[MS-DLTCS]:
Distributed Link Tracking Central Store Protocol Specification

---

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
03/14/2007	1.0	Major	Updated and revised the technical content.
04/10/2007	1.1	Minor	Updated the technical content.
05/18/2007	1.2	Minor	Addressed EU feedback
06/08/2007	1.3	Minor	Updated the technical content.
07/10/2007	1.3.1	Editorial	Revised and edited the technical content.
08/17/2007	1.3.2	Editorial	Revised and edited the technical content.
09/21/2007	1.3.3	Editorial	Revised and edited the technical content.
10/26/2007	2.0	Major	Converted document to unified format.
01/25/2008	2.0.1	Editorial	Revised and edited the technical content.
03/14/2008	2.0.2	Editorial	Revised and edited the technical content.
06/20/2008	2.0.3	Editorial	Revised and edited the technical content.
07/25/2008	2.0.4	Editorial	Revised and edited the technical content.
08/29/2008	3.0	Major	Added a section.
10/24/2008	3.0.1	Editorial	Revised and edited the technical content.
12/05/2008	4.0	Major	Updated and revised the technical content.
01/16/2009	4.0.1	Editorial	Revised and edited the technical content.
02/27/2009	4.0.2	Editorial	Revised and edited the technical content.
04/10/2009	4.0.3	Editorial	Revised and edited the technical content.
05/22/2009	4.0.4	Editorial	Revised and edited the technical content.
07/02/2009	4.0.5	Editorial	Revised and edited the technical content.
08/14/2009	4.0.6	Editorial	Revised and edited the technical content.
09/25/2009	4.1	Minor	Updated the technical content.
11/06/2009	4.1.1	Editorial	Revised and edited the technical content.
12/18/2009	4.1.2	Editorial	Revised and edited the technical content.
01/29/2010	4.2	Minor	Updated the technical content.
03/12/2010	4.2.1	Editorial	Revised and edited the technical content.

Date	Revision History	Revision Class	Comments
04/23/2010	4.2.2	Editorial	Revised and edited the technical content.
06/04/2010	4.2.3	Editorial	Revised and edited the technical content.
07/16/2010	4.2.3	No change	No changes to the meaning, language, or formatting of the technical content.
08/27/2010	4.2.3	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	4.2.3	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	4.2.3	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	4.2.3	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	4.2.3	No change	No changes to the meaning, language, or formatting of the technical content.

Contents

1 Introduction

This document specifies the Distributed Link Tracking: Central Store Protocol.

Distributed Link Tracking (DLT) refers to a set of protocols used to determine the new location of a file that has moved, whether the file has moved within a computer or between computers in a network that shares files with the Server Message Block (SMB) Protocol, as specified in [MS-SMB].

The Distributed Link Tracking: Central Store Protocol is used to store the tables of the Distributed Link Tracking: Central Manager Protocol, as specified in [MS-DLTM], and to transmit table updates between instances of the Distributed Link Tracking: Central Store Protocol servers.

This protocol is based on **Active Directory**, as specified in [MS-ADTS]. Specifically, this protocol treats Active Directory as a transport itself. For example, if a server of the Distributed Link Tracking: Central Manager Protocol writes an object to Active Directory according to the Distributed Link Tracking: Central Store Protocol, Active Directory replicates that object to another computer, where it can be read by another instance of a Distributed Link Tracking (DLT) Central Manager server. This Distributed Link Tracking: Central Store Protocol describes how Active Directory objects are defined, updated, and interpreted. The replication mechanism for Active Directory itself is as specified in [MS-ADTS].

In addition to the Distributed Link Tracking: Central Store Protocol, there are two other protocols that make up Distributed Link Tracking:

- The Distributed Link Tracking: Central Manager Protocol, as specified in [MS-DLTM], is a remote procedure call (RPC)-based protocol that is used to send information from protocol clients to servers about files that have been moved between computers or between volumes within a computer, information such as a unique ID for a file and the ID of the computer on which a file is currently located. This protocol is also used to query for the identity of the computer that currently holds a file.
- The Distributed Link Tracking: Workstation Protocol, as specified in [MS-DLTW], is an RPC-based protocol that is used to determine a file's current **Universal Naming Convention (UNC)** location. Clients of this protocol may use the Distributed Link Tracking: Central Manager Protocol to determine which computer currently holds a particular file. This allows the client to determine the correct instance of the DLT Workstation server to contact.

