[MS-TAIL]:
Telephony API Internet Locator Service 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 Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
07/20/2007	0.1	Major	MCPP Milestone 5 Initial Availability
09/28/2007	1.0	Major	Updated and revised the technical content.
10/23/2007	1.0.1	Editorial	Revised and edited the technical content.
11/30/2007	1.0.2	Editorial	Revised and edited the technical content.
01/25/2008	1.0.3	Editorial	Revised and edited the technical content.
03/14/2008	1.0.4	Editorial	Revised and edited the technical content.
05/16/2008	1.0.5	Editorial	Revised and edited the technical content.
06/20/2008	1.0.6	Editorial	Revised and edited the technical content.
07/25/2008	1.0.7	Editorial	Revised and edited the technical content.
08/29/2008	1.1	Minor	Updated the technical content.
10/24/2008	1.1.1	Editorial	Revised and edited the technical content.
12/05/2008	1.2	Minor	Updated the technical content.
01/16/2009	1.3	Minor	Updated the technical content.
02/27/2009	1.3.1	Editorial	Revised and edited the technical content.
04/10/2009	1.3.2	Editorial	Revised and edited the technical content.
05/22/2009	2.0	Major	Updated and revised the technical content.
07/02/2009	3.0	Major	Updated and revised the technical content.
08/14/2009	4.0	Major	Updated and revised the technical content.
09/25/2009	5.0	Major	Updated and revised the technical content.
11/06/2009	5.1	Minor	Updated the technical content.
12/18/2009	6.0	Major	Updated and revised the technical content.
01/29/2010	6.0.1	Editorial	Revised and edited the technical content.
03/12/2010	7.0	Major	Updated and revised the technical content.
04/23/2010	8.0	Major	Updated and revised the technical content.
06/04/2010	9.0	Major	Updated and revised the technical content.
07/16/2010	10.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
08/27/2010	11.0	Major	Significantly changed the technical content.
10/08/2010	11.0	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	11.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	11.0	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	11.1	Minor	Clarified the meaning of the technical content.
03/25/2011	12.0	Major	Significantly changed the technical content.
05/06/2011	12.0	No change	No changes to the meaning, language, or formatting of the technical content.
06/17/2011	12.1	Minor	Clarified the meaning of the technical content.

Contents

1 Introduction 7
1.1 Glossary 7
1.2 References 8
1.2.1 Normative References 8
1.2.2 Informative References 9
1.3 Overview 9
1.4 Relationship to Other Protocols 10
1.5 Prerequisites/Preconditions 10
1.6 Applicability Statement 10
1.7 Versioning and Capability Negotiation 11
1.8 Vendor-Extensible Fields 11
1.9 Standards Assignments 11

2 Messages 12
2.1 Transport 12
2.2 Message Syntax 12
2.2.1 Schema 12
2.2.1.1 Schema Additions 12
2.2.1.2 Dynamic Objects 12
2.2.2 rtApplicationUser – The User of an Application 13
2.2.3 rtPerson – An Online Person 13
2.2.4 rtConference – An Online Conference 15
2.2.5 Name Mapping 16
2.2.6 ILS Variations from the LDAP v3 Protocol 16
2.3 ILS Schema Objects 16
2.3.1 rtApplicationUser (Object Class) 16
2.3.2 rtPerson (Object Class) 17
2.3.3 rtConference (Object Class) 17
2.3.4 ntSecurityDescriptor (Schema Attribute) 17
2.3.5 schemaIDGUID (Schema Attribute) 18

3 Protocol Details 19
3.1 Abstract Data Model 19
3.2 Timers 19
3.3 Initialization 19
3.4 Higher-Layer Triggered Events 19
3.5 Message Processing Events and Sequencing Rules 20
3.5.1 Time-to-Live (TTL) Attribute 20
3.5.2 LDAP Bind to ILS 20
3.5.2.1 Authentication Methods 20
3.5.3 Client Registration with ILS 20
3.5.4 Unregister from ILS 21
3.5.5 Change User Information 22
3.5.6 List Conferences 22
3.5.7 List Users 22
3.5.8 List ILS Servers in Active Directory 22
3.5.9 Publishing an Internet Locator Service to Active Directory 23
3.5.10 Unpublish (Remove) an ILS Server from Active Directory 24
3.5.11 Refresh Request 24
3.6 Timer Events 24

3.7 Other Local Events 25

4 Protocol Examples 26
4.1 N-Client Registration with ILS 26
4.1.1 ILS Registration LDAP Bind 27
4.1.2 ILS Registration Add Operation 27
4.1.3 ILS Registration Modify Operation 27
4.1.4 ILS Registration Unbind Operation 28
4.1.5 ILS Registration LDAP Sequence Diagram 28
4.2 N-Client Stay Alive Refresh 28
4.2.1 Stay Alive Refresh Bind 29
4.2.2 Stay Alive Refresh – Search 29
4.2.3 Stay Alive Refresh Unbind Operation 29
4.2.4 Stay Alive LDAP Sequence Diagram 29
4.3 N-Client - Find Online User 30
4.3.1 LDAP Find Online User Bind Operation 30
4.3.2 LDAP Find Online User LDAP Search Operation 31
4.3.3 LDAP Find Online User Unbind Operation 31
4.3.4 LDAP Find Online User LDAP Sequence Diagram 31
4.4 N-Client - Unregister 31
4.4.1 Unregister LDAP Bind Operation 31
4.4.2 Unregister LDAP Delete Operation 32
4.4.3 Unregister – LDAP Unbind Operation 32
4.4.4 Unregister LDAP Sequence Diagram 32
4.5 TAPI Client – Connect to ILS Server 32
4.5.1 LDAP Bind Operation 33
4.5.2 LDAP Add rtApplicationUser Operation 33
4.5.3 LDAP Modify rtApplicationUser Operation 33
4.5.4 LDAP Add rtPerson Operation 34
4.5.5 LDAP Modify rtPerson Operation 34
4.5.6 LDAP Unbind Operation 35
4.5.7 ILS Registration Sequence Diagram 35
4.6 TAPI Client – Stay Alive Refresh 35
4.6.1 TAPI Client – Stay Alive Refresh rtApplicationUser 35
4.6.2 TAPI Client – Stay Alive Refresh rtPerson 36
4.6.3 ILS Stay Alive Sequence Diagram 36
4.7 TAPI Client - Create Conference 36
4.7.1 LDAP Bind Operation 36
4.7.2 LDAP Verify Access Rights 37
4.7.3 LDAP Create Conference 37
4.7.4 LDAP Modify TTL for Conference 38
4.7.5 LDAP Unbind Operation 38
4.8 TAPI Client – Find Conferences 38
4.8.1 LDAP Bind Operation 39
4.8.2 LDAP Search Operation 39
4.8.3 LDAP Unbind Operation 39
4.8.4 ILS Find Conferences Sequence Diagram 39
4.9 TAPI Client – Find People 40
4.9.1 LDAP Bind Operation 40
4.9.2 LDAP Search Operation 40
4.9.3 LDAP Unbind Operation 40
4.9.4 ILS Find Users Sequence Diagram 41
4.10 TAPI Client – Disconnect from ILS Server 41

4.11 Sample LDAP Search Filters for ILS 41
4.11.1 LDAP Search Filters Used by the TAPI Client 41
4.11.2 LDAP Search Filters Used by the N-Client..... 41

5 Security..... 43
5.1 Security Considerations for Implementers..... 43
5.2 Index of Security Parameters 43

6 Appendix A: Product Behavior..... 44

7 Change Tracking..... 49

8 Index 51

1 Introduction

The Internet Locator Service (ILS) Protocol is an extension to the **Lightweight Directory Access Protocol (LDAP)**. This protocol uses LDAP-style requests to store and retrieve information in an **Internet Locator Service (ILS)** dynamic instance store, such as **people** or **conferences**. It is used for communication between collaboration clients using the **Telephony Application Programming Interface (TAPI)** and an **ILS Server**. The ILS is a dynamic directory service, primarily used to enable a **client** to find another user's network presence (usually this means the user's IP address) while online. Similar to how a person's telephone number is located in a telephone directory, a person's network presence can be contained in a computer directory such as ILS. The primary difference is that telephone numbers do not change very often, while a user's IP address often changes every time a user connects to the Internet/network. ILS can store information related to peer-to-peer and conference or multicast events.

ILS is used by two Microsoft Windows®-based clients: Microsoft® NetMeeting 3.01 and TAPI Dialer 1.00. TAIL was accessible via the Internet Locator Service API library supplied as part of the NetMeeting 3.01 software development kit (SDK).

1.1 Glossary

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

Active Directory
binary large object (BLOB)
Component Object Model (COM)
container
Coordinated Universal Time (UTC)
distinguished name (DN)(4)
domain controller (DC)
Domain Name System (DNS)
Dynamic Host Configuration Protocol (DHCP)
globally unique identifier (GUID)
Lightweight Directory Access Protocol (LDAP)
NetBIOS
SASL
Transmission Control Protocol (TCP)
Unicode
Universal Naming Convention (UNC)
Voice over IP (VoIP)

The following terms are specific to this document:

call: A form of online communication that can be peer to peer or conferencing.

client: A user participating in or intending to participate in collaboration.

conference: A set of two or more communicating users along with the software they are using to communicate.

Dynamic Directory Object: A dynamic entry is an object in a directory tree that has a time to live associated with it. This time to live is set when the entry is created. If dynamic entries are not refreshed within a given time-out, they may be removed from the directory.

ILS Server: Synonymous with **Internet Locator Service (ILS)**.

Internet Locator Service (ILS): A service used for locating user IP addresses in **Voice over IP (VoIP)**.

people: Users participating in a multimedia **conference**.

relative distinguished name (RDN): The name of an **object** relative to its parent. This is the leftmost attribute-value pair in the **distinguished name (DN)** of an **object**. For example, in the **DN** "cn=Peter Houston, ou=NTDEV, dc=microsoft, dc=com", the **RDN** is "cn=Peter Houston". For more information, see [RFC2251].

session: A set of multimedia senders and receivers and the data streams flowing from senders to receivers. A multimedia **conference** is an example of a multimedia session.

Telephony Application Programming Interface (TAPI): A **Component Object Model (COM)** interface used for the development of communications applications.

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

1.2 References

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

1.2.1 Normative References

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

[H323] ITU-T, "Packet-based multimedia communications systems", Recommendation H.323, June 2006, http://www.itu.int/rec/T-REC-H.323-200606-S/en

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

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

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

[MS-NLMP] Microsoft Corporation, "NT LAN Manager (NTLM) Authentication Protocol Specification".

