that port.

Content offering begins with the content client sending a request message as the payload of an HTTP POST request to the Hosted Server. The message is a PCHC initial offer of content to the Hosted Cache; see message M1 in the previous figure ([MS-PCHC] section 2.2). When client authentication is in effect, the hosted cache server responds with an HTTP 401, Unauthorized. (see message M2 in the previous figure). The HTTP response is as follows.

```
Http: Response, HTTP/1.1, Status: Unauthorized, URL: /C574AC30-5794-4AEE-B1BB-6651C5315029/,
Using Negotiate
Date: Authentication
  ProtocolVersion: HTTP/1.1
  StatusCode: 401, Unauthorized
```

```
 Reason: NotAuthenticated
+ ContentType: text/html
 Server: Microsoft-HTTPAPI/2.0
- WWWAuthenticate: Negotiate
```

Message M2 in the previous figure is an example of such a message.

The authentication of Content Clients uses the mechanisms described in [RFC4559], which are based on GSS-API [RFC2743].

The exact number of packets exchanged during client authentication is dependent on the authentication mechanism <16>and whether the authentication is successful. In any case, if authentication fails for any reason the content client will not be able to offer content to the hosted cache server.

The Content Client, on receipt of the Unauthorized HTTP response, is expected to repeat the previous POST message, passing an HTTP Authorization header line (see message M3 in the previous figure). In the example shown in the Hosted cache content offering with client authentication figure, the content client is configured with the IPv4 address of the Hosted Server. In this situation, NTLMSSP is offered as the authentication mechanism. Note that it is the credentials of the computer (Content Client) that are authorized and offered during the NTLM challenge response exchange. (See message M6 in the previous figure).

After authentication, message processing proceeds in a similar manner to the non-authenticated client described in section 6.1.3.2 namely, the PCHC initial offer of content is made to the Hosted Cache server, which then responds depending on whether it needs the content or not.