1.1 Glossary

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

Active Directory
distinguished name (DN)
domain
domain controller (DC)
FileLocation
FileTable
HexConvertedUnicodeString
MachineID
ObjectID
RefreshTime
relative distinguished name (RDN)
relative identifier (RID)
StoreMaster

Unicode
Universal Naming Convention (UNC)
VolumeID
VolumeOwner
VolumeSecret
VolumeSequenceNumber
VolumeTable

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

FileID

The following terms are specific to this document:

CurrentRefreshTime: The current time, in units of days, measuring the time since the value was initialized.

IntegerConvertedUnicodeString: A Unicode string created from a binary value. The string is a representation of the integer interpretation of the binary value. For example, a value of 0x10 would be represented as the string "16".

SystemObject: An object with **Active Directory**. This object is always at the **distinguished name (DN)** "CN=System,DC=DomainName", where DomainName is the name of the **domain**.

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

1.2 References

This section contains the following information:

Normative references specify stable, published documents that must be read to understand or implement the technology in this document, or whose technology must be present for the technology in this protocol to work. This includes public specifications that define the relevant protocols, and documents that describe the Windows behavior (if other than the protocol specification).

Informative references are published documents that provide additional, optional information relevant to the protocol. For example, an informative reference might provide background or historical information. Informative references are not required to implement the technology in this protocol.

1.2.1 Normative References

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

[MS-ADA1] Microsoft Corporation, "Active Directory Schema Attributes A-L", June 2007.

[MS-ADA2] Microsoft Corporation, "Active Directory Schema Attributes M", July 2006.

[MS-ADA3] Microsoft Corporation, "Active Directory Schema Attributes N-Z", July 2006.

[MS-ADSC] Microsoft Corporation, "Active Directory Schema Classes", July 2006.

[MS-ADTS] Microsoft Corporation, "Active Directory Technical Specification", July 2006.

[MS-DLTM] Microsoft Corporation, "Distributed Link Tracking: Central Manager Protocol Specification", July 2006.

[MS-SMB] Microsoft Corporation, "Server Message Block (SMB) Protocol Specification", July 2006.

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

1.2.2 Informative References

[MS-DLTW] Microsoft Corporation, "Distributed Link Tracking: Workstation Protocol Specification", July 2006.

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

1.3 Overview

The Distributed Link Tracking: Central Store Protocol is designed to be used by the Distributed Link Tracking: Central Manager Protocol, as specified in [MS-DLTM]. The Distributed Link Tracking: Central Store Protocol describes how to store the **ServerVolumeTable** and **FileTable** tables, as specified in [MS-DLTM] section 3.1.1, in Active Directory objects, where the DLT Central Manager servers are running on ALL **domain controllers (DCs)** of a **domain**. Because the Active Directory objects are replicated between servers, this allows updates by one DLT Central Manager server (as specified in [MS-DLTM]) to table entries to be communicated to other protocol server instances that are part of the same domain.<1>

As described in the introduction, DLT is composed of three protocols. The following is an example of the three protocols working together:

- A file is created on computer M1. M1 assigns identifiers to the file.
- Computer M0 takes note of the file, locally storing its identifiers.
- The file moves from computer M1 to M2, and from there to M3.
- Computer M0 finds the file in its new location in one of two ways:

  1. Using only the Distributed Link Tracking: Workstation Protocol:

     - M0 contacts M1, using the identifiers stored previously, and learns that the file was moved to M2.
     - M0 contacts M2, and learns that the file was moved to M3.
     - M0 contacts M3, and learns the file's new name and location.

  2. Using all three protocols:

     - M0 contacts a DLT Central Manager server to query the current location of the file.
     - The DLT Central Manager server queries its tables, which are stored by the Distributed Link Tracking: Central Store Protocol, and determines that the file is currently on computer M3.

- M0 contacts the Distributed Link Tracking: Workstation Protocol on M3, and learns the file's new name and location.

1.4 Relationship to Other Protocols

The Distributed Link Tracking: Central Store Protocol is dependent on Active Directory, as specified in [MS-ADTS], which is the store for its tables.

The Distributed Link Tracking: Central Store Protocol imports conceptual tables used by the Distributed Link Tracking: Central Manager Protocol (as specified in [MS-DLTM] section 3.1.1) server implementation. It is used by a DLT Central Manager server to replicate information to other such servers.

1.5 Prerequisites/Preconditions

Agents using this protocol must have access to Active Directory, as specified in [MS-ADTS], and the right to access and modify objects within **SystemObject**.