[RFC1781] Kille, S., "Using the OSI Directory to Achieve User Friendly Naming", RFC 1781, March 1995, http://www.ietf.org/rfc/rfc1781.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

[RFC2251] Wahl, M., Howes, T., and Kille, S., "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997, http://www.ietf.org/rfc/rfc2251.txt

[RFC2252] Wahl, M., Coulbeck, A., Howes, T., and Kille, S., "Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions", RFC 2252, December 1997, http://www.ietf.org/rfc/rfc2252.txt

[RFC2254] Howes, T., "The String Representation of LDAP Search Filters", RFC 2254, December 1997, http://www.ietf.org/rfc/rfc2254.txt

[RFC2256] Wahl, M., "A Summary of the X.500(96) User Schema for use with LDAPv3", RFC 2256, December 1997, http://www.ietf.org/rfc/rfc2256.txt

[RFC2327] Handley, M., and Jacobson, V., "SDP: Session Description Protocol", RFC 2327, April 1998, http://www.ietf.org/rfc/rfc2327.txt

[RFC2589] Yaacovi, Y., Wahl, M., and Genovese, T., "Lightweight Directory Access Protocol (v3): Extensions for Dynamic Directory Services", RFC 2589, May 1999, http://www.ietf.org/rfc/rfc2589.txt

[RFC4512] Zeilenga, K., "Lightweight Directory Access Protocol (LDAP): Directory Information Models", RFC 4512, June 2006, http://www.rfc-editor.org/rfc/rfc4512.txt

1.2.2 Informative References

[Butler] Butler, P., Cales, R., Petersen, J., et al., "Using Microsoft Commercial Internet System: The Internet Locator Service Chapter 10", Que Pub; Special edition, April 1997, ISBN-13: 978-0789710161.

[LDAP] Microsoft Corporation, "About Lightweight Directory Access Protocol", http://msdn.microsoft.com/en-us/library/aa366075.aspx

If you have any trouble finding [LDAP], please check here.

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

[MSDN-ADDS] Microsoft Corporation, "Service Publication", http://msdn.microsoft.com/en-us/library/ms677950(VS.85).aspx

[MSDN-InternetLocSrvAPI] Microsoft Corporation, "Internet Locator Service API",http://msdn.microsoft.com/en-us/library/ms707540.aspx

[MSDN-MSTelephonyOvw] Microsoft Corporation, "Microsoft Telephony Overview", http://msdn.microsoft.com/en-us/library/ms733433.aspx

[MSDN-WSALookupServiceBegin] Microsoft Corporation, "WSALookupServiceBegin Function", http://msdn.microsoft.com/en-us/library/ms741633.aspx

[MSFT-SP] Microsoft Corporation, "How Service Publication and Service Principal Names Work", March 2003, http://technet.microsoft.com/en-us/library/cc755804(WS.10).aspx

[X501] ITU-T, "Information Technology - Open Systems Interconnection - The Directory: The Models", Recommendation X.501, August 2005, http://www.itu.int/rec/T-REC-X.501-200508-I/en

1.3 Overview

This document describes the following:

- How this protocol differs (the nonstandard behavior) from the requirements of LDAP [RFC2251], [RFC2252], and [RFC4512] (the LDAP RFC in place at the time of development of Internet Locator Service (ILS)).
- The operations used to manage conferences, people, classes, and attributes that are stored in ILS.

The ILS provides a real-time dynamic directory service (see [Butler] for a historical discussion). This allows online clients who are currently actively connected to the Internet to find other clients and to contact them for collaboration activity such as establishing a peer-to-peer session or joining a multicast conference. Clients who wish to be located when online register with the ILS and provide their name, e-mail address, and some personal information. During registration the connection point (usually an IP address) is associated and stored with the personal details. Other clients can connect to ILS and browse the list of registered clients looking for a specific person. They can then launch their client collaboration application, such as Microsoft® NetMeeting, which initiates a real time connection to the selected person.

The Telephony API Internet Locator Service protocol uses LDAP-style (see [LDAP]) requests to store, retrieve, and modify information in the ILS, such as people or conferences. Once a person knows the network presence of another they wish to contact, they can place a **call** to that person's computer. This call can be live chat, instant messages, shared white boarding, video conferencing, or even just a voice call.

This protocol is used by NetMeeting and TAPI clients to interact with an ILS server.

TAPI is a **Component Object Model (COM)** interface used for the development of communications applications. When communicating with an ILS server, only the client-side functions in the TAPI interface are used.

ILS is a dynamic directory service that associates people and conferences with the IP addresses of their computers. ILS maintains the correct IP address for people or conferences even when their IP address changes, which is particularly useful when people use **Dynamic Host Configuration Protocol (DHCP)**. To do so, ILS provides an LDAP interface and supports dynamic objects, as specified in [RFC2589].<1>

For a historical overview of this protocol, see [MSDN-InternetLocSrvAPI], [MSDN-MSTelephonyOvw], and [MSDN-WSALookupServiceBegin].

1.4 Relationship to Other Protocols

This protocol is an early implementation of the LDAP protocol and supports syntax and operations that do not form part of the current LDAP RFCs.

An ILS can support anonymous or authenticated users. ILS uses the LDAP Authentication mechanisms Simple Authentication and Sicily Authentication as specified in [MS-ADTS] section 5.1.1.1.3.<2>

1.5 Prerequisites/Preconditions

The TAPI Internet Locator Service Protocol assumes the availability of the following resources:

- A transport protocol, either **TCP** IPv4 or **NetBIOS** over TCP, to support reliable, in-order message delivery.
- LDAP **SASL** mechanisms for authentication.

1.6 Applicability Statement

This protocol is used by clients to query ILS servers for people and conference data, which is then used in collaboration activities.

1.7 Versioning and Capability Negotiation

This document covers versioning in the following areas:

- Protocol Versions: This protocol supports Binds using both LDAP v2 and LDAP v3.

1.8 Vendor-Extensible Fields

There are no vendor-extensible fields in this protocol.

1.9 Standards Assignments

ILS does not use any IANA published ports.

ILS uses by default TCP port number 1002. This is subject to user configuration.

2 Messages

2.1 Transport

The TAPI ILS (TAIL) Protocol is a series of data exchanges between a collaboration client and an ILS server. The data that is exchanged using this protocol is transported using the LDAP v3 protocol (with some limitations described in section 2.2.6).

2.2 Message Syntax

This ILS protocol is used to store and retrieve data in the following ILS Dynamic Objects. The schema for these objects is given in section 2.3.

Dynamic Object Class	Description
rtApplicationUser	Contains information about an ILS application user.
rtPerson	Contains information about an ILS user.
rtConference	Contains information about an ILS conference.

2.2.1 Schema

An ILS Server requires a schema definition (see [RFC2256]) for an object class before it can store an instance of that object; this applies to both static and dynamic objects. ILS Servers have a default schema that includes both static Member and Group objects, plus the rtPerson, rtConference, and rtApplicationUser. The details of the attributes of rtPerson, rtConference, and rtApplicationUser are given later in this section. The full schema, including inherited objects, is given in ILS Schema Objects (section 2.3).

2.2.1.1 Schema Additions

All dynamic entries MUST have the dynamicObject value in their **objectClass** attribute. This object class is defined section 5 of [RFC2589].

Furthermore, each dynamic entry MUST have the operational attribute **entryTtl** as described in section 5 of [RFC2589].

2.2.1.2 Dynamic Objects

Any object in the ILS schema with dynamicObject value as part of the **objectClass** attribute of the schema object is a dynamic object. Each dynamic object has its own time to live (TTL) operational attribute that the server periodically decrements. The TTL can be refreshed (updated) by a client; the server MAY automatically delete a dynamic object when its TTL reaches zero. For example, a server is free to delete the object immediately when its TTL reaches zero or to employ a sweeper process that periodically deletes objects with TTL values of zero. Clients should not depend on the server behaving one way or the other.

The TTL applies to the entire object; ILS does not support a TTL per attribute. Each object does have its own TTL, which can be set individually. Thus persistent objects, such as conferences, can be set to last longer with less refresh overhead than more temporal objects, such as online users. ILS supports an administrator-configurable, system-wide limit on the maximum value of any object's TTL.

2.2.2 rtApplicationUser – The User of an Application

Attribute	Description
applicationID	Identifies the application by name.<3>
appName	Application Name.<4>
Application Name	Identifies the application by name.
groupObject	Not used.
guid	Contains a global identifier. Populated with "008aff194794cf118796444553540000".
ILSA26214430	Indicates whether or not a call is active. The value is "1" if in a call or "0" if not in a call.<5>
ILSA26279966	A generated version number.
ILSA32833566	Indicates the capability to send audio. The value is "1" if the client can send audio or "0" if it cannot send audio.<6>
ILSA32964638	Indicates the capability to send video. Clients populate this attribute with the capability of sending video. The value is "1" if they can send video; or "0" if they cannot send video.<7>
ILSA39321630	Indicates the restriction category. Clients populate this attribute with the restriction value. The value is "1" if it is primarily used for personal uses, "2" if it is primarily used for business, and "4" if it allows adult content. "3" is unused. The default value is "0".<8>
mimeType	Contains the data type of the call, set to "text/iuls".<9>
objectClass	Identifies the list of classes from which this object is derived.
port	A multivalue attribute. Typical attributes are the value "1503" (Data port) and "1720" (Audio/Video port).
protocolGUID	Contains the **GUID**, as specified in [C706] section A.1, set to "008aff194794cf118796444553540000".<10>
protocolID	Contains the protocol of the conference, set to [H323].
protocolMimeType	A multivalue attribute: Contains the data type of the protocol, set to the values "text/120" and "text/h23".
sFlags	Show me (in the directory) flags, 1=Show, 0=Hidden.
userObject	Identifies the user.

2.2.3 rtPerson – An Online Person

Attribute	Description
c	This attribute is the two-letter code from ISO:3166:1999. <11>
cn	Contains the identity of the user.<12>

Attribute	Description
comment	Contains general comments about the conference.<13>
givenName	Contains the given name attribute for the user object that is being created. Clients populate this attribute with the name that the user supplied during setup or later.
guid	Contains a global identifier. "008aff194794cf118796444553540000".
ILSA26214430	Indicates whether or not a call is active. Clients populate this attribute with the status of being in a call. The value is "1" if in a call or "0" if not in a call.<14>
ILSA26279966	A build number of product 84020942 decimal is hexadecimal 5020ECE; the 5020 corresponds to build 5.2. ECE corresponds to 3790, hence build 5.2.3790.
ILSA32833566	Indicates the capability to send audio. Clients populate this attribute with the capability of sending audio. The value is "1" if it can send audio or "0" if it cannot send audio.<15>
ILSA32964638	Indicates the capability to send video. Clients populate this attribute with the capability of sending video. The value is "1" if they can send video; or "0" if they cannot send video.<16>
ILSA39321630	Indicates the restriction category. Clients populate this attribute with the restriction value. The value is "1" if it is primarily used for personal uses, "2" if it is primarily used for business, and "4" if it allows adult content. "3" is unused. The default value is "0".<17>
ipAddress	Contains a calculated value for the IP address of the user. Clients populate this attribute with a string that represents the IP4 address of the machine. For instance, if the IP address is 10.70.41.96, the string contains 1613317642. The value is calculated as 10 + 256(70 + 256(41 + 256 * 96)). TAPI **clients** do not populate this attribute.
location	A text field entered by users to indicate their location.
mimeType	Contains the data type of the call, set to "text/iuls"
o	Identifies the organization that the user belongs to.
port	A multivalue attribute: Typical attributes are the value "1503" (Data port) and "1720" (Audio/Video port).<18>
protocolMimeType	A multivalue attribute: Contains the data type of the protocol, set to the values "text/120" and "text/h23".
rfc822mailbox	Contains the e-mail address that the user supplied.<19>
sFlags	Show me (in the directory) flags, 1=Show, 0=Hidden.
smodop	Contains the ILS special modify-operation code during an LDAP Modify Request.<20> Several types of modification operations can be performed on the LDAP server: adding an application, deleting an application, modifying user information, and

