[MS-OXABREF]:
Address Book Name Service Provider Interface (NSPI) Referral Protocol Specification

---

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability.
04/25/2008	0.2		Revised and updated property names and other technical content.
06/27/2008	1.0		Initial Release.
08/06/2008	1.01		Revised and edited technical content.
09/03/2008	1.02		Updated references.
12/03/2008	1.03		Updated IP notice.
04/10/2009	2.0		Updated technical content and applicable product releases.
07/15/2009	3.0	Major	Revised and edited for technical content.
11/04/2009	4.0.0	Major	Updated and revised the technical content.
02/10/2010	4.1.0	Minor	Updated the technical content.
05/05/2010	4.2.0	Minor	Updated the technical content.
08/04/2010	5.0	Major	Significantly changed the technical content.
11/03/2010	6.0	Major	Significantly changed the technical content.

Contents

1 Introduction

This document describes the Address Book Name Service Provider Interface (NSPI) Referral Service (NSPIReferral). NSPIReferral is a **remote procedure call (RPC)** service that supplies a caller with the name of an **NSPI** server. Additionally, NSPIReferral can return the **Domain Name System (DNS) fully qualified domain name (FQDN)** of a **mailbox** server, given the **distinguished name (DN)** of that server.

1.1 Glossary

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

Address Book object
binding handle
distinguished name (DN)
Domain Name System (DNS)
dynamic endpoint
endpoint (2)
flags
fully qualified domain name (FQDN)
Interface Definition Language (IDL)
mailbox
Name Service Provider Interface (NSPI)
Network Data Representation (NDR)
NTLM
opnum
remote procedure call (RPC)
RPC protocol sequence
property (1)
public folder
security provider
universal unique identifier (UUID)
well-known endpoint

The following terms are specific to this document:

Kerberos: An authentication system that enables two parties to exchange private information across an otherwise open network by assigning a unique key (called a ticket) to each user that logs on to the network and then embedding these tickets into messages sent by the users. For more information, see [MS-KILE].

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

1.2 References

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.

[C706] The Open Group, "DCE 1.1: Remote Procedure Call", C706, August 1997, http://www.opengroup.org/public/pubs/catalog/c706.htm

[MS-DTYP] Microsoft Corporation, "Windows Data Types", March 2007, http://msdn.microsoft.com/en-us/library/cc230273.aspx

[MS-KILE] Microsoft Corporation, "Kerberos Protocol Extensions", July 2006, http://msdn.microsoft.com/en-us/library/cc206927.aspx

[MS-OXOABK] Microsoft Corporation, "Address Book Object Protocol Specification", April 2008.

[MS-RPCE] Microsoft Corporation, "Remote Procedure Call Protocol Extensions", July 2006, http://msdn.microsoft.com/en-us/library/cc243560.aspx

[RFC1035] Mockapetris, P., "DOMAIN NAMES – IMPLEMENTATION AND SPECIFICATION", RFC 1035, November 1987, http://www.ietf.org/rfc/rfc1035.txt

[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-OXGLOS] Microsoft Corporation, "Exchange Server Protocols Master Glossary", April 2008.

1.3 Overview

NSPIReferral serves to return the network name of a server to a client. It has two methods, RfrGetNewDSA, which returns the name of an NSPI server, and RfrGetFQDNFromServerDN, which returns the DNSFQDN of a mailbox server. Figure 1 shows the RfrGetNewDSA method call. Figure 2 shows the RfrGetFQDNFromServerDN method call.

