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

Intellectual Property Rights Notice for Protocol Documentation

- **Copyrights.** This protocol 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 protocols, and may distribute portions of it in your implementations of the protocols 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 protocol documentation.
- **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 protocols. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, the protocols may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp). If you would prefer a written license, or if the protocols are not covered by the OSP, patent licenses are available by contacting protocol@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.

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. This protocol documentation is 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. A protocol specification does 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.

Revision Summary			
Author	Date	Version	Comments
Microsoft Corporation	April 4, 2008	0.1	Initial Availability.
Microsoft Corporation	April 25, 2008	0.2	Revised and updated property names and other technical content.
Microsoft Corporation	June 27, 2008	1.0	Initial Release.
Microsoft Corporation	August 6, 2008	1.01	Revised and edited technical content.
Microsoft Corporation	September 3, 2008	1.02	Updated references.

Microsoft Corporation	December 3, 2008	1.03	Updated IP notice.

Table of 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
distinguished name (DN)
domain
Domain Name System (DNS)
dynamic endpoint
endpoint (2)
Interface Definition Language (IDL)
Name Service Provider Interface (NSPI)
Network Data Representation (NDR)
remote procedure call (RPC)
RPC protocol sequence
security provider
Universal Unique Identifier (UUID)

The following terms are specific to this document:

directory service (DS): A service that stores and organizes information about a computer network's users and resources, and that allows network administrators to manage network resources, users, and their access to network resources.

flags: A set of values used to configure or report options or settings.

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

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

[MS-ERREF] Microsoft Corporation, "Windows Error Codes", January 2007, http://go.microsoft.com/fwlink/?LinkId=112243.

[MS-NSPI] Microsoft Corporation, "Name Service Provider Interface (NSPI) Protocol Specification", June 2008.

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

[MS-RPCE] Microsoft Corporation, "Remote Procedure Call Protocol Extensions", July 2006, http://go.microsoft.com/fwlink/?LinkId=112246.

[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

[MSDN-MIDL] Microsoft Corporation, "Microsoft Interface Definition Language (MIDL)", http://go.microsoft.com/fwlink/?LinkId=112244.

[MSDN-RPC] Microsoft Corporation, "Remote Procedure Call", http://go.microsoft.com/fwlink/?LinkId=112245.

1.3 Protocol 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 DNS FQDN of a mailbox server. Figure 1 shows the RfrGetNewDSA method call. Figure 2 shows the RfrGetFQDNFromServerDN method call.

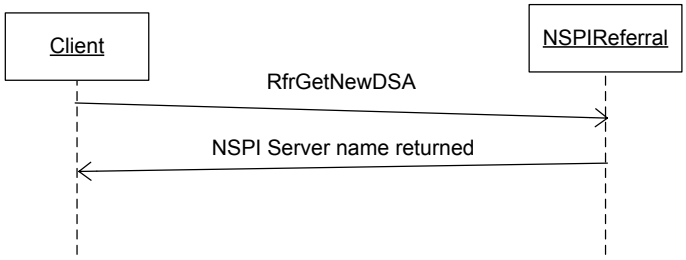

Figure 1: Relationship between client and NSPIReferral

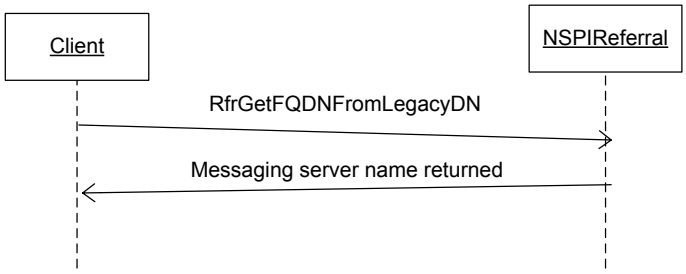

Figure 2: Client retrieving mailbox server name from NSPIReferral

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:

1) 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.

2) When connecting to the wrong mailbox or public folder server, an error will be returned containing the DN of the correct server.

3) When connecting to another user's mailbox, having only the **PidTagAddressBookHomeMessageDatabase** property, as specified in [MS-OXOABK], for that mailbox.

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. These authentication methods are defined in section 3.1.4.

Capability negotiation is performed by the client.