1.6 Applicability Statement

The Distributed Link Tracking: Central Store Protocol is applicable in configurations in which the Distributed Link Tracking: Central Manager Protocol, as specified in [MS-DLTM], is being used.

1.7 Versioning and Capability Negotiation

There is no versioning or capability negotiation in this protocol.

1.8 Vendor-Extensible Fields

This protocol does not define any vendor-extensible fields.

1.9 Standards Assignments

Parameter	Value	Reference
VolumeTable object	"CN=VolumeTable,CN=FileLinks,CN=System"	As specified in [MS-ADTS] (see also section 2.2.2)
FileTable object	"CN=ObjectMoveTable,CN=FileLinks,CN=System"	As specified in [MS-ADTS] (see also section 2.2.4)

2 Messages

The following sections specify how Distributed Link Tracking: Central Store Protocol messages are transported, and their syntax.

2.1 Transport

Data MUST be transferred between agents of this protocol by replication of Active Directory, as specified in [MS-ADTS]. That is, this protocol treats Active Directory itself as a protocol. Writing objects into Active Directory causes the Active Directory replication mechanism to transfer those objects over the Active Directory underlying protocol where they can be read by another agent.

2.2 Message Syntax

This section defines the individual objects in Active Directory that make up this protocol. The full schema of these objects is as specified in [MS-ADTS].

2.2.1 FileLinks Object

The FileLinks object is the container object for all link information and is found at the following **relative distinguished name (RDN)** within SystemObject.

```
CN=FileLinks
```

The schema definition for the FileLinks object is the **fileLinkTracking** object, as specified in [MS-ADSC] section 2.48.

Exactly one FileLinks object MUST exist.

2.2.2 VolumeTable Object

The VolumeTable object represents the ServerVolumeTable, as specified in [MS-DLTM] section 3.1.1, and MUST be stored in Active Directory as an object at the following RDN within SystemObject.

```
CN=VolumeTable,CN=FileLinks
```

The schema definition for the VolumeTable object MUST be the **linkTrackVolumeTable** object, as specified in [MS-ADSC] section 2.80.

2.2.3 VolumeTableEntry Object

The VolumeTableEntry object represents an entry in the ServerVolumeTable, as specified in [MS-DLTM] section 3.1.1, and MUST be stored in the VolumeTable object (section 2.2.2).

The schema definition for the VolumeTableEntry object MUST be the **linkTrackVolEntry** object specified in [MS-ADSC] section 2.79, and has an RDN of the following form.

```
VolumeTableRDN = "CN=" + VolumeIDString
VolumeIDString = A VolumeID in the form of a HexConvertedUnicodeString
```

A sample VolumeTableEntry object **distinguished name (DN)** is shown in the following example where "DomainName" is the name of the domain.

```
CN=E3D954B2D0A711D08CB600C04FD90F85,
+ CN=VolumeTable,CN=FileLinks,CN=System,
+ DC=DomainName
```

There are two special VolumeTableEntry objects with specific RDNs in the **VolumeTable** object that are not used as in the preceding example:

- **CurrentRefreshTimeEntry**: This entry MUST have an RDN of "0000000000000000", as if its RDN indicated a **VolumeID** consisting of a string of all "0" characters. Only the **seqNotification** attribute of this entry is used. It MUST be a 32-bit unsigned integer value in the form of an **IntegerConvertedUnicodeString**. The **seqNotification** attribute MUST be set to the value of the **CurrentRefreshTime**. All other attributes MUST NOT be set and MUST be ignored.

  Updates to this value are specified in Timer Events (section 3.1.6).

- **FileTableCounterEntry**: This entry MUST have an RDN of "QT_Counter". Only the **linkTrackSecret** attribute of this entry SHOULD be used. All other attributes MUST NOT be set, and MUST be ignored. This is an 8-byte binary value, representing a partial count of the number of entries in the FileTable. Updates to this value are specified in Timers (section 3.1.2).

2.2.4 FileTable Object

The FileTable object represents the FileTable, as specified in [MS-DLTM] section 3.1.1, and MUST be stored in Active Directory as an object at the following RDN within SystemObject.

```
CN=ObjectMoveTable,CN=FileLinks
```

The schema definition for the FileTable object MUST be the **linkTrackObjectMoveTable** object specified in [MS-ADSC] section 2.77.

2.2.5 FileTableEntry Object

The FileTableEntry object represents an entry in the FileTable, as specified in [MS-DLTM] section 3.1.1, and MUST be stored in the FileTable object (section 2.2.4).

