WSA - HW - 4:

2. Find and document information for the LDAP protocol. Provide information for some applications that are integrated with the LDAP protocol.

What is LDAP?

The Lightweight Directory Access Protocol (LDAP) is a directory service protocol that runs directly over the TCP/IP stack. The information model (both for data and namespaces) of LDAP is similar to that of the X.500 OSI directory service, but with fewer features and lower resource requirements than X.500. Unlike most other Internet protocols, LDAP has an associated API that simplifies writing Internet directory service applications. The LDAP API is applicable to directory management and browser applications that do not have directory service support as their primary function. LDAP cannot create directories or specify how a directory service operates.

Lightweight Directory Access Protocol

Purpose

The Lightweight Directory Access Protocol (LDAP) is a directory service protocol that runs on a layer above the TCP/IP stack. It provides a mechanism used to connect to, search, and modify Internet directories.

The LDAP directory service is based on a client-server model. The function of LDAP is to enable access to an existing directory.

The data model (data and namespace) of LDAP is similar to that of the X.500 OSI directory service, but with lower resource requirements. The associated LDAP API simplifies writing Internet directory service applications.

Where applicable

The LDAP API is applicable to directory management and browser applications that do not have directory service support as their primary function. Conversely, LDAP is neither applicable to creating directories, nor specifying how a directory service operates.

Developer audience

The LDAP API documentation in the Platform Software Development Kit (SDK) is intended for experienced C and C++ programmers and Internet directory developers.

LDAP supports the C and C++ programming languages.

A familiarity with directory services and the LDAP Client/Server Model are necessary for the development with the LDAP API.

Run-time requirements

Client applications that use the LDAP API, run on Windows Vista, Windows XP, and Windows 2000. All platforms must have TCP/IP installed.

Active Directory servers that support client applications using the LDAP API include Windows Server 2008, Windows Server 2003, and Windows 2000 Server.

History

Telecommunication companies' understanding of directory requirements was well developed after some 70 years of producing and managing telephone directories. These companies introduced the concept of directory services to information technology and computer networking, their input culminating in the comprehensive X.500 specification, a suite of protocols produced by the International Telecommunication Union (ITU) in the 1980s.

X.500 directory services were traditionally accessed via the X.500 Directory Access Protocol (DAP), which required the Open Systems Interconnection (OSI) protocol stack. LDAP was originally intended to be a lightweight alternative protocol for accessing X.500 directory services through the simpler (and now widespread) TCP/IP protocol stack. This model of directory access was borrowed from the DIXIE and Directory Assistance Service protocols.

Standalone LDAP directory servers soon followed, as did directory servers supporting both DAP and LDAP. The latter has become popular in enterprises, as LDAP removed any need to deploy an OSI network. Today, X.500 directory protocols including DAP can also be used directly over TCP/IP.

The protocol was originally created by Tim Howes of the University of Michigan, Steve Kille of Isode Limited, Colin Robbins of Nexor and Wengyik Yeong of Performance Systems International, circa 1993. Mark Wahl of Critical Angle Inc., Tim Howes, and Steve Kille started work in 1996 on a new version of LDAP, LDAPv3, under the aegis of the Internet Engineering Task Force (IETF). LDAPv3, first published in 1997, superseded LDAPv2 and added support for extensibility, integrated the Simple Authentication and Security Layer, and better aligned the protocol to the 1993 edition of X.500. Further development of the LDAPv3 specifications themselves and of numerous extensions adding features to LDAPv3 has come through the IETF.

In the early engineering stages of LDAP, it was known as Lightweight Directory Browsing Protocol, or LDBP. It was renamed with the expansion of the scope of the protocol beyond directory browsing and searching, to include directory update functions. It was given its Lightweight name because it was not as network intensive as its DAP predecessor and thus was more easily implemented over the internet due to its relatively modest bandwidth usage.

LDAP has influenced subsequent Internet protocols, including later versions of X.500, XML Enabled Directory (XED), Directory Service Markup Language (DSML), Service Provisioning Markup Language (SPML), and the Service Location Protocol (SLP).

Protocol overview

A client starts an LDAP session by connecting to an LDAP server, called a Directory System Agent (DSA), by default on TCP port and UDP port 389. The client then sends an operation request to the server, and the server sends responses in return. With some exceptions, the client does not need to wait for a response before sending the next request, and the server may send the responses in any order. All information is transmitted using Basic Encoding Rules (BER).

The client may request the following operations:

StartTLS — use the LDAPv3 Transport Layer Security (TLS) extension for a secure connection

Bind — authenticate and specify LDAP protocol version

Search — search for and/or retrieve directory entries

Compare — test if a named entry contains a given attribute value

Add a new entry

Delete an entry

Modify an entry

Modify Distinguished Name (DN) — move or rename an entry

Abandon — abort a previous request

Extended Operation — generic operation used to define other operations

Unbind — close the connection (not the inverse of Bind)

In addition the server may send "Unsolicited Notifications" that are not responses to any request, e.g. before the connection is timed out.
A common alternative method of securing LDAP communication is using an SSL tunnel. This is denoted in LDAP URLs by using the URL scheme "ldaps". The default port for LDAP over SSL is 636. The use of LDAP over SSL was common in LDAP Version 2 (LDAPv2) but it was never standardized in any formal specification. This usage has been deprecated along with LDAPv2, which was officially retired in 2003.