6.1.4 Reading a File Distributed Cache Stage

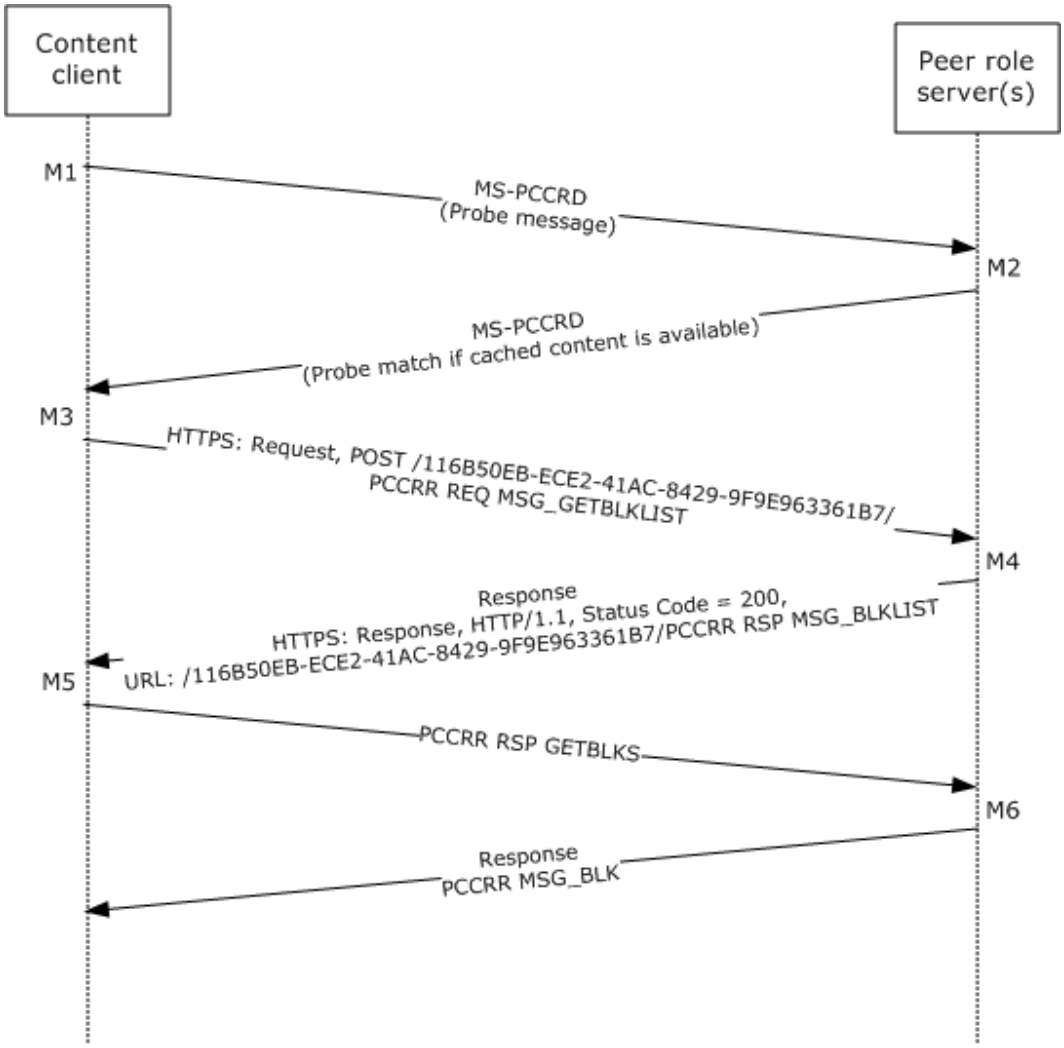

Figure 20: Distributed mode discovery and retrieval

6.1.4.1 Content Retrieval

When distributed cache mode is configured a content client on receipt of content metadata needs to "discover" whether peers have any of the Content that the content client requires. Unlike the Hosted Cache mode where Content Clients are pre-configure with the FQDN of the hosted cache server. Discovery is done using [MS-PCCRD] a protocol based on the Web service dynamic discovery protocol. Discover involves a Unicast broadcast using segment identifiers to discover whether there are any peers on the local subnet that possess the required content. The process of discovery and retrieval is as follows:

1. On successfully obtaining a file hash, the client calculates the segment identifiers (HoHoDk) see 2.2 [MS-PCCRC] for the required content.

2. The content client performs a multicast broadcasts probe message for the required content see 2.2.1 of [MS-PCCRD] (see the Distributed Cache mode discovery and retrieval figure, message M1). The broadcast is targeted at any peers listening on the local subnet.

3. If any peers have the required content, they respond with a Unicast Probe-Match Message (see section 2.2.2.1 [MS-PCCRD]). The Probe-Match Message includes segment identifiers and the address of the peer that holds the content.

4. If no Probe-Match Messages are received the full data will be retrieved from the content server using the originating protocol (SMB2.1 or HTTP)

5. If the client receives a Probe-Match message it verifies that the response was sent by a peer on the local subnet, and checks the HoHoDks to verify that at least one is associated with a Probe message on the Outstanding Probe List. [MS-PCCRD]

6. The client then retrieves data by initiating the transport with the peer by sending an HTTP POST request to the root path of {116B50EB-ECE2-41ac-8429-9F9E963361B7/} of the peer with the data [MS-PCCRR]. (See the Distributed Cache mode discovery and retrieval figure, note that this HTTP session is completely independent of any session that may have been initiated to retrieve content from a Content Server. The initial HTTP POST carries a [MS-PCCRR] REQUEST-MESSAGE (MSG_GETBLKLIST) (carried as an entity-body of the HTTP POST). This is a request for a download of a block list.

7. The peer responds with a HTTP status code of 200 (OK) for the URL /116B50EB-ECE2-41ac-8429-9F9E963361B7/ [MS-PCCRR] (see the Distributed Cache mode discovery and retrieval figure, the response carries a [MS-PCCRR] RESPONSE-MESSAGE (MSG_BLKLIST) indicating the blocks currently available for download from the peer.

8. The content client then sends a number of [MS-PCCRR] REQUEST_MESSAGE (GETBLKS) to the Server-role peer to retrieve all the required data.

9. The Server-role peer responds with [MS-PCCRR] RESPONSE_MESSAGE(MSG_BLK).

Step M5 and M6 are repeated until all the required blocks within the identified segment have been retrieved. If multiple segments are required then multiple segment retrieval sessions may be initiated (steps M3-M6) with one or more Server-role peers.