The schema definition for the FileTableEntry object MUST be the **linkTrackOMTEntry** object specified in [MS-ADSC] section 2.78, and MUST have an RDN of the following form.

```
FileTableRDN = "CN=" + VolumeIDString + ObjectIDString
VolumeIDString = VolumeID in the form of a HexConvertedUnicodeString
ObjectIDString = ObjectID in the form of a HexConvertedUnicodeString
```

A sample FileTableEntry object DN is shown in the following example, where "DomainName" is the name of the domain.

```
CN=E3D954B2D0A711D08CB600C04FD90F85
+ 6454147C5A29427F-B7B03982C5202C2A,
+ CN=ObjectMoveTable,CN=FileLinks,CN=System,
+ DC=DomainName
```

2.3 Directory Service Schema Elements

The Distributed Link Tracking: Central Store Protocol accesses the Directory Service schema classes and attributes listed in the following table. For the syntactic specifications of the following <Class> or <Class> <Attribute> pairs, refer to [MS-ADSC], [MS-ADA1], [MS-ADA2], and [MS-ADA3].

Class	Attribute
fileLinkTracking (section 2.48)	
linkTrackVolumeTable (section 2.80)	
linkTrackVolEntry (section 2.79)	seqNotification (section 2.238), linkTrackSecret (section 2.362), volTableIdxGUID (section 2.365), timeRefresh (section 2.307)
linkTrackObjectMoveTable (section 2.77)	
linkTrackOMTEntry (section 2.78)	currentLocation (section 2.137), birthLocation (section 2.84), oMTIndxGuid (section 2.50), timeRefresh (section 2.307)

3 Protocol Details

The Distributed Link Tracking: Central Store Protocol is used to store file link information in a VolumeTable table and a FileTable table within Active Directory. This protocol is used as a storage implementation by the DLT Central Manager server, as specified in [MS-DLTM], and is used to coordinate between instances of such servers.

3.1 Central Store Details

3.1.1 Abstract Data Model

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

The Distributed Link Tracking: Central Store Protocol maintains information about linked files. Information is in the form of two conceptual tables imported from (that is, shared with) the Distributed Link Tracking: Central Manager Protocol, as specified in [MS-DLTM]. Those tables are called ServerVolumeTable and FileTable, as specified in [MS-DLTM] and detailed in section 3.1.1. Hereafter in this document, the ServerVolumeTable is referred to simply as the VolumeTable.

The fields of VolumeTable MUST map to the **linkTrackVolEntry** object attributes, as specified in section 2.2.2, as follows.

Field	Attribute name
VolumeSequenceNumber	seqNotification
VolumeSecret	linkTrackSecret
VolumeOwner	volTableIdxGUID
RefreshTime	timeRefresh

The VolumeID of an entry in the VolumeTable MUST correspond to the RDN of the **linkTrackVolEntry** object, as described in section 2.2.3.

The fields of FileTable MUST map to the **linkTrackOMTEntry** object attributes, as specified in section 2.2.4, as follows:

Field	Attribute name
FileLocation	currentLocation
FileID	birthLocation
Flags	oMTIndxGuid
RefreshTime	timeRefresh

The **PreviousFileLocation** of an entry in the FileTable MUST correspond to the RDN of the **linkTrackOMTEntry**, as described in section 2.2.5.

The **Flags** field provides information on the state of an object, as follows.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
0	0	U	D	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

U (Flags_Uncounted): This flag indicates that an entry in FileTable is not represented in the **FileTableCounterEntry**, as specified in section 2.2.3.

D (Flags_Deleted): This flag indicates that an entry in FileTable represents a file that has been deleted.

Some operations on the tables are performed by a single entity. This entity is called **StoreMaster**. The StoreMaster entity is the DC that is the **relative identifier (RID)** Master DC, as specified in [MS-ADTS].

3.1.2 Timers

TableMaintenanceTimer: Causes the StoreMaster agent to perform maintenance on VolumeTable, as specified in 2.2.2, and FileTable, as specified in 2.2.4. For the maintenance performed, see section 3.1.6. This is a recurring timer and MUST fire once per day.

3.1.3 Initialization

Implementations of this protocol MUST have access to the VolumeTable object and FileTable object. Access to Active Directory objects is specified in [MS-ADTS].

If the **CurrentRefreshTimeEntry** object specified in section 2.2.3 does not exist, it MUST be created with a CurrentRefreshTime value initially set to 0.