Attribute	Description
	modifying application information. The type of modification is specified by using the appropriate **smodop** attribute. The value is either "0" (ADD), "1" (DELETE), "2" (MODIFYUSER), or "3" (MODIFYAPP).
surName	Contains the last name of the user.<21>
securityToken	Contains the securityTokenID value of the user.<22>

2.2.4 rtConference – An Online Conference

Attribute	Description
advertisingScope	This attribute is not used. Indicates whether an entry should be advertised outside a corporate gateway or proxy.
announcementScope	Not used.
applicationID	ms-netmeeting
attendees	Not used.
attendeesCount	Not used.
confAddr	Not used.
conferenceBlob	Contains a Session Description Protocol (SDP) binary large object (BLOB), as specified in [RFC2327].
contactInformation	Not used.
generalDescription	Contains a comment for the conference.
isEncrypted	Set to "1" if the conference is encrypted; otherwise, set to "0".
maxAttendeesCount	Not used.
originator	Contains the name of the user that created this conference.
protocolID	Contains the protocol that the conferenceBlob attribute uses to describe the conference.
rating	Not used.
startTime	Contains the **Coordinated Universal Time (UTC)** when the conference becomes available to users.
stopTime	Contains the UTC when the conference becomes unavailable to users (they are unable to join this conference).
subType	Not used.
url	Not used.
uid	Contains the unique identifier of the conference. Entered by the user.

2.2.5 Name Mapping

For backward compatibility with previous locator products, there have been name changes to schema entries. The following name mapping applies to LDAP searches and the corresponding field matched. The name on the left is seen in LDAP messages and corresponds to the attribute named on the right.

LDAP message name	Attribute
sAppGUID	guid
sAppID	applicationID
sAppName	AppName
sIPAddress	IPAddress
sMimeType	MimeType
sPort	Port
sProtGuid	protocolGuid
sProtId	protocolID
sProtMimeType	protocolMimeType
sSecurity	securityToken
sTTL (value in minutes)	entryTTL (value in seconds)

2.2.6 ILS Variations from the LDAP v3 Protocol

ILS communication differs from LDAP v3 in the following ways:

- ILS does not support modify **distinguished name** (ModifyDN) requests. To move entries from one part of the directory to another, delete the entries and add them to the desired location.
- The ILS LDAP server does not support server-side sorting.
- The ILS LDAP server does not support TAPI **session** control operations.
- The ILS LDAP server does not support friendly distinguished names (DNs), as specified in [RFC1781].

2.3 ILS Schema Objects

The details below include inherited objects that were not described in section 2.2.<23>

2.3.1 rtApplicationUser (Object Class)

Type	Object
General	OID 1.2.840.113556.1.4.611.1103010708051503150108150101130009121504000012000415120210000708 Name rtApplicationUser

Type	Object
Properties	Superior top KindStructural (0x01)
Required attributes	cn, objectClass
Optional attributes	appName, applicationID, groupObject, guid, ILSA26214430, ILSA26279966, ILSA32833566, ILSA32964638, ILSA39321630, mimeType, port, protocolGUID, protocolID, protocolMimeType, sFlags, userObject

2.3.2 rtPerson (Object Class)

Type	Object
General	OID 1.2.840.113556.1.4.611.1103010708051504150108150101130009121504000012000415120210000708 Name rtPerson
Properties	Superior top KindStructural (0x01)
Required attributes	cn, objectClass
Optional attributes	appName, c, comment, dwFlags, friendlyName, givenName, guid, ILSA26214430, ILSA26279966, ILSA32833566, ILSA32964638, ILSA39321630, ipAddress, location, mimeType, o, port, protocolMimeType, rfc822Mailbox, sFlags, securityToken, smodop, surName

2.3.3 rtConference (Object Class)

Type	Object
General	OID 1.2.840.113556.1.4.611.1103010708051501150108150101130009121504000012000415120210000708 Name rtConference
Properties	Superior top KindStructural (0x01)
Required attributes	objectClass, uid, cn
Optional attributes	advertisingScope, announcementScope, applicationID, attendees, attendeesCount, confAddr, conferenceBlob, contactInformation, generalDescription, isEncrypted maxAttendeesCount, originator, protocolID, rating, startTime, stopTime, subtype, url

2.3.4 ntSecurityDescriptor (Schema Attribute)

ntSecurityDescriptor is an attribute Schema Mandatory attribute.

Type	Object
cn	cn=ntSecurityDescriptor,cn=Schema,ou=Admin,o=Intranet
objectClass	attributeSchema
displayName	security-descriptor
u2AttributeID	1.2.840.113556.1.5.108.1103010708070215150108150101130009121504000012000415120210000708
description	The security descriptor for the object
schemaIDGUID	Unique GUID for this attribute<24>
attributeSyntax	Binary
isSingleValued	1
isSearchable	0

2.3.5 schemaIDGUID (Schema Attribute)

schemaIDGUID is an attribute Schema Mandatory attribute.

Type	Object
cn	cn=schemaIDGUID,cn=Schema,ou=Admin,o=Intranet
objectClass	attributeSchema
displayName	schema-id-guid
u2AttributeID	1.2.840.113556.1.4.148
description	GUID of a class or attribute definition
schemaIDGUID	Unique GUID for this attribute<25>
attributeSyntax	Octet String
isSingleValued	1
isSearchable	0

3 Protocol Details

The Internet Locator Service (ILS) Protocol uses LDAP to connect, disconnect, and modify user information and list conference entries in a directory. The LDAP used by the ILS Protocol differs from standard LDAP v3. The specific differences are described in section 2.2.6. Note that it is possible to use LDAP v3 to perform equivalent actions to the nonstandard operations.

3.1 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 document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

Collaboration clients use an ILS Server (LDAP dynamic object store) to store and retrieve information about other potential collaboration partners. ILS provides a client/server mechanism with which online users can locate each other and thus allow them to start a collaboration session. The basic idea behind ILS is simple: Two users running compatible communication applications want to communicate. For the two users to start the communication session, they first need to identify each other on the network.

A caller needs a mechanism to identify a public IPv4 address of a collaboration partner. Thus, ILS provides the necessary mechanism to handle the discovery and resolution of the IP addresses. On a more abstract level, ILS acts as a dynamic user directory that provides a rendezvous mechanism for users. This rendezvous mechanism allows users to find each other on a network.

At a database level, the ILS Server stores two main types of objects: rtPerson objects representing the online users, and rtConference objects representing data on available online conferences.

3.2 Timers

The ILS Server maintains an entryTTL attribute. This attribute has a value that is a TTL marker for dynamic objects. An LDAP search request for the attribute "sttl" (name mapped) in the form "sttl=<value>" resets the timer to the given value. If the timer goes to zero, the dynamic entry MAY be removed from the database by the server. Client applications need to perform occasional refreshes of the attribute value to ensure that the server maintains the dynamic objects.

An ILS client maintains a timer that mirrors the value sent to the ILS Server. The client MUST refresh the TTL value of dynamic objects. If, on a refresh search, it fails to find a dynamic object, the client MUST register the dynamic object.

3.3 Initialization

The ILS Protocol imposes no initialization requirements beyond those of standard LDAP v3. Dynamic objects created in the ILS Server will require their TTL attribute to be initialized on creation of an object in the directory.

3.4 Higher-Layer Triggered Events

None.

3.5 Message Processing Events and Sequencing Rules

The general model adopted by this protocol is one of clients performing protocol operations against servers as per LDAP v3 [RFC2251]. In this model, a client transmits a protocol request describing the operation to be performed to an ILS Server. The ILS Server is then responsible for performing the necessary operation(s) in the directory. Upon completion of the operation(s), the server returns a response containing any results or errors to the requesting client.

3.5.1 Time-to-Live (TTL) Attribute

The ILS server maintains an **entryTTL** attribute that has a value that is a time-to-live marker for dynamic objects. An LDAP search request for the attribute sttl (name mapped to attribute **entryTTL**) of the form "sttl=<value>" resets the timer to the value. If the timer goes to zero, the dynamic entry MAY be removed from the database.

For details of the schema requirements for dynamicObject and **entryTTL**, see [RFC2589] section 5, Schema Additions.

3.5.2 LDAP Bind to ILS

ILS supports authenticated binds as per the LDAP v3 RFC (see [RFC2251]). The only accepted SASL mechanism is NTLM (Sicily Authentication).

Note that this also includes binds using a version number of 2 or 3.

3.5.2.1 Authentication Methods

ILS can be configured to support anonymous or authenticated users. ILS does not have user accounts in its static store. ILS does not support authenticated binds for security principals whose accounts are not stored in the **Active Directory** directory service. ILS uses the LDAP Authentication mechanisms Simple Authentication and Sicily Authentication as specified in [MS-ADTS]. On query using an LDAP of the root DSE, an ILS Server returns NTLM, as specified in [MS-NLMP], as the supported SASL mechanism.

3.5.3 Client Registration with ILS

Client registration with an ILS server is made in four distinct LDAP operations:

- LDAP Bind
- LDAP Add
- LDAP Modify
- LDAP Unbind

The registration MUST be initiated by making an LDAP bind request to an ILS server as specified in section 3.5.2. If LDAP v2 is offered, only simple authentication can be used. <26>This is the normally the case when users are located on the Internet. If LDAP v3 is offered with credentials, the SASL mechanism will be NTLM. The default server port is 1002.

Once a successful bind has been made (LDAP Bind Response, resultCode == 0), it MUST be followed by an LDAP Add operation [RFC2251]. The purpose of the Add operation is to create a dynamic entry of the named user in the directory. The LDAP entry named in the entry field of the Add request is as follows:

- c=-,o=Microsoft, cn=<the e-mail address of the user>, objectClass=rtPerson.

This is a modified LDAP entry as per section 2.2.6.It has the effect of creating the following dynamicObject in the directory:

- Cn=<the e-mail address of the user>, ou=Dynamic, o=Intranet

  (where the entry type is objectClass=rtPerson, objectClass=dynamicObject).

Once a successful Add operation has been performed (LDAP Add Response, resultCode == 0), an LDAP Modify operation MUST be performed as follows:

- ModifyRequest: Object:c=-, o=Microsoft, cn==<the e-mail address of the user>,objectClass=rtPerson

The following attributes of the user (rtPerson) are then modified as follows. Note that Name Mapping (section 2.2.5) may apply:

- modop: The show mode, to indicate whether the user is to be visible.
- sappid: applicationID is set to NetMeeting.
- smimetype: MimeType is set to text/iuls.
- sappguid: guid is set to 008aff194794cf118796444553540000.
- sprotid: protocolID is set to T120 AND H323.
- sprotmimetype: protocolMimeType is set to (text/t120) and (text/h232).
- sport: port attributes set to 1503 and 1720.

The entries made when connecting to ILS are **Dynamic Directory Objects** as defined in [RFC2589].