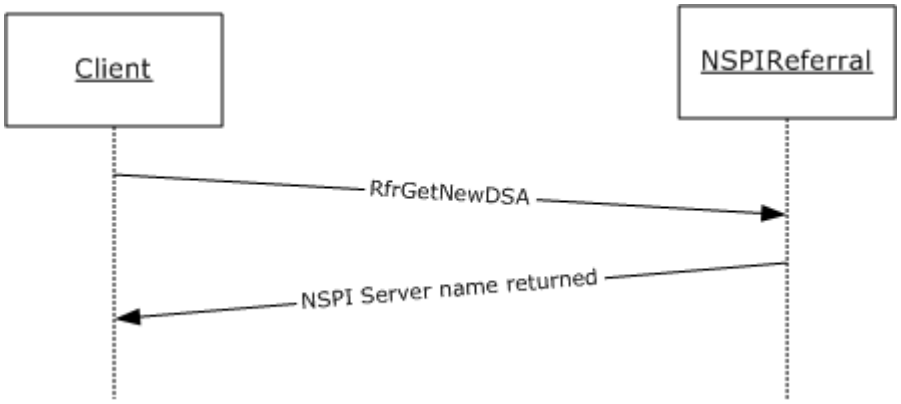

Figure 1: Relationship between client and NSPIReferral

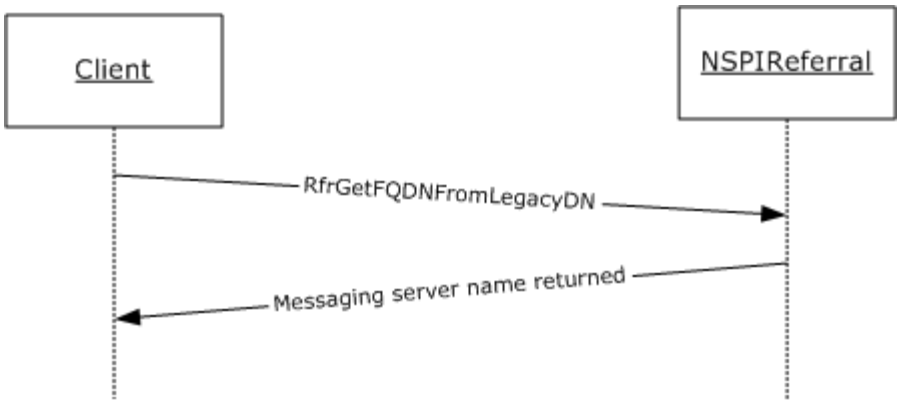

Figure 2: Client retrieving mailbox server name from NSPIReferral

1.4 Relationship to Other Protocols

NSPIReferral is built on the Microsoft remote procedure call (RPC) interface (as specified in [C706] and [MS-RPCE]). It supports only **RPC protocol sequences** ncacn_ip_tcp and ncacn_http, as specified in [MS-RPCE].

1.5 Prerequisites/Preconditions

None.

1.6 Applicability Statement

Clients request an NSPI server by using the **RfrGetNewDSA** method of NSPIReferral before the client engages in any NSPI requests. This gives the NSPIReferral server the ability to control which NSPI server an NSPI client will connect to, for purposes including but not limited to balancing the client load across multiple NSPI servers, choosing the best version of NSPI server for that particular client, or satisfying network requirements that are not discernible by the client.

The **RfrGetFQDNFromServerDN** method of NSPIReferral is appropriate when a client knows the distinguished name (DN) of a mailbox server but not the FQDN of the mailbox server (as specified in [RFC1035]) with which it can make a network connection to that server. In practice, this is necessary in several cases:

- When creating client mail settings, a client uses an NSPI server to read an **Address Book object** representing its mailbox, which includes the DN of the messaging server that hosts the mailbox.
- When connecting to the wrong mailbox or **public folder** server, an error will be returned containing the DN of the correct server.
- When connecting to another user's mailbox, having only the PidTagAddressBookHomeMessageDatabase property for that mailbox. For more details about the PidTagAddressBookHomeMessageDatabase **property**, see [MS-OXOABK] section 2.2.4.67.

1.7 Versioning and Capability Negotiation

This specification covers versioning issues in the following areas:

- **Supported Transports:** This protocol uses multiple RPC protocol sequences as specified in section 2.1.

- **Protocol Versions:** NSPIReferral has only one interface version. The use of these methods is specified in section 3.1.
- **Security and Authentication Methods:** NSPIReferral supports the following authentication methods: **NTLM** and **Kerberos**.

There is no capability negotiation.

1.8 Vendor-Extensible Fields

RfrGetNewDSA and **RfrGetFQDNFromServerDN** return HRESULT values as defined in [MS-DTYP]. Vendors MAY define their own HRESULT values, provided that they set the C bit (0x20000000) for each vendor-defined value to indicate that the value is a customer code.

RfrGetNewDSA can also return other error values. Any nonzero return code indicates an error.

1.9 Standards Assignments

This protocol uses a **well-known endpoint**, as specified in section 2.1. This protocol uses RPC **dynamic endpoints** as defined in [C706] part 4.

Parameter	Value	Reference
RFRI RPC interface **UUID**	1544f5e0-613c-11d1-93df-00c04fd7bd09	Appendix A

2 Messages

2.1 Transport

The NSPIReferral protocol works over the following RPC protocol sequences:

- ncacn_ip_tcp — RPC over TCP
- ncacn_http — RPC over HTTP

The NSPIReferral protocol uses a well-known endpoint, 6002, for the RPC protocol sequence ncacn_http. For ncacn_ip_tcp, the NSPIReferral protocol uses a dynamic endpoint, as specified in [C706] Part 4. A dynamic endpoint is configurable by the owner of the server to enable the owner to open a minimum set of ports in a firewall to give clients access to the server.