Note Because this protocol is used by servers that conform to the standards specified in [MS-DLTM], which run only on DCs, this protocol MUST only be accessed on DCs as well.

3.1.4 Higher-Layer Triggered Events

Aside from simply querying the entries in the VolumeTable object or FileTable object, agents using this protocol might need to perform more complex operations on the table. The following sections explain how to perform some of these operations.

3.1.4.1 An Agent Wants to Add an Entry to FileTable

In this scenario, the entry MUST be added to FileTable, according to the schema definition. In addition, the **Flags** field MUST be set to **Flags_Uncounted**.

3.1.4.2 An Agent Wants to Delete an Entry from FileTable

The entry in FileTable MUST NOT actually be deleted. Rather, the **Flags_Deleted** flag MUST be added to the **Flags** field for the entry in FileTable.

3.1.4.3 An Agent Wants to Know the Size of FileTable

Servers of the Distributed Link Tracking: Central Manager Protocol, as specified in [MS-DLTM] section 3.1.4.2, require information about the size of the FileTable table to determine whether the FileTable size limit has been reached during the processing of a MOVE_NOTIFICATION request.

This value MUST be calculated by incrementing the value of **FileTableCounterEntry**, as specified in section 2.2.3, once for each entry in the FileTable table that has the **Flags_Uncounted** flag set in the **Flags** field but that does not have the **Flags_Deleted** flag set.

Note This calculation does not modify the value of the **FileTableCounterEntry**.

3.1.5 Message Processing Events and Sequencing Rules

There are no messages to process in this protocol.

3.1.6 Timer Events

When the TableMaintenanceTimer timer expires, the StoreMaster agent MUST perform the following operations:

- Deleted files are cleaned up in the FileTable table:

  When a file is deleted, the **Flags_Deleted** flag is set in the **Flags** field of the corresponding FileTable entry, as specified in section 3.1.4. Any FileTable entry with this flag set MUST be deleted.

- Multiple entries representing a single chain are coalesced in the FileTable:

  - If multiple entries in the FileTable represent a single chain, they MUST be coalesced into a single FileTable entry. A chain of FileTable entries is a sequence of two or more consecutive entries that track a file through multiple moves, beginning with the file's original location in the first entry's **PreviousFileLocation** column, and ending in the file's most recent location in the last entry's **NextFileLocation** column.

  - The process of coalescing FileTable entries involves condensing a chain of FileTable entries into a single entry. The condensed single entry contains the **PreviousFileLocation** field that was originally specified in the first FileTable entry and the **NextFileLocation** field that was originally specified in the last FileTable entry.

  - Assume that FileTable contains two entries, and that the **NextFileLocation** field of the first entry matches the **PreviousFileLocation** field of the second entry. The second entry MUST be deleted from FileTable, and the first entry MUST be updated so that its **NextFileLocation** field has the value of the **NextFileLocation** field of the deleted entry.

3.1.7 Other Local Events

There are no additional local events.

4 Protocol Examples

Following is an example of the FileTable Coalescence, as specified in section 3.1.6. In this example, the file's original FileLocation entry (and therefore the file's FileID entry) is "A". Subsequently, the FileLocation entry changes to "B" and then to "C". FileTable then has the following entries.

PreviousFileLocation	NextFileLocation	FileID
A	B	(Unspecified)
B	C	A

Because these two entries represent a single file's move, they are consolidated into a single entry.

PreviousFileLocation	NextFileLocation	FileID
A	C	(Unspecified)

Note that where the **PreviousFileLocation** entry is the same as FileID, the **FileID** attribute is not stored in the entry, causing **FileID** to be inferred from the **PreviousFileLocation** value.

As a similar example, a file starts as A, moves to B, and then to C, and then to D. But in this example, the FileTable entry for the move from B to C has not been recorded yet. This causes the following entries to be in the FileTable, which cannot be coalesced.

PreviousFileLocation	NextFileLocation	FileID
A	B	(Unspecified)
C	D	A

5 Security

The following sections specify security considerations for implementers of the Distributed Link Tracking: Central Store Protocol.

5.1 Security Considerations for Implementers

There are no additional security considerations for implementers.

5.2 Index of Security Parameters

There are no security parameters associated with this protocol.

6 Appendix A: Product Behavior

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

- Microsoft Windows® 2000 Server operating system
- Windows Server® 2003 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 1.3: This protocol is implemented only on Windows 2000 Server and Windows Server 2003.

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8 Index