3.5.4 Unregister from ILS

Unregistration from an ILS Server by a client is made in three LDAP operations:

- LDAP Bind
- LDAP Delete
- LDAP Unbind

The unregistration process MUST be initiated by making an LDAP Bind to an ILS Server. If LDAP v2 is offered, only simple authentication can be used.<27> This is the normally the case when users are located on the Internet. Once a successful Bind has been made, it MUST be followed by an LDAP Delete operation. The Delete operation allows a client to request the removal of an entry from the directory. The Delete operation is as follows:

- DelRequest: c=-,o=Microsoft, cn=<the e-mail address of the user>,objectClass=rtPerson

This is a modified LDAP entry as per section 2.2.6, ILS differences from LDAP v3. It has the effect of removing the following two dynamicObjects from the directory:

- Cn=<the e-mail address of the user>, ou=Dynamic, o=Intranet

(where the entry type is objectClass=rtPerson, objectClass=dynamicObject).

- Cn= <the e-mail address of the user>, appName=MS-NetMeeting, ou=Applications, o=Intranet

  (where the entry type is objectClass=rtApplicationUser, objectClass=dynamicObject).

Upon receipt of a Delete Request, the ILS Server MUST attempt to perform the entry removal requested. The result of the Delete Request will be returned to the client in the Delete Response using a standard LDAP response.

The unregister is completed using an LDAP Unbind operation. The function of the Unbind operation is to terminate a protocol session. The Unbind operation has no response defined. Upon transmission of an Unbind request, a protocol client can assume that the protocol session is terminated. Upon receipt of an Unbind Request, a protocol server can assume that the requesting client has terminated the session and that all outstanding requests can be discarded, and can close the connection.

3.5.5 Change User Information

The ILS Server does not support ModifyDN requests, since all ILS entries are dynamic. Therefore to modify an entry, it is necessary to delete the original entry and then add a new entry. Changing user information requires the following LDAP operations:

- LDAP Bind
- LDAP Del
- LDAP Unbind
- LDAP Bind
- LDAP Add
- LDAP Modify
- LDAP Unbind

3.5.6 List Conferences

Clients can determine the available conferences by performing an LDAP search for the class rtConference. The LDAP search uses the filter (ObjectClass=rtConference) with a search DN ou=Dynamic,o=intranet. Active conferences will have a UID present.

(cn=rtConference,ou=Dynamic,o=Intranet)

3.5.7 List Users

Clients can determine available collaboration clients by performing an LDAP search for the class rtPerson. The LDAP search uses the filter (ObjectClass=rtPerson) with a search DN ou=Dynamic,o=intranet.

3.5.8 List ILS Servers in Active Directory

To determine the available ILS Servers in an Active Directory domain, it is necessary to query Active Directory for the Winsock Services and search for the GUID of the ILSServiceClass. If that class is found, it is assumed that the server is running the ILS service. (Note that in practice this may not always be accurate as this class may be present in machines from which ILS has been removed.)

The process to determine the availability of ILS Servers within Active Directory is as follows:

- LDAP Bind (v3 authenticated) to Active Directory LDAP Service (port 389)
- LDAP Search CN=WinsockServices,CN=System,DC=testdomain,DC=int
- Filter (objectClass=serverInstance)(CN=*)

  (serviceClassID=40:79: F1:C9:A7:79:D1:11:B0:08:00:C0:4F:C3:1: EE)

3.5.9 Publishing an Internet Locator Service to Active Directory

Service publication is the act of creating and maintaining data about one or more instances of a given service so that network clients can find and use the service. The presence of an active Internet Locator Service can be published to a Windows Active Directory Server for subsequent discovery by applications using standard LDAP v3 calls. An explanation of Service Publication is available at [MSFT-SP] and [MSDN-ADDS].

The process of "publishing" involves adding an entry of type serviceInstance to the available Windows Sockets(WinSock) Services within Active Directory. The serviceInstance object class is used by Windows Sockets (Winsock) Services that publish information about themselves by using registration and resolution (RnR). (For the schema of the class serviceInstance, see [MS-ADSC] section 2.219.)

This publishing is done under the container with **RDN** CN=WinsockServices within the 'System' well-known object (section 7.1.1.4.11 of [MS-ADTS]). For example, if the fully qualified domain name of an ILS server is ILSServer.testdomain.int, the serviceInstance object with RDN of CN=ILSServer.testdomain.int will be created under the container with DN:

cn=WinsockServices, cn=System, DC=testdomain, dc=int

- The LDAP server of an Active Directory domain can be obtained by querying the DNS SRV records for LDAP entries. (See [MS-ADTS] section 7.3.2.)
- To add the entry, it is necessary to bind with authorized credentials to the LDAP server on port 389 of the domain controller.

  The sequence of steps to publish an ILS running on a machine with a fully qualified domain name of ILSServer.testdomain.int are as follows:

  - A new container is added to cn=WinsockServices,cn=System,DC=testdomain,dc=int
  - The LDAP Add operation will have an entry as follows:
    - Entry: CN=ILSServer.testdomain.int,CN=WinsockServices,CN=System,DC=testdomain,DC=intPartial Attributes:
      - CN=(ILSServer.testdomain.int)
      - displayName=(ILSServer.testdomain.int)
      - objectClass=(serviceInstance)
      - serviceClassID=(40:79: F1:C9:A7:79:D1:11:B0:08:00:C0:4F:C3:1: EE)
      - serviceInstanceVersion=(05:00:00:00:01:00:00:00)

The UUID 40:79: F1:C9:A7:79:D1:11:B0:08:00:C0:4F:C3:1: EE is the unique identifier for the Internet Location Service.

Once the container entry has been added to the Active Directory LDAP repository, it is necessary to perform a further LDAP operation. The additional LDAP operation is to modify the attribute winsockAddresses of serviceInstance to represent the IPv4 address of the server that hosts the ILS. Details on the attribute winsockAddresses can be found in [MS-ADA3].

- The sequence is completed by performing an LDAP Unbind.

3.5.10 Unpublish (Remove) an ILS Server from Active Directory

To unpublish an ILS Server in Active Directory, perform an LDAP modify ([RFC2251]) using the delete operation on the WinsockAddresses attribute of the ILS Server object located in WinsockServices in Active Directory. For example, assuming an ILS Server called ils.testdomain.internal, the target attribute would be

```
CN=ils.testdomain.internal, CN=WinsockServices,CN=System,DC=testdomain,DC=internal
```

3.5.11 Refresh Request

To refresh the time-to-live attribute of any dynamic object stored in the ILS directory, an LDAP search operation is performed with sttl attribute as part of the LDAP filter. The full operation requires the following LDAP operations:

- LDAP Bind
- LDAP Search
- LDAP Unbind

The refresh request MUST be initiated by making an LDAP Bind Request to an ILS Server. If LDAP v2 is offered, only simple authentication can be used.<28> This is the normally the case when users are located on the Internet. Once a successful Bind has been made (LDAP Bind Response, resultCode == 0), it MUST be followed by an LDAP Search operation [RFC2251]. The search operation identifies the object whose sttl/entryTTL value needs to be refreshed.

To refresh the rtPerson object "mailto:cn= egruber@contoso.com" and set the time to live to 10 minutes, the search request would be as follows:

```
SearchRequest: BaseDN: objectClass=rtPerson, SearchScope: base Object
LDAPFilter Filter: (&(objectClass=rtPerson)(mailto:cn= egruber@contoso.com)(sttl=10))
```

The server will respond with two LDAP PDUs: Search Result Entry and returning the matched object followed by Search Result Done. The refresh is completed by performing an Unbind.

Note A standard LDAP modify operation on the dynamicObject attribute entryTTL can also be performed to reset the time to live. The value for sttl is given in minutes; the value for entryTTL is given in seconds.

3.6 Timer Events

As described in section 3.2, the ILS Server maintains an entryTTL attribute that has a value that is a TTL marker for dynamic objects in seconds. When the timer expires (counts down to zero) on a

dynamic object, an ILS Server MAY<29> remove it from the database by the server. Client applications need to perform occasional refreshes of the attribute to ensure that the server maintains the dynamic objects. The value of entryTTL can be reset by searching for the attribute sttl. Note that sttl values are in minutes, entryTTLs are in seconds.

3.7 Other Local Events

None.

4 Protocol Examples

4.1 N-Client Registration with ILS

When a client is started for the first time, the user is prompted to enter information that others can use to find them in a directory.

It is also possible for the user to enter a directory name. This can be a NetBIOS name, a DNS name, or an IPv4 address. Once this information is entered (it is not validated), the user can log on to the specified directory. It is possible to configure the client to log on to the named directory server on startup.

Figure 1: Client registration options

Logging on, or directory registration, is a four-step LDAP process. The first three steps obtain a server response; there is no response to the Unbind operation. The four LDAP operations are as follows:

1. LDAP Bind
2. LDAP Add
3. LDAP Modify
4. LDAP Unbind

4.1.1 ILS Registration LDAP Bind

The registration is initiated by making a LDAP Bind request to an ILS Server. If LDAP v2 is offered, then no authentication occurs. When no authentication is to be performed, then the simple authentication option must be chosen, and the password must be of zero length. <30>

The default port for an ILS Server is TCP port 1002.

When a successful bind has been made an LDAP Bind Response, with a resultCode == 0 is returned.

4.1.2 ILS Registration Add Operation

The LDAP Bind operation is followed by an LDAP Add operation [RFC2251]. The Add operation creates a dynamic entry of the named user in the directory. Using the example data in section 4.1.1, the LDAP entry named in the entry field of the Add Request is as follows:

```
c=-,o=Microsoft, cn=egruber@contoso.com, objectClass=rtPerson
```

Although this entry looks like LDAP, it is actually ILS-specific as described in section 2.2.5. ILS has some LDAP limitations and these manifest themselves at this point in the syntax.

The Add operation includes data for 15 attributes of the class rtPerson. These are initialized with the values shown. Details of all the attributes of the class rtPerson are given in section 2.2.3.

```
cn=( egruber@contoso.com )
givenname=( Eric )
surname=( Gruber )
rfc822mailbox=( egruber@contoso.com )
location=( Seattle )
comment=( Participant )
c=( - )
sipaddress=( 3758139584 )
sflags=( 1 )
ssecurity=( 21912384 )
ilsA26214430=( 0 )
ilsA26279966=( 84020942 )
ilsA32833566=( 1 )
ilsA32964638=( 1 )
ilsA39321630=( 2 )
```

4.1.3 ILS Registration Modify Operation

After a successful Add Operation has been performed (LDAP Add Response, resultCode == 0), an LDAP modify operation must be performed. The purpose of this modify is to update application-specific information related to NetMeeting options. Note that since ILS is a dynamic directory, after an Unbind has been performed entries cannot be modified but must be deleted and re-created if they need to be changed.

ModifyReqest: Object:c=-, o=Microsoft, mailto:cn=egruber@contoso.com, objectClass=rtPerson The following attributes of the user (rtPerson) are then modified as follows. Note that the name mapping in section 2.2.5 applies to the visible LDAP operation and the attribute names may differ.

smodop: The show mode, to indicate whether the user is to be visible in the directory.

sappid: The applicationID is set to ms-netmeeting.