6.1.5 Reading a File Using BITS with Content Caching

This example illustrates detailed use case 3.3.5.7 reading a file using the HTTP protocol, with the transfer initiated by the use of BITS [MC-BUP]. (see Summary Use Case described in 3.3.4.5

6.1.5.1 BITS Initial Stages of Content Caching and Retrieval

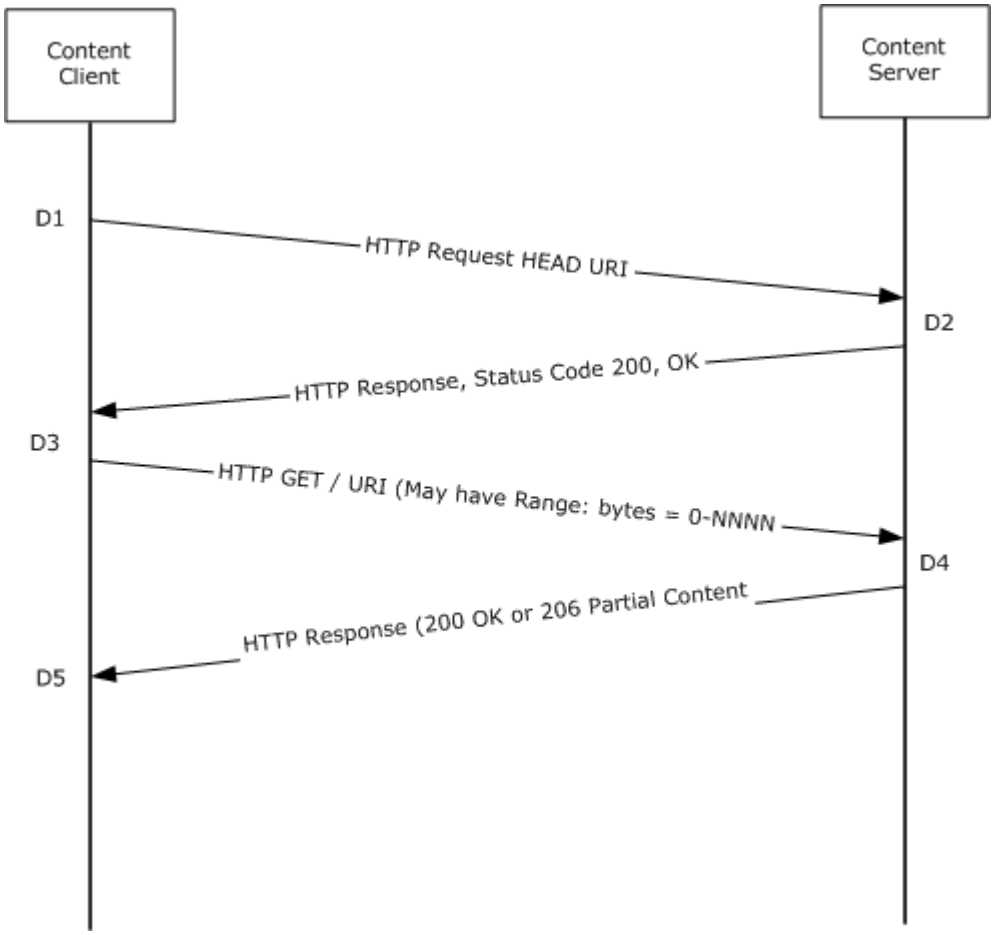

Figure 21: Reading a File BITS common stages

The application invokes the BITS Content Client (D1 in the preceding figure), requesting a handle to a file on the Content Server. This results in an HTTP HEAD command being transmitted by the content client to the content server (see [RFC2616] section 9.4). The HEAD method is identical to the GET method except that the server does not return a message- body in the response. When a BITS content client transmits the HEAD method, the UserAgent field is indicated as Microsoft BITS/ followed by a Major.Minor version number.

The server responds with an HTTP 200 message (D2 in the preceding figure) and indicates the Content-Length ([RFC2616] sections 14.13 and 14.19, respectively).

The content client sends an HTTP GET request to the content server (D3 in the preceding figure). The client indicates in the HTTP request header that it is PeerDist-enabled by listing PeerDist as an accepted encoding as specified in [MS-PCCRTP], and it also indicates "identity" encoding (see [RFC2616] section 14.3). The request header also includes an "If-Modified-Since" (see [RFC2616] section 14.25). The request header may include byte ranges to control the flow.

In the HTTP response, the content server indicates that the content is encoded using the PeerDist content encoding. See [MS-PCCRTP] section 3.2.5.1 for details of the response. If range control has been specified, then the response may be a status code of 206 indicating partial content (D3 in the

preceding figure). The response carries within the payload a PCCRTP Body field (see [RFC2616] section 4) that contains Content Information Data structures as described in [MS-PCCRC] section 2.3.

6.2 Communication Details

Protocol layering was described in section 5.2

The Content Caching and Retrieval System does not define any communication constraints or additional message types beyond those described in the specifications of the member protocols listed in section 2.2.