The following table summarizes the **endpoints** used by the NSPIReferral protocol.

RPC Protocol Sequence	Endpoint
ncacn_http	6002 (well-known endpoint)
ncacn_ip_tcp	dynamic endpoint, as specified in [C706] Part 4

The NSPIReferral protocol supports NTLM (RPC_C_AUTHN_WINNT), and Negotiate (RPC_C_AUTHN_GSS_NEGOTIATE) **security providers**. A Negotiate security provider determines whether to use NTLM or Kerberos authentication. The default is Kerberos. A Negotiate security provider selects NTLM authentication only in the following cases:

- One of the systems that is involved in the authentication cannot use Kerberos authentication.
- The client does not provide sufficient information to use Kerberos authentication.

Callers MUST be authenticated but no further authorization checks are performed.

2.2 Common Data Types

This protocol MUST indicate to the RPC runtime that it is to support the **Network Data Representation (NDR)** transfer syntax only, as specified in [C706] Part 4.

In addition to RPC base types and definitions specified in [C706] and [MS-RPCE], additional data types are defined in this section.

2.2.1 handle_t

The **handle_t** data type is used to represent an explicit RPC **binding handle**, as specified in [C706] and [MS-RPCE] section 2. It is a primitive type of the **Interface Definition Language (IDL)** and does not require an explicit declaration.

3 Protocol Details

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

3.1 NSPIReferral Server Details

This is a simple single-request, single-response protocol.

3.1.1 Abstract Data Model

A data structure that tracks the available NSPI servers and their current state is beneficial to any implementation. This internal state means the client is more likely to get a good NSPI server name and connect successfully on the first try. The NSPIReferral server is not required to connect to the NSPI server in order to service clients; therefore, it is important for an NSPIReferral implementation to use some method to maintain up-to-date information about available NSPI servers. This ensures that clients who call RfrGetNewDSA are not given the name of an NSPI server that is not functioning.

3.1.2 Timers

None required.

3.1.3 Initialization

Initialization occurs at server startup. NSPIReferral registers the NSPIReferral protocol interface with the RPC system using the RFRI RPC interface UUID, from section 1.9.

3.1.4 Message Processing Events and Sequencing Rules

This protocol MUST indicate to the RPC runtime that it is to perform a strict NDR data consistency check at target level 5.0, as specified in [MS-RPCE] section 3.

The following table lists the methods that are included in this interface.

Method	Description
RfrGetNewDSA	Returns the name of an NSPI server. **opnum**: 0
RfrGetFQDNFromServerDN	Returns the domain name system (DNS) FQDN of the server corresponding to the passed DN. For more details about domain name systems, see [RFC1035]. opnum: 1

All methods MUST NOT throw exceptions.

3.1.4.1 RfrGetNewDSA (opnum 0)

The **RfrGetNewDSA** method returns the name of an NSPI server.

```
//opnum 0
```

```
long RfrGetNewDSA(
 [in]          handle_t       hRpc,
 [in]              unsigned long   ulFlags,
 [in, string]    unsigned char *   pUserDN,
 [in,out,unique, string]    unsigned char * * ppszUnused,
 [in,out,unique, string]    unsigned char * * ppszServer);
```

hRpc: An RPC binding handle parameter, as specified in [C706] section 2. MUST NOT be NULL.

ulFlags: An unsigned long value, containing a set of bit **flags**. Unused; SHOULD be set to zero. Other values MUST be ignored by the server.

pUserDN: Optional, a DN indicating the mailbox owned by the client user. The client SHOULD pass this to the server. If supplied, the server SHOULD use that DN to affect which NSPI server is returned to the caller.

ppszUnused: A string. Unused; SHOULD be set to NULL. Other values MUST be ignored by the server.

ppszServer: A string. If the server does not return an error, ppszServer contains the FQDN of an NSPI server. On failure, the value is undefined.

Return Values: The server returns 0 for a successful execution. An error results in an HRESULT or other nonzero error code.

Exceptions Thrown: No exceptions are thrown beyond those thrown by the underlying RPC protocol as specified by [MS-RPCE].