The applicationID is set to ms-netmeeting: The MimeType is set to text/iuls.

sappguid: The GUID is set to 008aff194794cf118796444553540000.

sprotid: The protocolID attribute is set to T120 and H323.

sprotmimetype: The protocolMimeType attribute set to (text/t120) and (text/h232).

sport: The port attribute is set to 1503 and 1720.

The entries made when connecting to ILS are dynamic directory objects as defined in [RFC2589].If the entry is successfully updated a LDAP response of status SUCCESS is returned.

4.1.4 ILS Registration Unbind Operation

On completion of the three prior steps, an LDAP Unbind is the final operation.

4.1.5 ILS Registration LDAP Sequence Diagram

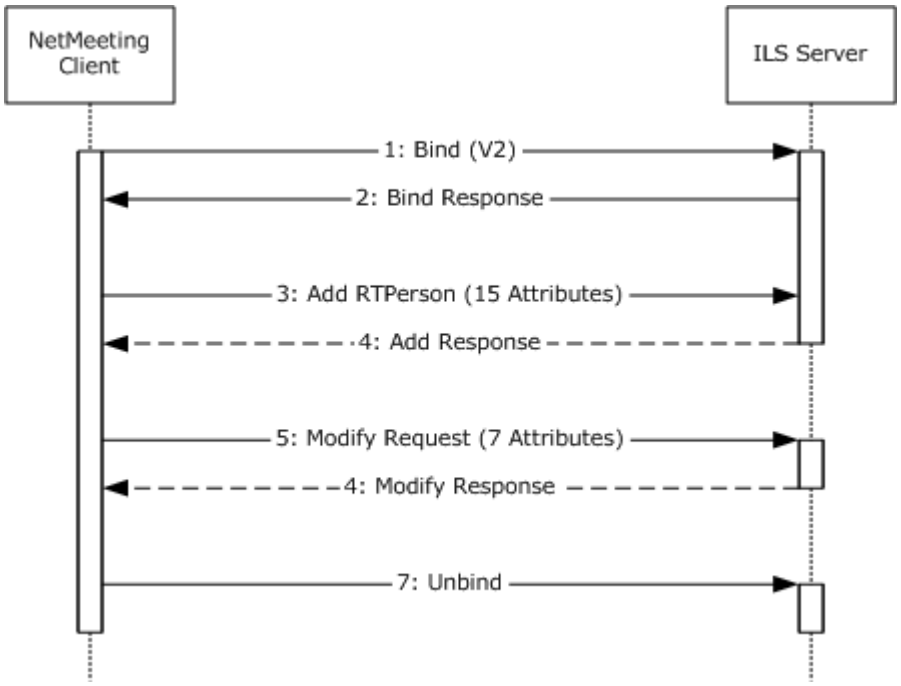

Figure 2: ILS registration LDAP sequence

4.2 N-Client Stay Alive Refresh

Periodically, the client must perform a refresh of the time-to-live (TTL) value for any user it has registered with the ILS Server. This process is performed as follows:

1. LDAP Bind

2. LDAP Search

3. LDAP Unbind

4.2.1 Stay Alive Refresh Bind

The TTL refresh is initiated by making a LDAP Bind request to an ILS Server. If LDAP v2 is offered, then no authentication happens. When no authentication is to be performed, the simple authentication option must be chosen, and the password must be of zero length.<31>

The default port for an ILS Server is TCP port 1002.

When a successful bind has been made, an LDAP Bind Response, with a resultCode == 0 is returned.

4.2.2 Stay Alive Refresh – Search

A search is then made for the specific rtPerson object associated with the registered user. For example:

BaseDN: objectClass=rtPerson, SearchScope: base Object, SearchAlias: neverDerefAliases

LDAPFilter Filter: (&(objectClass=rtPerson)(cn= egruber@contoso.com)(sttl=10))

- If the rtPerson is successfully found, the TTL value has been reset.
- If the object is not found, then if the client still requires the Person to be registered, it must recreate the entry as specified in N-Client Registration with ILS (section 4.1).

4.2.3 Stay Alive Refresh Unbind Operation

On completion of the two prior LDAP operations, an LDAP Unbind is the final operation.

4.2.4 Stay Alive LDAP Sequence Diagram

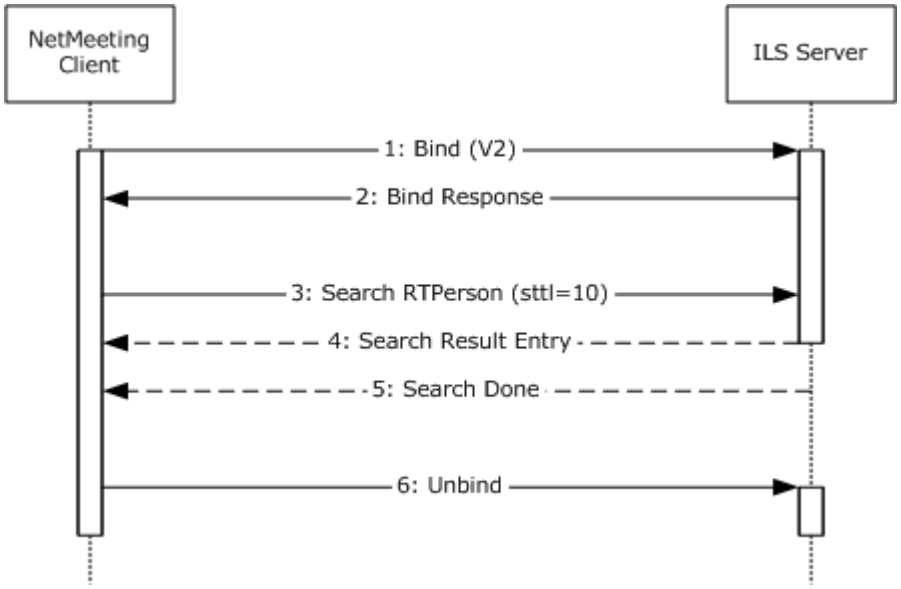

Figure 3: LDAP Stay Alive sequence

4.3 N-Client - Find Online User

When one online user wants to collaborate with another online user, they can check the ILS directory to discover other users. The directory interface in Microsoft® NetMeeting performs an LDAP search on the ILS directory.

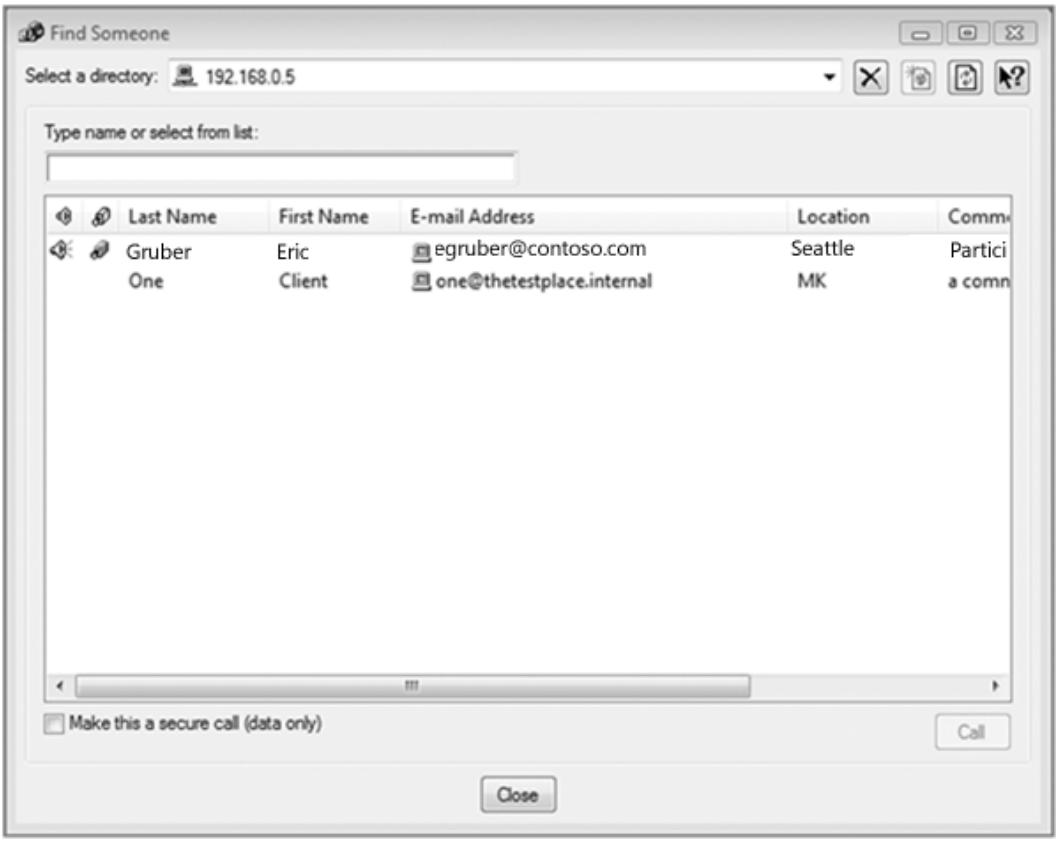

Figure 4: Find Someone dialog box

The process of finding an online user is a three-step LDAP process. The three LDAP operations are as follows:

1. LDAP Bind
2. LDAP Search
3. LDAP Unbind

4.3.1 LDAP Find Online User Bind Operation

The registration is initiated by making an LDAP Bind Request to an ILS Server. If LDAP v2 is offered, no authentication happens. When no authentication is to be performed, then the simple authentication option must be chosen, and the password must be of zero length. When a successful Bind has been made, an LDAP Bind Response, with a `resultCode == 0`, is returned.

4.3.2 LDAP Find Online User LDAP Search Operation

The search for online users is achieved by performing an LDAP search. The search is performed with objectClass=rtPerson, SearchScope: base Object. Three additional attributes are used in the search:

- cn=%. This is an ILS LDAP variation indicating a wild card search on the cn.
- Return entries with sappid=ms-netmeeting. (Note that sappid maps to applicationID.)
- Return entries with a sprotid=h323. (Note that sprotID maps to protocolID.)

4.3.3 LDAP Find Online User Unbind Operation

On completion of the two prior steps, an LDAP Unbind is the final operation.