The Content Caching and Retrieval System supports the SMB 2.1 and HTTP 1.1 protocols. Applications do not need to directly communicate with the system although they can if they need to. <17> However, applications accessing SMB2.1 and HTTP interfaces in the Windows 7 and Windows Server 2008 R2 operating systems transparently benefit from the Content Caching and Retrieval System when it is enabled.

During content retrieval, if the Content is made up of multiple segments, the content client must manage that retrieval and concatenate blocks into segments to reconstruct the original content. The unit of retrieval is the block.

6.3 Transport Requirements

The Content Caching and Retrieval System does not define any transports or restrictions on transports beyond those described in the specifications of the member protocols.

The [MS-PCCRTP] transport is HTTP [RFC2616].

[MS-PCCRD] is an implementation of [WS-Discovery], which is transported by [MSDN-SOAPoverUDP].

The [MS-PCHC] transport is HTTP over Transport Layer Security [RFC2818]

The [MS-PCCRR] transport is HTTP [RFC2616].

[MS-SMB2]: The **Server Message Block (SMB)** Version 2.1 Protocol is transported either directly over TCP or over NetBIOS over TCP.

The [MC-BUP] transport is HTTP [RFC2616].

6.4 Timers

This section explains the timers that are significant to the state of the entire system. The system is dealt with in terms of a Client Framework that describes the peer-role and server-role peers. It also includes a summary of the timers for each member protocol.

6.4.1 Member Protocol Timer Summary

Member Protocol	Timer Summary
[MS-PCCRTP]	None
[MS-PCCRD]	Two timers are associated with the Discovery Protocol operations, back off and request. See [MS-PCCRD] section 3.1.2 [MS-PCCRD].

Member Protocol	Timer Summary
[MS-PCHC]	None
[MS-PCCRR]	Associated with the request timer and upload timer. See [MS-PCCRR] section 3.1.2.

6.4.2 Client Framework

6.4.2.1 Hosted Cache Mode

The following timers are associated with the client framework operations:

- **Download Timer**: The **Download Timer** MUST be set by the higher-layer applications in the client at the beginning of each segment retrieval session.

  When the **Download Timer** expires, the client MUST abort the segment retrieval session.

6.4.2.2 Distributed Cache Mode

The following timers are associated with the client framework operations:

- Download Timer:

MUST be set by the higher-layer applications in the client at the beginning of each segment retrieval session. The segment retrieval session MUST abort when the Download Timer expires. <18>

- Server List Timer:

A separate instance of this timer MUST be started for each empty **Server Information List** created, and for each previously populated **Server Information List** that becomes empty.

It MUST be disabled if the **Server Information List** becomes non-empty.

An empty **Server Information List** MUST be removed when its Server List Timer expires. The default timeout value MUST be set to 15 seconds. <19>

When the **Server List Timer** for a **Server Information List** of a segment expires, the list MUST be deleted.

- Server Entry Timer: The timer for each server entry in a Server Information List.

The timer for each server entry in a **Server Information List**.

It MUST be started after an entry is added into the **Server Information List**. An entry MUST be removed from the list after its timer expires. The default timeout value MUST be set to 15 seconds. <20>

When the **Server Entry Timer** for an entry in a segment ID's **Server Information List** expires, the entry MUST be deleted from the cache.

6.5 Non-Timer Events

This section explains the non-timers events that are significant to the state of the entire system. The system is dealt with in terms of a Client Framework that describes the peer-role and server-role peers. It also includes a summary of the non-timers events for each member protocol.

6.5.1 Member Protocol Non-Timer Events Summary

Member Protocol	Non-Timer Event Summary
[MS-PCCRTP]	None
[MS-PCCRD]	Receive Probe, Receive Probe-Match; see [MS-PCCRD] section 3.1.4.
[MS-PCHC]	None
[MS-PCCRR]	GetBlockList Initiation (see section 3.1.4.1 of [MS-PCCRR]) GetBlocks Initiation (see section 3.1.4.2 of [MS-PCCRR])

6.5.2 Client Framework - Hosted Cache Mode, Higher-Layer Triggered Events

6.5.2.1 Content Retrieval Request

When the client instance of the framework receives a Content Retrieval Request from a higher-layer application, it adds the list of segments and the corresponding block ranges for each segment to its **Content Cache**. The framework then initiates a segment retrieval session for each segment in the requested content.