1.8 Vendor-Extensible Fields

RfrGetNewDSA and **RfrGetFQDNFromServerDN** return HRESULT values. Vendors can define their own HRESULT values, provided they set the C bit (0x20000000) for each vendor-defined value, indicating that the value is a customer code.

RfrGetNewDSA can also return other error values. Any non-zero return code indicates an error.

1.9 Standards Assignments

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

2 Messages

2.1 Transport

NSPIReferral works over the protocol sequences listed in the following table.

Protocol sequence
ncacn_ip_tcp
ncacn_http

NSPIReferral uses a well-known **endpoint** for network protocol sequence "ncacn_http". The following well-known endpoint is used:

Server	Protocol sequence	Endpoint
NSPIReferral	ncacn_http	6002

For all other network protocol sequences, the protocol uses **RPC dynamic endpoints** as specified in Part 4 of [C706]. These endpoints SHOULD be 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 NSPIReferral.

NSPIReferral uses SSP security. It supports NTLM (RPC_C_AUTHN_WINNT), and Negotiate (RPC_C_AUTHN_GSS_NEGOTIATE) **security providers**. 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 **NDR** transfer syntax only, as specified in Part 4 of [C706].

NSPIReferral makes use of RPC base types and definitions as specified in [C706] and [MS-RPCE].

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**, 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 DNS FQDN, as specified in [RFC 1035], of the server corresponding to the passed DN. 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 section 2 of [C706].

ulFlags: An unsigned long value, containing a set of bit **flags**. Unused; SHOULD be set to zero. Other values MUST be ignored by 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, as specified in [RFC1035], of an NSPI server <1>. On failure, the value is undefined.

Return Values: The server returns 0 for a successful execution. An error results in an HRESULT or other non-zero 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 <2>. The server MAY take other constraints into account, such as 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 non-zero 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** <3>.

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

3.1.4.2 RfrGetFQDNFromServerDN (opnum 1)

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

hRpc: An **RPC binding** handle parameter, as specified in section 2 of [C706]. 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 **szLegacyDN** string, including terminating NULL. The value is at least 10, at most 1024.

szMailboxServerDN: A DN identifying a mailbox server <4>.

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

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 *szLegacyDN*. 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:

1. 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 * *
```

2. Server responds to **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 SHOULD require authentication but SHOULD NOT restrict any caller who is authenticated.

5.2 Index of Security Parameters

Security parameter	Section
DS Referral authentication protocols	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 2.2.4.8,** 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: Office/Exchange Behavior

The information in this specification is applicable to the following versions of Office/Exchange:

- Office 2003 with Service Pack 3 applied
- Exchange 2003 with Service Pack 2 applied
- Office 2007 with Service Pack 1 applied
- Exchange 2007 with Service Pack 1 applied

Exceptions, if any, are noted below. Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies Office/Exchange behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies Office/Exchange does not follow the prescription.

<1> Section 3.2.4.1: Exchange 2007 SP1 and Exchange 2003 SP2: The Exchange implementation of NSPIReferral returns an NSPI server that listens on the **endpoint** 6004 for the ncacn_http protocol sequence. This works around network topology considerations which make it technically difficult to expose the NSPI servers directly to NSPI clients outside the local network.

<2> Section 3.2.4.1: Exchange 2007 SP1 and Exchange 2003 SP2: The Exchange implementation of NSPIReferral follows 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 implementation of NSPIReferral, the administrator of the Exchange NSPIReferral service can configure NSPIReferral to reverse the priorities of properties 3 and 4.

<3> Section 3.1.4.1: Office 2007 SP1 and Office 2003 SP3: Outlook 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, Outlook has not yet determined which type of messaging server it is connecting to, and therefore it will try to connect to the messaging server's co-located NSPI server. On subsequent connections to that server, Outlook 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.

<4> Section 3.2.4.2: Exchange 2007 SP1 and Exchange 2003 SP2: The value in the *szMailboxServerDN* parameter MUST match the server's implementation of server identities. In Exchange, this is a 5-element DN. It follows this format:

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

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=" short-messaging-server-name
"/cn=Microsoft Private MDB"
```

Or

```
"/o=" organization-name "/ou=" administrative-group-name
"/cn=configuration/cn=servers/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**.

Index