Upon receiving this message, the server MUST process the data from the client to the following constraints. If pUserDN is present and contains the DN of an Address Book object, the server MUST prioritize an NSPI server that contains a writeable copy of that Address Book object over NSPI servers that do not, and return a client access array or a server from the user's mailbox site.<1> The server SHOULD take other constraints into account, such as the network location of the NSPI server in comparison to the NSPIReferral server or the client. The server MUST prioritize available, responsive NSPI servers over unresponsive ones. The server SHOULD consider load balancing of clients when more than one NSPI server has equal priority. After considering these constraints, NSPIReferral SHOULD return one NSPI server name in the ppszServer parameter and a return value of zero. If any errors occur and NSPIReferral is not able to return the name of an NSPI server, a nonzero value MUST be returned.

Because the goal of the server is to balance load across multiple NSPI servers, clients MUST NOT expect the same NSPI server to be returned from **RfrGetNewDSA**, even if all inputs are the same.

A client SHOULD call **RfrGetNewDSA** in the NSPIReferral service and connect to the NSPI server returned from that method. The client SHOULD NOT connect to an NSPI server without first requesting a server name from **RfrGetNewDSA**.<2>

The NSPI server returned in ppszServer MUST support the same RPC protocol sequence used by the RPC binding handle.

3.1.4.2 RfrGetFQDNFromServerDN (opnum 1)

The **RfrGetFQDNFromServerDN** method returns the Domain Name System (DNS) FQDNof the server corresponding to the passed DN.

[C++]

```
// opnum 1
long RfrGetFQDNFromServerDN(
   [in]         handle_t         hRpc,
   [in]         unsigned long      ulFlags,
   [in, range(10,1024)]   unsigned long     cbMailboxServerDN,
   [in, string, size_is(cbMailboxServerDN)]    unsigned char   *   szMailboxServerDN,
   [out,ref,string]    unsigned char   **   ppszServerFQDN);
```

hRpc: An RPC binding handle parameter, as specified in [C706] section 2. MUST NOT be NULL.

ulFlags: An unsigned long value, containing a set of bit flags. Unused; SHOULD be set to zero. Other values MUST be ignored by the server.

cbMailboxServerDN: An unsigned long value containing the number of bytes in the **szMailboxServerDN** string, including terminating NULL. The value is at least 10, at most 1024.

szMailboxServerDN: A 5 or 6-element DN identifying a mailbox server, which MUST match the server's implementation of server identities. It follows this format:

```
"/o=" organization-name "/ou=" administrative-group-name
"/CN=configuration/CN=servers/CN=" instance-name "/CN=" short-messaging-
server-name
```

The `CN=" instance-name "` element is optional.<3>

Note that the client MAY receive a DN identifying a specific database on this server, from sources listed in section 1.6. This DN follows this format:

```
"/o=" organization-name "/ou=" administrative-group-name
"/CN=configuration/CN=servers/CN=" instance-name "/CN=" short-messaging-
server-name "/CN=Microsoft Private MDB"
```

Or

```
"/o=" organization-name "/ou=" administrative-group-name
"/CN=configuration/CN=servers/CN=" instance-name "/CN=" short-messaging-
server-name "/CN=Microsoft Public MDB"
```

If this is the DN available, it is the client's responsibility to remove the final element before passing the DN to RfrGetFQDNFromServerDN.

ppszServerFQDN: A string. If the server does not return an error, ppszServerFQDN contains the FQDN of the mailbox server identified by szMailboxServerDN.

Return Values: The server returns 0 for a successful execution. An error results in an HRESULT or other nonzero error code.

Exceptions Thrown: No exceptions are thrown beyond those thrown by the underlying RPC protocol as specified by [MS-RPCE].

The server MUST process the data from the client to the following constraints when receiving this **message**. NSPIReferral MUST perform some lookup to determine the FQDN of the server identified by szMailboxServerDN. After considering these constraints, NSPIReferral SHOULD return one mailbox server name in ppszServerFQDN and 0 as a return value. If any errors occur and NSPIReferral is not able to return the name of a mailbox server, a failing HRESULT SHOULD be returned.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

NSPIReferral is a simple protocol that is well-explained by the following example:

Client requests an NSPI server name from the server by calling RfrGetNewDSA() with pUserDN set to the client's mailbox DN.

Typical parameters will look like the following:

```
    // RPC handle returned by RPC binding functions
    hRpc
       0x00010480   handle_t
    ulFlags
       0x00000000   unsigned long
    pUserDN
       "/o=First Organization/ou=Exchange Administrative Group
(FYDIBOHF23SPDLT)/cn=Recipients/cn=user1"      unsigned char *
    ppszUnused
       0x00000000   unsigned char * *
    // memory address which will receive output string
    ppszServer
       0x62348000   unsigned char * *
```

Server responds to the RfrGetNewDSA call with return code 0 and a valid server name.