6.5.2.2 Segment Retrieval Session Initiation

A client in Hosted Cache mode (either a peer downloading content from a hosted cache, or vice versa) MUST perform the following actions when a segment retrieval session is initiated:

- The client MUST start the **Download Timer** for that segment retrieval session.
- Add the server into the **Server Information List** of the segment ID, and set the corresponding server status as "free".
- If the requested block ranges in the segment consist of three or fewer consecutive blocks, the client MUST:
  - Start a Download Schedule Session, if the Download Initiated Flag has not been set.
- Otherwise, initiate a Retrieval Protocol GetBlockList request (MSG_GETBLKLIST) to the server.

6.5.3 Client Framework - Distributed Cache Mode, Higher-Layer Triggered Events

6.5.3.1 Content Retrieval Request

When the client instance of the framework receives a content retrieval request from a higher-layer application, it receives a list of segments and block ranges for each segment. The framework concurrently initiates a segment retrieval session for each segment in the requested content.

6.5.3.2 Segment Retrieval Session Initiation

The client MUST perform the following actions in the order specified when a segment retrieval session is initiated for a segment ID:

1. Start a Download Timer for that segment retrieval session.
2. Check the status of the Server Information List of the segment ID:

1. If the Server Information List for the segment ID exists and is empty, then the client MUST abort the segment retrieval session.

2. If the Server Information List for the segment ID exists and is not empty, then the client MUST:

    1. Clear the available block ranges for every server in the list, and set each server status as "free".

    2. If the requested block ranges of the segment consist of three or fewer consecutive blocks, the client MUST set the Download Initiated Flag (see section 5.1) and start a **download schedule session**

    3. Otherwise, the client MUST initiate a Retrieval Protocol GetBlockList request to each server in the list.

3. Otherwise (the Server Information List does not exist), the client MUST:

    1. Create a Server Information List for this segment ID.

    2. Start the Server List Timer for the list.

    3. If the maximum number of Server Information Lists has been reached, the client MUST delete the least recently used Server Information List.

    4. If the Discovery Frequency Counter is greater than or equal to the threshold, the client MUST abort the segment retrieval session. Otherwise, the client MUST start a Discovery Protocol instance (as specified in [MS-PCCRD]), passing it the segment ID.

6.5.4 Client Framework - Hosted Cache Mode, Other Local Events

6.5.4.1 Download Schedule Session

When a Download Schedule Session is started, then:

- If all requested blocks are marked as "downloaded", then the client MUST return all blocks in the requested block ranges and exit the segment retrieval session.
- If none of the requested blocks in the segment is marked as "idle", the client MUST clear the **Download Initiated Flag** and exit the Download Schedule Session.
- If the server status is marked as "free" in the **Server Information List** of the segment ID:
    - If the available block ranges of the server are empty:
        - If the requested block ranges in the segment consist of three or less consecutive blocks, then the client MUST:
            - Locate the first block that is marked as "idle" and change it to "downloading".
            - Change the server status to "busy" and initiate a Retrieval Protocol GetBlocks request to the server for that block.
            - Clear the **Download Initiated Flag** and exit the Download Schedule Session.
    - If the server has a set of available block ranges of the segment:

- The client MUST locate the first block in the host cache's available block ranges that is marked as "idle", change the **Block Download Status** (see 5.1 Abstract data Model) of that block to "downloading", change the server status to "busy", and initiate a Retrieval Protocol GetBlocks request to the server for that block. Then the client MUST clear the **Download Initiated Flag** and exit the Download Schedule Session.

- (No free server) If the server is marked as "complete", the client MUST abort the segment retrieval session and notify the framework of missing blocks.

- Otherwise (server is busy), the client MUST clear the **Download Initiated Flag** and exit the Download Schedule Session.

6.5.4.2 Retrieval Protocol GetBlockList Succeeds

When a Retrieval Protocol GetBlockList exchange returns valid block ranges of the requested segment, the client MUST:

- Record the block ranges into the corresponding server entry in the **Server Information List** of the segment ID, and set the server status as "free".

- If the **Download Initiated Flag** is not set; set the flag and start the Download Schedule Session.

6.5.4.3 Retrieval Protocol GetBlocks Succeeds

When a Retrieval Protocol GetBlocks exchange returns a valid block of the requested segment block ranges, the client MUST:

- Store the block in the **Content Cache** (see 5.1.2.1 Abstract Data Model) and mark the block status as "downloaded".

- If all blocks in the available block ranges of the server are all completed, mark the server status as "complete". Otherwise, mark the server status as "free" in the **Server Information List**.