4.3.4 LDAP Find Online User LDAP Sequence Diagram

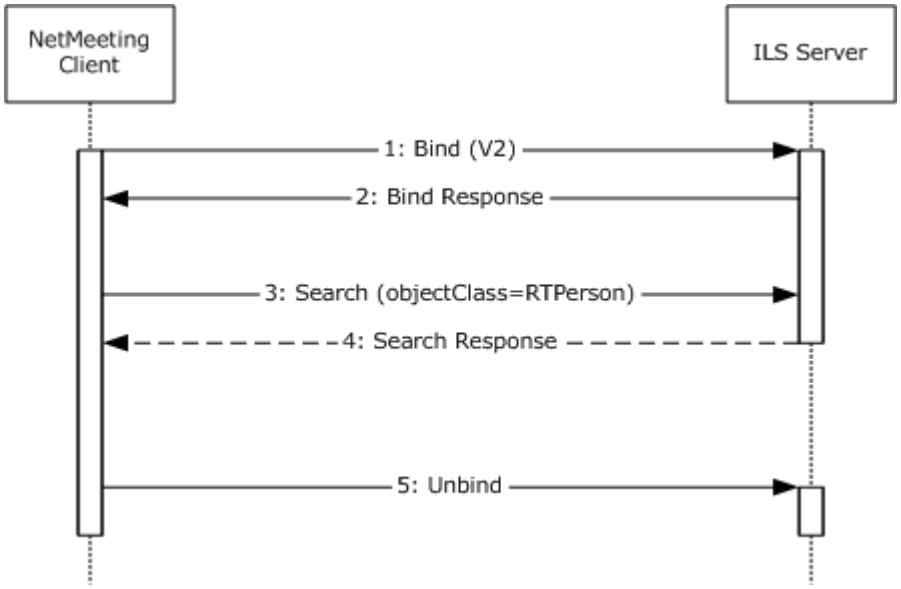

Figure 5: LDAP Find Online User sequence

4.4 N-Client - Unregister

4.4.1 Unregister LDAP Bind Operation

UnRegister is initiated by making an LDAP Bind Request to an ILS Server. If LDAP v2 is offered, no authentication happens. When no authentication is to be performed, the simple authentication option must be chosen, and the password must be of zero length.

The default port for an ILS server is TCP port 1002.

When a successful Bind has been made, an LDAP Bind Response, with a resultCode == 0 is returned.

4.4.2 Unregister LDAP Delete Operation

A successful Bind must be followed by an LDAP Delete operation. The Delete operation allows a client to request the removal of an entry from the directory. The delete operation is as follows:

DelRequest: c=-, o=Microsoft, mailto:cn=egruber@contoso.com%20, objectClass=rtPerson

This is a modified LDAP entry as described in ILS Variations from the LDAP V3 Protocol (section 2.2.6).It has the effect of removing the named rtPerson dynamicObjects from the directory.

4.4.3 Unregister – LDAP Unbind Operation

On completion of the two prior steps, an LDAP Unbind is the final operation.

4.4.4 Unregister LDAP Sequence Diagram

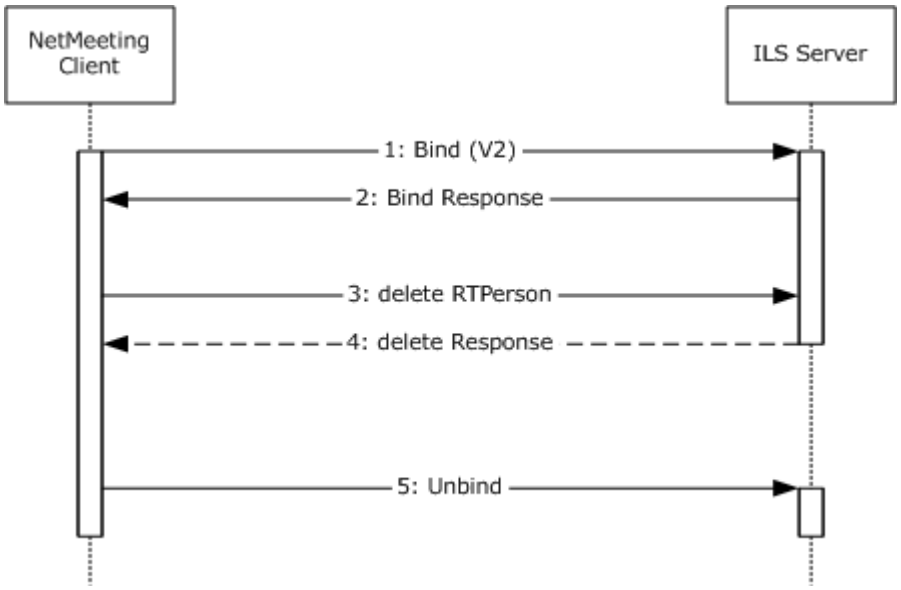

Figure 6: N-Client LDAP Unregister sequence

4.5 TAPI Client – Connect to ILS Server

When Dialer (the TAPI client) is started, it attempts to find the ILS Server as documented in section 3.5.8.

Alternatively, it is also possible for the user to enter a directory name; this can be a NetBIOS name, a DNS name, or an IPv4 address. After this information has been entered (it is not validated), the user can log on to the specified directory server.

The process of logging on or Directory Registration is a six-step LDAP process. The first three steps obtain a server response; there is no response to the Unbind operation. The six LDAP operations are as follows:

1. LDAP Bind

2. LDAP Add rtApplicationUser

3. LDAP Modify rtApplicationUser
4. LDAP Add rtPerson
5. LDAP Modify rtPerson
6. LDAP Unbind

4.5.1 LDAP Bind Operation

The registration is initiated by making a LDAP Bind Request to an ILS server.

The default port for an ILS server is TCP port 1002.

When a successful bind has been made, an LDAP Bind Response, with a resultCode == 0, is returned.

4.5.2 LDAP Add rtApplicationUser Operation

An rtApplicationUser (section 2.2.2) class is created using a LDAP Add operation [RFC2251]. The purpose of the Add operation is to create a dynamic entry of the application user in the directory. Using the example data shown above, the LDAP entry named in the entry field of the Add Request is as follows:

```
cn= egruber]w2kils.testdomain.int,appName=MS-NetMeeting,ou=applications,o=Intranet
```

The Add operation includes data for 12 attributes of the class rtApplicationUser. These are initialized with the values shown. Details of all the attributes of the class rtApplicationUser are given in section.

```
ObjectClass=( rtApplicationUser ) ( DynamicObject )
UserObject=( cn=egruber]w2kils.testdomain.int )
applicationId=( ms-netmeeting )
mimetype=( text/iuls )
GUID=( 008aff194794cf118796444553540000 )
protocolID=( H323 )
protocolMimeType=( text/h323 )
port=( 1720 )
ILSA39321630=( 4 )
ILSA26214430=( 0 )
ILSA32964638=( 1 )
ILSA32833566=( 1 )
```

4.5.3 LDAP Modify rtApplicationUser Operation

After a successful Add Operation has been performed, (LDAP Add Response, resultCode == 0), an LDAP Modify operation must be performed. The purpose of this modify is to update the TTL of this object.

```
ModifyRequest: Object: cn= egruber]w2kils.testdomain.int,appName=MS-
NetMeeting,ou=applications,o=Intranet
Operation: replace
```

The following attributes of the application user (rtApplicationUser) are then modified as follows:

- EntryTTL: the time-to-live is set to 1800

The entries made when connecting to ILS are dynamic directory objects as defined in [RFC2589]. If the entry is successfully updated, an LDAP response of status SUCCESS is returned.

4.5.4 LDAP Add rtPerson Operation

LDAP Modify rtApplicationUser Operation is followed by a request to create an rtPerson class using a LDAP Add operation [RFC2251]. The purpose of the Add operation is to create a dynamic entry of the user in the directory. Using the example data shown above, the LDAP entry named in the entry field of the Add Request is as follows:

```
cn=egruber]w2kils.testdomain.int,ou=dynamic,o=Intranet
```

The Add operation includes data for 19 attributes of the class rtPerson. These are initialized with the values shown. Details of all the attributes of the class rtPerson are given in section 2.2.3.

```
ObjectClass=( rtPerson )( DynamicObject )
ipAddress=( 3808471232 )
rfc822mailbox=( egruber )
givenName=( eric )
surname=( gruber )
location=( N/A )
sflags=( 1 )
c=( US )
comment=( Generated by TAPI3 )
ssecurity=( 1508109 )
smodop=( 0 )
mimetype=( text/iuls )
GUID=( 008aff194794cf118796444553540000 )
ProtocolMimeType=( H323 )
port=( 1720 )
ILSA39321630=( 4 )
ILSA26214430=( 0 )
ILSA32964638=( 1 )
ILSA32833566=( 1 )
```

4.5.5 LDAP Modify rtPerson Operation

After a successful Add Operation has been performed (LDAP Add Response, resultCode == 0), an LDAP modify operation must be performed. The purpose of this modify operation is to update the TTL of this object.

ModifyRequest: Object: cn=egruber]w2kils.testdomain.int,ou=dynamic,o=IntranetOperation: replace

The following attributes of the user (rtPerson) are then modified as follows:

- EntryTTL: the time-to-live is set to 1800.
- The entries made when connecting to ILS are dynamic directory objects as defined in [RFC2589].
- If the entry is successfully updated, an LDAP response of status SUCCESS is returned.

4.5.6 LDAP Unbind Operation

On completion of the above steps, an LDAP Unbind operation is performed. There is no response from the ILS Server for this request.

4.5.7 ILS Registration Sequence Diagram

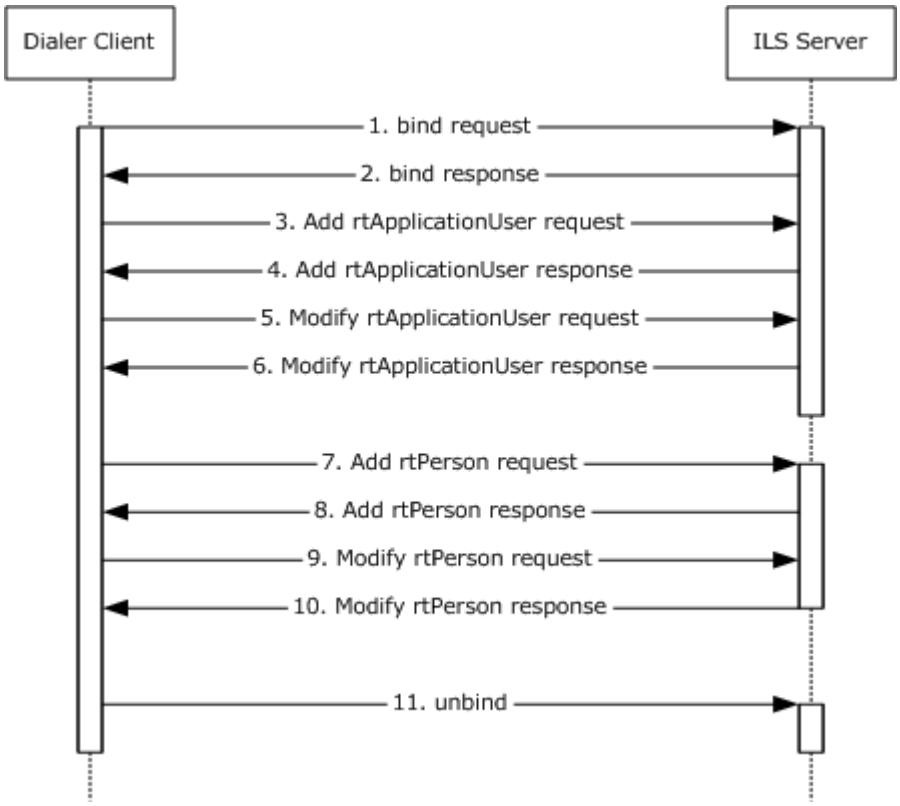

Figure 7: ILS registration sequence

4.6 TAPI Client – Stay Alive Refresh

Dialer ensures that the rtApplicationUser and the rtPerson classes are kept alive on the ILS Server by sending periodic modify requests to reset the EntryTTL for these objects.