Typical parameters will look like the following:

```
    ppszServer   "server1.example.com"   unsigned char * *
```

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to NSPIReferral. General security considerations pertaining to the underlying RPC-based transport apply (see [MS-RPCE]). NSPIReferral usually requires authentication, but generally does not restrict any caller who is authenticated.

5.2 Index of Security Parameters

Security parameter	Section
Protocol Sequences	2.1

6 Appendix A: Full IDL

For ease of implementation, the full IDL is provided below. The syntax uses the IDL syntax extensions as specified in [MS-RPCE] sections 2.2.4 and 3.1.5.1. For example, as specified in [MS-RPCE] section 2.2.4.9, a pointer_default declaration is not required and pointer_default(unique) is assumed.

rfri.idl:

```
[ uuid (1544f5e0-613c-11d1-93df-00c04fd7bd09),
  version(1.0),
  pointer_default(unique)]
interface rfri
{
long RfrGetNewDSA(
 [in]         handle_t       hRpc,
 [in]             unsigned long   ulFlags,
 [in, string]    unsigned char *   pUserDN,
 [in,out,unique, string]    unsigned char * * ppszUnused,
 [in,out,unique, string]    unsigned char * * ppszServer);

long RfrGetFQDNFromServerDN(
   [in]         handle_t         hRpc,
   [in]         unsigned long      ulFlags,
   [in, range(10,1024)]   unsigned long     cbMailboxServerDN,
   [in, string, size_is(cbMailboxServerDN)]    unsigned char *   szMailboxServerDN,
   [out,ref,string]    unsigned char   **   ppszServerFQDN);
}
```

7 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products:

- Microsoft® Office Outlook® 2003
- Microsoft® Exchange Server 2003
- Microsoft® Office Outlook® 2007
- Microsoft® Exchange Server 2007
- Microsoft® Outlook® 2010
- Microsoft® Exchange Server 2010

Exceptions, if any, are noted below. If a service pack number appears with the product version, behavior changed in that service pack. 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 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 product does not follow the prescription.

<1> Section 3.1.4.1: The Exchange 2003 and Exchange 2007 implementation of NSPIReferral follow these NSPI server preference rules: NSPI servers have four properties:

1. Server is up and functioning.
2. Server supports the client's protocol sequence.
3. Server has a writeable copy of the object represented by pUserDN.
4. Server is physically close to NSPIReferral server.

The NSPI servers are compared on these four properties in the order above. If two servers both satisfy or both do not satisfy 1, then 2 is used as a tie-breaker; if two servers both satisfy or both do not satisfy 1 and both satisfy or both don't satisfy 2, then 3 is used as a tie-breaker; and so on. The server that breaks the tie by satisfying a property that the other one does not satisfy is the preferred server. If multiple servers tie after comparing all four properties, those servers are returned in "round robin" order, meaning that each call to RfrGetNewDSA will return the next server in the list of tied servers. In the Exchange 2003 and Exchange 2007 implementation of NSPIReferral, the administrator of the NSPIReferral service can configure NSPIReferral to reverse the priorities of properties 3 and 4.

<2> Section 3.1.4.1: Office Outlook 2003 and Office Outlook 2007 can connect to a messaging server with a co-located NSPI server and no NSPIReferral server, as well as a messaging server with an NSPIReferral server. When first connecting, Office Outlook 2003 and Office Outlook 2007 have not yet determined which type of messaging server they are connecting to, and therefore they will try to connect to the messaging server's co-located NSPI server. On subsequent connections to that server, Office Outlook 2003 and Office Outlook 2007 will use NSPIReferral. This is one exception to the protocol documentation that states that clients SHOULD always use NSPIReferral. Clients written to this protocol documentation have no reason to connect to an NSPI server before using NSPIReferral.

<3> Section 3.1.4.2: In Exchange 2003 and Exchange 2007, the CN=" instance-name " element is not supported.

8 Change Tracking

This section identifies changes that were made to the [MS-OXABREF] protocol document between the August 2010 and November 2010 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.
- Changes made for template compliance.
- Removal of a document from the documentation set.

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

The revision class **Editorial** means that the 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 revision 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
1.4 Relationship to Other Protocols	58626 Removed links to specific sections in [MS-RPCE].	N	Editorially updated.
3.1.4.1 RfrGetNewDSA (opnum 0)	57408 Removed product behavior note from the ppszServer description. Updated the product behavior note in the Exceptions Thrown description and updated the text preceding the product behavior note about prioritization of NSPI servers.	Y	Product behavior note updated.

9 Index