- If the **Download Initiated Flag** is not set, set the flag and start the Download Schedule Session.

6.5.4.4 Retrieval Protocol Failure (GetBlockList or GetBlocks)

When a Retrieval Protocol GetBlockList request (see [MS-PCCRR] section 2.2.4.2) fails, the client MUST:

- Remove the server from the **Server Information List** of the segment ID.

- If the **Download Initiated Flag** is not set, set the flag and start a Download Schedule Session.

When a Retrieval Protocol GetBlocks request (see [MS-PCCRR] section 2.2.4.3) fails, the client MUST:

- Set the status of the requested block to idle.

- Remove the server from the **Server Information List** of the segment ID.

- If the **Download Initiated Flag** is not set, set the flag and start a Download Schedule Session.

6.5.5 Client Framework - Distributed Cache Mode, Other Local Events

6.5.5.1 Server Peer Discovered by the Discovery Protocol

When a discovered peer is passed to the client by the Discovery Protocol, the client MUST perform the following actions.

- If the **Server Information List** of the segment ID contains the maximum number of server entries, delete the least recently used server. <21>
- Add the newly discovered server into the **Server Information List** of the segment ID, and set the corresponding server status as free.
- If the requested block ranges in the segment consist of three or fewer consecutive blocks <22>, the client MUST:
  - Start a Download Schedule Session with the newly discovered server if the **Download Initiated Flag** has not been set.
- If the requested ranges consist of disjoint blocks or more than three consecutive blocks, the client MUST initiate a Retrieval Protocol GetBlockList request (see [MS-PCCRR] section 2.2.4.2) to the newly discovered server.

6.5.5.2 Discovery Protocol Failure - No Server Found

If the Discovery Protocol instance returns without a server being found, the client MUST abort the segment retrieval session.

6.5.5.3 Download Schedule Session

When a Download Schedule Session is started, then:

- If all requested blocks are marked as "downloaded", then the client MUST return all blocks in the requested block ranges and exit the segment retrieval session.
- If none of the requested blocks in the segment is marked as "idle", the client MUST clear the **Download Initiated Flag** and exit the Download Schedule Session.
- For every server entry marked as "free" in the **Server Information List** for the Segment ID:
  - If the list of available block ranges for the server in the **Server Information List** is empty, then:
    - If the requested block ranges in the segment consist of three or less consecutive blocks, then the client MUST:
      - Locate the first block that is marked as "idle" and change it to "downloading".
      - Change the server status to "busy" and initiate a Retrieval Protocol GetBlocks request (see [MS-PCCRR] section 2.2.4.3) to the server for that block.
    - Otherwise, the client MUST skip to the next free server.
  - If there is a list of available block ranges for the server in the **Server Information List** for the segment, then:

- The client MUST locate the first block in the server's available block ranges that is marked as "idle", change the **Block Download Status** of that block to "downloading", change the server status to "busy", and initiate a Retrieval Protocol GetBlocks request to the server for that block.
- Skip to the next free server.

- (No free server) If all the servers are marked as "complete", the client MUST abort the segment retrieval session and notify the framework of missing blocks.
- Otherwise (still some busy servers), the client MUST clear the **Download Initiated Flag** and exit the Download Schedule Session.

6.5.5.4 Retrieval Protocol GetBlockList Succeeds

When a Retrieval Protocol GetBlockList exchange (see [MS-PCCRR] section 2.2.4.2) returns valid block ranges of the requested segment, the client MUST:

- Record the block ranges into the corresponding server entry in the Server Information List of the segment ID, and set the server status as "free".
- If the Download Initiated Flag is not set, set the flag and start the Download Schedule Session

6.5.5.5 Retrieval Protocol GetBlocks Succeeds

When a Retrieval Protocol GetBlocks exchange (see [MS-PCCRR] section 2.2.4.3) returns a valid block of the requested segment block ranges, the client MUST:

- Store the block in the Content Cache and mark the block status as "downloaded".
- If all blocks in the available block ranges of the server are all completed, mark the server status as "complete". Otherwise, mark the server status as "free" in the Server Information List.
- If the Download Initiated Flag is not set, set the flag and start the Download Schedule Session.

6.5.5.6 Retrieval Protocol Failure (GetBlockList or GetBlocks)

The cause of Retrieval Protocol failure could be that the exchange is aborted (see [MS-PCCRR] section 3.1.1.5), or that the Request Timer for the Retrieval Protocol expires<23>. This section specifies the client action when each type of request (GetBlockList (MSG_GETBLKLIST [MS-PCCRR] section 2.2.4.2) or GetBlocks (MSG_GETBLKS [MS-PCCRR] section 2.2.4.3) fails.