4.6.1 TAPI Client – Stay Alive Refresh rtApplicationUser

The rtApplicationUser class is refreshed periodically by sending the modify request.

- ModifyRequest: Object: cn= egruber]w2kils.testdomain.int,appName=MS-NetMeeting,ou=applications,o=Intranet
- Operation: replace

The following attributes of the application user (rtApplicationUser) are then modified as follows:

- EntryTTL: the time-to-live is set to 1800.

- The entries made when connecting to ILS are dynamic directory objects as defined in [RFC2589].
- If the entry is successfully updated an LDAP response of status SUCCESS is returned.

4.6.2 TAPI Client – Stay Alive Refresh rtPerson

The rtPerson class is refreshed periodically by sending the modify request.

ModifyRequest: Object: cn=egruber]w2kils.testdomain.int,ou=dynamic,o=IntranetOperation: replace

The following attributes of the user (rtPerson) are then modified as follows:

- EntryTTL: the time to live is set to 180000
- The entries made when connecting to ILS are dynamic directory objects as defined in [RFC2589]
- If the entry is successfully updated, an LDAP response of status SUCCESS is returned.

4.6.3 ILS Stay Alive Sequence Diagram

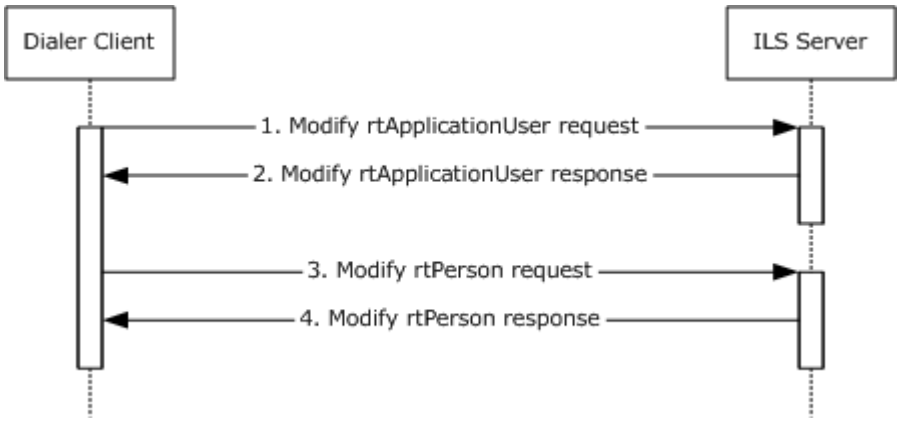

Figure 8: ILS Stay Alive sequence

4.7 TAPI Client - Create Conference

A Dialer creates a new conference using the following steps:

1. LDAP Bind Operation.
2. Verify access rights.
3. LDAP Create conference.
4. LDAP Modify TTL for conference.
5. LDAP Unbind Operation.

4.7.1 LDAP Bind Operation

The registration is initiated by making an LDAP Bind Request to an ILS Server.

The default port for an ILS Server is TCP port 1002.

When a successful bind has been made, an LDAP Bind Response, with a resultCode == 0, is returned.

4.7.2 LDAP Verify Access Rights

Dialer verifies that the user has appropriate rights to create a conference by performing the following steps:

- Create a temporary conference with the same security descriptor.
- Modify the **entryTTL** for the temporary conference.
- Delete the temporary conference.

The conference object is created only if the above operations are all successful. The temporary conference is created using an **LDAP** request as follows:

- uid=19168,ou=dynamic,o=Intranet where the uid is a randomly generated number.

The temporary conference is created with the following attributes:

```
ObjectClass=( RTConference )( DynamicObject )
ntSecurityDescriptor=( )
```

The ntSecurityDescriptor is initialized with the following rights:

- The **SID** for Everyone (S-1-1-0) has read permissions on the object.
- The SID for the user has all permissions on the object.

On successful creation, the temporary conference is modified as follows:

- **ModifyRequest**: Object: uid=19168,ou=dynamic,o=Intranet

The following attributes of the conference (rtConference) are then modified as follows:

- **entryTTL**: the time-to-live is updated (e.g. 300).

On successful update of the **entryTTL**, the temporary conference is deleted using the following LDAP request:

- **DelRequest**: uid=19168,ou=dynamic,o=Intranet

4.7.3 LDAP Create Conference

The conference is created only on successful verification of access rights.

The purpose of the Add operation is to create a dynamic entry for the conference in the directory. Using the example data shown above, the LDAP entry named in the entry field of the Add Request is as follows:

```
Uid=Demo Conference,ou=dynamic,o=Intranet
```

The Add operation includes data for the following attributes of the class rtConference. These are initialized with the values shown. Details of all the attributes of the class rtConference are given in 2.2.4.

```
ObjectClass=( RTConference )( DynamicObject )
protocolId=( IP Conference )
conferenceBlob=( )
ntSecurityDescriptor=( )
As an example, the conferenceBlob can contain the below value
"v=0o=administrator 3456462469 1 IN IP4 w2kils.testdomain.int
s=Demo Conference
i=A description goes here
c=IN IP4 224.0.0.0/15
t=3456462448 3456464248
m=audio 22716 RTP/AVP 0
c=IN IP4 224.2.152.239/15/1
m=video 51232 RTP/AVP 34
c=IN IP4 224.2.188"
```

The ntSecurityDescriptor is initialized with the following rights:

- The SID for Everyone (S-1-1-0) has read permissions on the object.
- The SID for the user has all permissions on the object.

4.7.4 LDAP Modify TTL for Conference

After a successful Add Operation has been performed (LDAP Add Response, resultCode == 0), an LDAP modify operation must be performed. The purpose of this modify operation is to update the **TTL** of this object.

ModifyRequest: Object: uid=Demo Conference,ou=dynamic,o=Intranet

The following attributes of the conference (**rtConference**) are then modified as follows:

- EntryTTL: the time-to-live is updated.
- The entries made when connecting to **ILS** are dynamic directory objects, as defined in [RFC2589].
- If the entry is successfully updated, an LDAP response of status SUCCESS is returned.

4.7.5 LDAP Unbind Operation

On completion of the previous steps, an LDAP Unbind operation is performed. There is no response from the ILS Server for this request.

4.8 TAPI Client – Find Conferences

A Dialer user who wants to join an online conference can check the ILS Directory to discover other users. The Directory interface in Dialer performs an LDAP search on the ILS Directory.

The process of finding an online conference is a three-step LDAP process, as follows:

1. LDAP Bind Operation
2. LDAP Search Operation
3. LDAP Unbind Operation

4.8.1 LDAP Bind Operation

The registration is initiated by making an LDAP Bind Request to an ILS Server. The default port for an ILS Server is TCP port 1002. When a successful bind has been made an LDAP Bind Response, with a resultCode == 0, is returned.

4.8.2 LDAP Search Operation

Dialer searches for a list of conferences on the ILS Server using the following criteria:

```
search base: "ou=dynamic, o=Intranet"
search filter: "(uid=*)"
search scope: "one level"
```

The attributes requested are:

- "uid"
- "protocolId"
- "conferenceBlob"
- "ntSecurityDescriptor"

4.8.3 LDAP Unbind Operation

On completion of the previous steps, an LDAP Unbind operation is performed. There is no response from the ILS Server for this request.

4.8.4 ILS Find Conferences Sequence Diagram

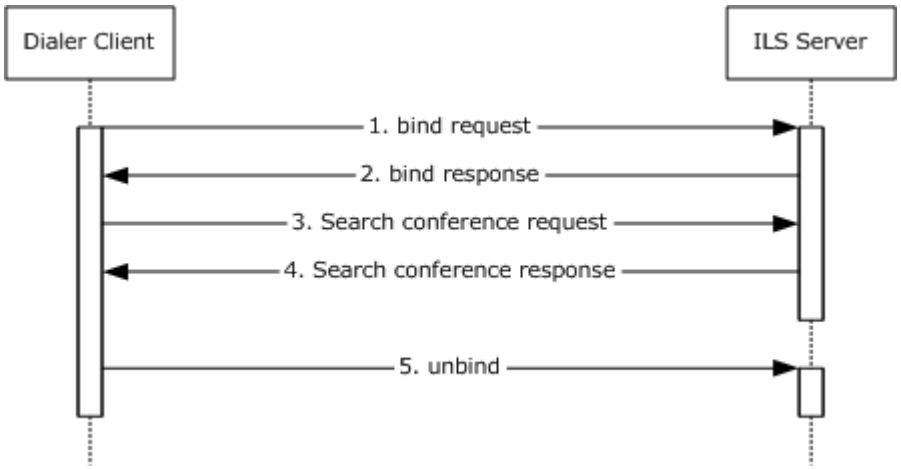

Figure 9: ILS Find Conferences sequence

4.9 TAPI Client – Find People

To collaborate with another online user, a Dialer user can check the ILS. Directory to discover other users. The Directory interface in Dialer performs an LDAP search on the ILS Directory. The process of finding an online user is a three-step LDAP process as follows:

1. LDAP Bind Operation
2. LDAP Search Operation
3. LDAP Unbind Operation

4.9.1 LDAP Bind Operation

The registration is initiated by making a LDAP Bind Request to an ILS Server. The default port for an ILS Server is TCP port 1002. When a successful Bind has been made, an LDAP Bind response with a resultCode == 0 is returned.

4.9.2 LDAP Search Operation

Dialer searches for a list of users on the ILS Server using the following search criteria.

```
search base: "ou=dynamic,o=Intranet"
search filter: "(cn=*)"
search scope: "one level"
```

The attributes requested are:

- "cn"
- "ipAddress"
- "ntSecurityDescriptor"

4.9.3 LDAP Unbind Operation

On completion of the above steps, an LDAP Unbind operation is performed. There is no response from the ILS Server for this request.

4.9.4 ILS Find Users Sequence Diagram

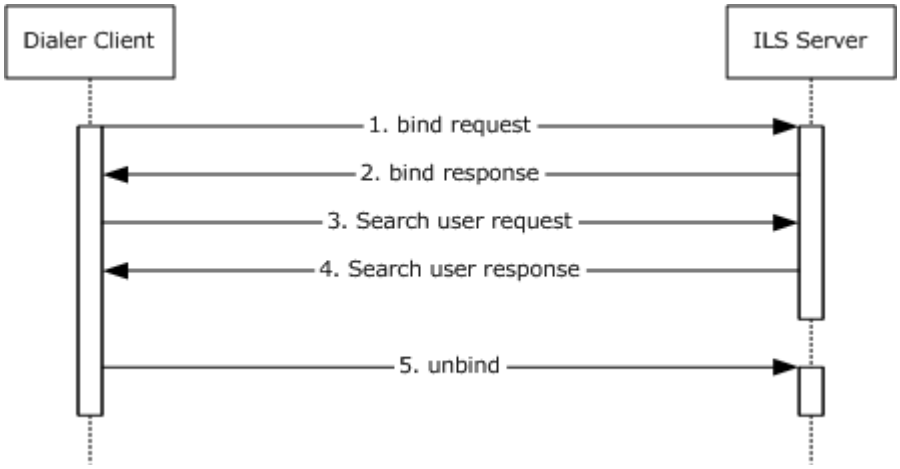

Figure 10: ILS Find Users sequence

4.10 TAPI Client – Disconnect from ILS Server

Dialer (the TAPI client) does not take any specific action to disconnect from the ILS Server. The rtApplicationUser (Object Class) and rtPerson objects created are deleted by the ILS Server when the EntryTTL expires.