When a Retrieval Protocol GetBlockList request fails, the client MUST:

- Remove the server from the Server Information List of the segment ID if the number of failures exceeds the maximum number allowed. <24>
- If the Download Initiated Flag is not set, set the flag and start a Download Schedule Session.

When a Retrieval Protocol GetBlocks request fails, the client MUST:

- Set the status of the requested block to "idle".
- Remove the server from the Server Information List of the segment ID.
- If the Download Initiated Flag is not set, set the flag and start a Download Schedule Session.

6.6 Initialization and Reinitialization Procedures

[MS-PCCRTP] - The HTTP Client is initialized as per [MS-PCCRTP] section 3.1.4.

[MS-PCCRD] - Probe and Probe-Match messages are initialized as per [MS-PCCRD] section 3.1.3.

[MS-PCHC] - The hosted cache is initialized as per [MS-PCHC] section 3.1.3.

[MS-PCCRR] - The peer role server side is initialized as per [MS-PCCRR] section 2.1.1.

6.6.1 Client Framework

6.6.1.1 Hosted Cache Mode

A hosted cache client MUST be configured with the server's address.

6.6.1.2 Distributed Cache Mode

No specific initialization required.

6.7 Status and Error Returns

See the individual member protocol Technical Documents for full details.

[MS-PCCRR] Block Download Status - For each block in the requested block ranges of the segment, a flag is stored to indicate the download status (idle, downloading, downloaded, or error).

[MS-PCCRD] - Message processing and error handling refer to [WS-Discovery] .

[MS-PCHC] - HTTP Status Code 401 Response Received - The client MUST resend the request, indicating to the transport that SPNEGO-based HTTP authentication should be performed.

7 Security

This section documents system-wide security issues that are not otherwise described in the Technical Documents (TDs) for the Member Protocols. It does not duplicate what is already in the Member Protocol TDs unless there is some unique aspect that applies to the system as a whole.

This section describes the security of data during transportation and at rest (in the client cache or Hosted Cache).

A typical sequence of events is as follows:

A content client requests data from a content server, and by requesting metadata (note that there are different mechanisms for SMB2.1 and HTTP) indicates that it is capable of understanding content caching.

1. The content server authenticates and authorizes the client in exactly the same way it would if caching were not being used. That is, authentication and authorization of a client to access data are independent of content caching.

2. The content server recognizes that the content client can utilize content caching, and checks to make sure that the stored metadata is up-to-date with the content.

3. The content server then sends the metadata on the same channel that data normally would have been sent. If an SSL connection has been established between the client and the server, then the hashes are sent back over this encrypted SSL connection.

4. The content client that is requesting content obtains the metadata and uses it to discover local availability.

5. The content client establishes a connection with the caching computer (a Hosted Cache server when Hosted Cache mode is used, or a peer caching computer when distributed cache mode is used), and requests the blocks that it requires.

6. The caching computer encrypts the blocks with an encryption key that is derived from the content metadata (using AES 128 by default) and sends it to the Content Client. The details of the encryption process for AES 128 can be found in [FIPS197].

7. The content client decrypts the data by using the same encryption key as the caching computer. The content client and the caching computer compute the same encryption key because they derive it from the same content metadata, which is sent by the Content Server.

8. After the content client decrypts the data, it validates the data. To do this, the content client computes the block hashes on the blocks received, and then compares them to the block hashes received in the content metadata from the server. If the hashes do not match, the content client discards the data.

The data in the cache is accessible. The data is stored in the clear in the distributed cache and the Hosted Cache, which is similar to other caches and data on the system.

7.1 Use of Cryptography

Cryptographic algorithms used in within the Content Caching and Retrieval System are AES-128 and SHA256 ([FIPS197] and [FIPS180-2]).

8 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: Windows Server 2008 R2 is the first server to support the server feature BranchCache retrieval – Hosted Cache Mode. The Hyper-V Core and Home Server SKU are not branch cache capable. All other server Windows Server 2008 R2 SKUs can act as branch cache clients and/or content servers.

<2> Section 1: Windows Server 2008 R2 and Windows 7 support the client feature BranchCache retrieval – Hosted Cache Mode and Distributed Mode. BranchCache retrieval is available with Windows 7 Enterprise and Ultimate.

<3> Section 5.1: Windows 7 clients will offer MS KRB 5, KRB5 SNMPV2, and NTLMSSP if they are configured with the FQDN of the hosted cache server. If a NetBIOS address or IP address is given for the hosted cache server, then only NTLMSSP is offered.

<4> Section 5.1: Windows uses a value of 6 for retries and will retry up to 6 times with an interval of 10 seconds between tries.

<5> Section 5.1.2.2: Segment IDs for which the Discovery Protocol discovered no peers are also stored here, marked as "negative discoveries".

<6> Section 5.1.2.2: The Windows implementation uses a 15 -second lifetime for an entry in the discovery cache with no associated address.

<7> Section 5.1.2.2: The Windows implementation uses 1 minute as the lifetime for each address that is added to a discovery cache entry.

<8> Section 5.1.2.2: The Windows implementation uses 256 as the default max size. The value is configurable from 16 to 16,384, inclusive.

<9> Section 5.1.2.2: The Windows implementation uses 16 as the default max size. The value is configurable from 1 to 10,000, inclusive.

<10> Section 5.1.2.2: The Windows implementation uses a FMax = 100 discoveries/seconds. FMax is configurable from 1 and 10 to 10,000, inclusive.

<11> Section 5.1.2.2: The Windows implementation uses a DMax = 512 as default max number of active discoveries. DMax is configurable from 1 to 1024, inclusive.

<12> Section 5.2.2: In Windows Vista and Windows Server 2008, support for the client-side elements of Content Caching and Retrieval is available only via the optional installation of the Background Intelligent Transfer Service (BITS) via the Windows Management Framework. Support for the server-side elements of this protocol is not available for Windows Server 2008. When the Windows Management Framework is installed, the BITS service use of [MS-BPDP] is replaced by [MS-PCCRD] for discovery and [MS-BPCR] is replaced by [MS-PCCRR] for content retrieval.

<13> Section 5.2.3: In Windows Server 2008 R2, if a client requests a hash for a file or if no hash exists, this will trigger a user-mode service to create or re-create the hash for the specified file.

<14> Section 6.1.1.1: In Windows 7 and Windows Server 2008 R2, there are two ways for an application to use branch cache retrieval: directly by calling the branch cache platform APIs, or indirectly by calling WinINET/WinHTTP or SMBv2 APIs, which are instrumented to use branch cache, but transparent to the applications.

<15> Section 6.1.3.2: For a given client assembling a segment, an offer of blocks is made to the Hosted cache when 20%, 40%, 60%, 80%, and 100% of the segment has been assembled and validated. The Hosted Cache does not recognize files, only segments and blocks.

<16> Section 6.1.3.3: Windows 7 clients will offer MS KRB 5, KRB5 SNMPV2, and NTLMSSP if they are configured with the FQDN of the hosted cache server. If a NetBIOS address or IP address is given for the hosted cache server, then only NTLMSSP is offered.

<17> Section 6.2: In Windows 7 and Windows Server 2008 R2, there are two ways for an application to use branch cache retrieval: directly by calling the branch cache platform APIs, or indirectly by calling WinINET/WinHTTP or SMBv2 APIs, which are instrumented to use branch cache, but transparent to the applications.

<18> Section 6.4.2.2: In the Windows implementation, the timeout value is set by the higher-layer applications. The recommended value is 5 seconds for each segment retrieval session. The range of permitted timeout values is from 0 to 4,294,967,294 milliseconds.

<19> Section 6.4.2.2: Windows uses a 15-second lifetime for an entry in the discovery cache with no associated address.

<20> Section 6.4.2.2: Windows uses 1 minute as the lifetime for each address that is added to a discovery cache entry.

<21> Section 6.5.5.1: The Windows implementation uses 16 as the default maximum number of peers used per download. The number is configurable from 1 to 16384, inclusive.

<22> Section 6.5.5.1: The Windows implementation carries out a simple download each time a download involves less than four consecutive blocks in a single block range, which implies that the blocks are also adjacent (consecutive) in the segment.

<23> Section 6.5.5.6: Windows uses a 2-second timeout for each request message. The timeout is configurable between 1 millisecond and 1 minute.

<24> Section 6.5.5.6: The client-role peer Windows implementation allows a serving-role peer to time out up to a configurable three times, before excluding the peer from the current download. This value is configurable from 1 to 100, inclusive.

9 Change Tracking

This section identifies changes that were made to the [MS-CCRSO] 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.

10 Index