4.11 Sample LDAP Search Filters for ILS

4.11.1 LDAP Search Filters Used by the TAPI Client

The following search filters are used by the TAPI client when searching for a conference or person (as specified in [RFC2254]). All uppercase strings are user-provided values.<32>

To search for a conference matching NAME:

```
(uid=NAME*)
```

To search for a person matching NAME:

```
(cn=NAME*)
```

4.11.2 LDAP Search Filters Used by the N-Client

To search for a person by name:

```
(&(objectClass=rtPerson)(cn=NAME))
```

To search for a person by name and set the TTL to 10 minutes:

```
(&(objectClass=rtPerson)(cn=NAME)(sttl=10))
```

To search for a person by name and sappid:

```
(&(objectClass=rtPerson)(cn=NAME)(sappid=SAPPID))
```

To search for a person by name, sappid, and sprotid:

```
(&(objectClass=rtPerson)(cn=NAME)(sappid=SAPPID)(sprotid=SPROTID))
```

To search for a person by name and sprotid:

```
(&(objectClass=rtPerson)(cn=NAME)(sprotid=SPROTID))
```

To search for a conference by name:

```
(&(objectClass=rtConference)(cn=NAME))
```

To search for a conference by name and set the "time to live" to 10 minutes:

```
(&(objectClass=rtConference)(cn=NAME)(sttl=10))
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

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

- Microsoft Windows NT® operating system
- Microsoft Windows® 2000 operating system
- Windows® XP 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:

The TAPI ILS variant of the LDAP protocol supported by the ILS server is available only on particular versions of Windows:

- ILS is supported only on Windows 2000 Server. On Windows Server 2003, dynamic object features were added to Active Directory to support TAPI clients on Windows XP.
- TAPI 3.0, which is included in Windows 2000, requires a Windows 2000 ILS server or Site Server 3.0.
- TAPI 3.1, which is included in Windows XP, can communicate with a Windows 2000 ILS server or a Windows Server 2003 application partition.

<2> Section 1.4: In the absence of an ILS server, TAPI clients can interact directly with Windows Server 2003 using standard LDAP V3 protocol to manipulate objects that are instances of the classes msTAPI-rtPerson and msTAPI-rtConference.

<3> Section 2.2.2: The TAPI client uses the value "ms-netmeeting".

<4> Section 2.2.2: By default, TAPI clients populate this attribute with a value of "0"; they do not use the value.

<5> Section 2.2.2: NetMeeting Clients populate this attribute with the status of being in a call.

<6> Section 2.2.2: By default, TAPI clients populate this attribute with a value of "1"; they do not use the value.

<7> Section 2.2.2: By default, TAPI clients populate this attribute with a value of "1"; they do not use the value.

<8> Section 2.2.2: By default, TAPI clients populate this attribute with the value of "4"; they do not use the value.

<9> Section 2.2.2: This attribute is present for NetMeeting compatibility.

<10> Section 2.2.2: This attribute is required by the schema, although it is only used for NetMeeting 2.0 compatibility. Originally, the attribute was used by NetMeeting 2.0 as a filter when searching a directory for a user. It is not required for interoperability with NetMeeting 3.0 or later.

<11> Section 2.2.3: It is no longer used and is retained to be compatible with Site Server 3.0. NetMeeting clients populate this attribute with "-" (a dash).

By default, TAPI clients populate this attribute with "US" (United States); they do not use the value.

<12> Section 2.2.3: NetMeeting clients populate this attribute with the e-mail address that the user supplied during setup or later. TAPI clients populate this attribute with the common (display) name of the user.

<13> Section 2.2.3: NetMeeting clients populate this attribute with the name that the user supplied during setup or later.

By default, TAPI clients populate this attribute with "Generated by TAPI3"; they do not use the value.

<14> Section 2.2.3: By default, TAPI clients populate this attribute with a value of "0"; they do not use the value.

<15> Section 2.2.3: By default, TAPI clients populate this attribute with a value of "1"; they do not use the value.

<16> Section 2.2.3: By default, TAPI clients populate this attribute with a value of "1"; they do not use the value.

<17> Section 2.2.3: By default, TAPI clients populate this attribute with the value of "4"; they do not use the value.

<18> Section 2.2.3: NetMeeting populates this attribute with the value "1503" (Data port) and "1720" (Audio/Video port).

<19> Section 2.2.3: NetMeeting clients populate this attribute with the e-mail address that the user supplied during setup or later.

<20> Section 2.2.3: NetMeeting clients populate this attribute with the operation commands in ModifyRequest. By default, TAPI clients populate this attribute with "0" (ADD); they do not use the value.

<21> Section 2.2.3: NetMeeting clients populate this attribute with the name that the user supplied during setup or later.

TAPI clients populate this attribute with a single white space " " (no surname); they do not use the value.

<22> Section 2.2.3: NetMeeting clients populate this attribute with a string form of a **DWORD**, which is based on a semi-unique ID obtained from a clock tick. It is used to identify a user in the case of NetMeeting problems and a reconnect with the server is required.

By default, TAPI clients populate this attribute with "1508109"; they do not use the value.

<23> Section 2.3: The following tables list the NetMeeting and Dialer attributes, and indicate whether the attribute is used or not. Use of the attribute by NetMeeting or Dialer does not imply visible behavior on the wire.

rtApplicationUser – The User of an Application

Attribute	Used by NetMeeting	Used by Dialer
applicationID	Yes	Yes
appName	Yes	Yes
Application Name	Yes	No
groupObject	No	No
guid	Yes	Yes
ILSA26214430	Yes	Yes
ILSA26279966	Yes	No
ILSA32833566	Yes	Yes
ILSA32964638	Yes	Yes
ILSA39321630	Yes	Yes
mimeType	Yes	Yes
objectClass	Yes	Yes
port	Yes	Yes
protocolGUID	Yes	No
protocolID	Yes	Yes
protocolMimeType	Yes	Yes
sFlags	Yes	No
userObject	Yes	Yes

rtPerson – An Online Person

Attribute	Used by NetMeeting	Used by Dialer
c	Yes	Yes
cn	Yes	Yes
comment	Yes	Yes
givenName	Yes	Yes
guid	Yes	Yes
ILSA26214430	Yes	Yes
ILSA26279966	Yes	No
ILSA32833566	Yes	Yes
ILSA32964638	Yes	Yes

Attribute	Used by NetMeeting	Used by Dialer
ILSA39321630	Yes	Yes
ipAddress	Yes	Yes
location	Yes	Yes
mimeType	Yes	Yes
o	Yes	Yes
port	Yes	Yes
protocolMimeType	Yes	Yes
rfc822mailbox	Yes	Yes
sFlags	Yes	Yes
smodop	Yes	Yes
surName	Yes	Yes
securityToken	Yes	Yes

rtConference – An Online Conference

Attribute	Used by NetMeeting	Used by Dialer
advertisingScope	No	No
announcementScope	No	No
applicationID	No	No
Attendees	No	No
attendeesCount	No	No
confAddr	No	No
conferenceBlob	No	Yes
contactInformation	No	No
generalDescription	No	No
isEncrypted	No	No
maxAttendeesCount	No	No
Originator	No	No
protocolID	No	Yes
rating	No	No
startTime	No	No

Attribute	Used by NetMeeting	Used by Dialer
stopTime	No	No
subType	No	No
url	No	No
uid	No	Yes

<24> Section 2.3.4: The unique GUID used is 6D12FC99-B836-11D0-9601-00C04FC30E1A3.

<25> Section 2.3.5: The unique GUID used is ed1af3b3-bcfa-11d0-853a-00a0c90c93e1.

<26> Section 3.5.3: NetMeeting 3.01 only performs a simple bind using LDAP v2 and therefore never performs an authenticated bind.

<27> Section 3.5.4: The TAPI Client Phone Dialer 1.00 included with Windows 2000 Server and Windows 2000 Workstation performs authentication using Sicily Authentication. Only Active Directory security principles can authenticate in this manner.

<28> Section 3.5.11: NetMeeting 3.01 specifies "base" scope in search requests, when an LDAP server would normally use "sub".

<29> Section 3.6: An ILS Server will remove objects from the database when the entryTTL attribute reaches zero. This removal may not be immediate and will depend on various indeterminate factors such as processor speed, processor load, available memory, etc.

<30> Section 4.1.1: NetMeeting clients work in this manner.

<31> Section 4.2.1: NetMeeting clients work in this manner.

<32> Section 4.11.1: NetMeeting 3.01 specifies "base" scope in search requests, when an LDAP server would normally use "sub".

7 Change Tracking

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

8 Index

A

Abstract data model 19
Applicability 10

C

Capability negotiation 11
Change tracking 49
Connecting to ILS Server example (TAPI client) 32

D

Data model - abstract 19
Disconnecting from ILS Server example (TAPI client) 41

E

Examples
- connecting to ILS Server (TAPI client) 32
- disconnecting from ILS Server (TAPI client) 41
- finding conferences example (TAPI client) 38
- finding online user (NetMeeting) 30
- finding people (TAPI client) 40
- NetMeeting registration with ILS 26
- sample LDAP search filters for ILS 41
- stay alive refresh (NetMeeting) 28
- stay alive refresh (TAPI client) 35
- unregistering (NetMeeting) 31

F

Fields - vendor-extensible 11
Finding conferences example (TAPI client) 38
Finding online user example (NetMeeting) 30
Finding people example (TAPI client) 40

G

Glossary 7

H

Higher-layer triggered events 19

I

ILS schema objects 16
Implementer - security considerations 43
Index of security parameters 43
Informative references 9
Initialization 19
Introduction 7

L

LDAP - sample search filters for ILS 41
Local events 25

M

Message processing 20
Messages
- ILS schema objects 16
- syntax 12
- transport 12

N

NetMeeting registration with ILS example 26
Normative references 8

O

Objects - ILS schema 16
Overview (synopsis) 9

P

Parameters - security index 43
Preconditions 10
Prerequisites 10
Product behavior 44

R

References
- informative 9
- normative 8

Relationship to other protocols 10

S

Sample LDAP search filters for ILS 41
Security
- implementer considerations 43
- parameter index 43

Sequencing rules 20
Standards assignments 11
Stay alive refresh example (NetMeeting) 28
Stay alive refresh example (TAPI client) 35
Syntax 12

T

TAPI ILS protocol [TAPI ILS] 9
Telephony API Internet Locator Data Structure - TAPI ILS 9
Timer events 24
Timers 19
Tracking changes 49
Transport 12
Triggered events - higher-layer 19

U

Unregistering example (NetMeeting) 31

V

Vendor-extensible fields 11
Versioning 11