Directory structure

The protocol provides an interface with directories that follow the 1993 edition of the X.500 model:
An entry consists of a set of attributes.
An attribute has a name (an attribute type or attribute description) and one or more values. The attributes are defined in a schema (see below).
Each entry has a unique identifier: its Distinguished Name (DN). This consists of its Relative Distinguished Name (RDN), constructed from some attribute(s) in the entry, followed by the parent entry's DN. Think of the DN as the full file path and the RDN as its relative filename in its parent folder (e.g. if /foo/bar/myfile.txt were the DN, then myfile.txt would be the RDN).
A DN may change over the lifetime of the entry, for instance, when entries are moved within a tree. To reliably and unambiguously identify entries, a UUID might be provided in the set of the entry's operational attributes.
An entry can look like this when represented in LDAP Data Interchange Format (LDIF) (LDAP itself is a binary protocol):

```
dn: cn=John Doe,dc=example,dc=com

cn: John Doe

givenName: John

sn: Doe

telephoneNumber: +1 888 555 6789

telephoneNumber: +1 888 555 1232

mail: john@example.com
```

manager: cn=Barbara Doe,dc=example,dc=com

objectClass: inetOrgPerson

objectClass: organizationalPerson

objectClass: person

objectClass: top

"dn" is the distinguished name of the entry; it is neither an attribute nor a part of the entry. "cn=John Doe" is the entry's RDN (Relative Distinguished Name), and "dc=example,dc=com" is the DN of the parent entry, where "dc" denotes 'Domain Component'. The other lines show the attributes in the entry. Attribute names are typically mnemonic strings, like "cn" for common name, "dc" for domain component, "mail" for e-mail address, and "sn" for surname.

A server holds a subtree starting from a specific entry, e.g. "dc=example,dc=com" and its children. Servers may also hold references to other servers, so an attempt to access "ou=department,dc=example,dc=com" could return a referral or continuation reference to a server that holds that part of the directory tree. The client can then contact the other server. Some servers also support chaining, which means the server contacts the other server and returns the results to the client.

LDAP rarely defines any ordering: The server may return the values of an attribute, the attributes in an entry, and the entries found by a search operation in any order. This follows from the formal definitions - an entry is defined as a set of attributes, and an attribute is a set of values, and sets need not be ordered.

Schema

The contents of the entries in a subtree are governed by a directory schema, a set of definitions and constraints concerning the structure of the directory information tree (DIT).
The schema of a Directory Server defines a set of rules that govern the kinds of information that the server can hold. It has a number of elements, including:

- Attribute Syntaxes—Provide information about the kind of information that can be stored in an attribute.
- Matching Rules—Provide information about how to make comparisons against attribute values.
- Matching Rule Uses—Indicate which attribute types may be used in conjunction with a particular matching rule.
- Attribute Types—Define an object identifier (OID) and a set of names that may be used to refer to a given attribute, and associates that attribute with a syntax and set of matching rules.
- Object Classes—Define named collections of attributes and classify them into sets of required and optional attributes.
- Name Forms—Define rules for the set of attributes that should be included in the RDN for an entry.
- Content Rules—Define additional constraints about the object classes and attributes that may be used in conjunction with an entry.
- Structure Rule—Define rules that govern the kinds of subordinate entries that a given entry may have.

Attributes are the elements responsible for storing information in a directory, and the schema defines the rules for which attributes may be used in an entry, the kinds of values that those attributes may have, and how clients may interact with those values.

Clients may learn about the schema elements that the server supports by retrieving an appropriate subschema subentry.

The schema defines object classes. Each entry must have an objectClass attribute, containing named classes defined in the schema. The schema definition of the classes of an entry defines what kind of object the entry may represent - e.g. a person, organization or domain. The object class definitions also define the list of attributes that must contain values and the list of attributes which may contain values.

For example, an entry representing a person might belong to the classes "top" and "person". Membership in the "person" class would require the entry to contain the "sn" and "cn" attributes, and allow the entry also to contain "userPassword", "telephoneNumber", and other attributes. Since entries may have multiple ObjectClasses values, each entry has a complex of optional and mandatory attribute sets formed from the union of the object classes it represents. ObjectClasses can be inherited, and a single entry can have multiple ObjectClasses values that define the available and required attributes of the entry itself. A parallel to the schema of an objectClass is a class definition and an instance in Object-oriented programming, representing LDAP objectClass and LDAP entry, respectively.

Directory servers may publish the directory schema controlling an entry at a base DN given by the entry's subschemaSubentry operational attribute. (An operational attribute describes operation of the directory rather than user information and is only returned from a search when it is explicitly requested.)

Server administrators can add additional schema entries in addition to the provided schema elements. A schema for representing individual people within organizations is termed a white pages schema.

Variations

A lot of the server operation is left to the implementor or administrator to decide. Accordingly, servers may be set up to support a wide variety of scenarios.

For example, data storage in the server is not specified - the server may use flat files, databases, or just be a gateway to some other server. Access control is not standardized, though there has been work on it and there are commonly used models. Users' passwords may be stored in their entries or elsewhere. The server may refuse to perform operations when it wishes, and impose various limits.

Most parts of LDAP are extensible. Examples: One can define new operations. Controls may modify requests and responses, e.g. to request sorted search results. New search scopes and Bind methods can be defined. Attributes can have options that may modify their semantics.

Other data models

As LDAP has gained momentum, vendors have provided it as an access protocol to other services. The implementation then recasts the data to mimic the LDAP/X.500 model, but how closely this model is followed varies. For example, there is software to access SQL databases through LDAP, even though LDAP does not readily lend itself to this. X.500 servers may support LDAP as well.

Similarly, data previously held in other types of data stores are sometimes moved to LDAP directories. For example, Unix user and group information can be stored in LDAP and accessed via PAM and NSS modules. LDAP is often used by other services for authentication.

An example of such data model is the GLUE Schema,which is used in a distributed information system based on LDAP that enable users, applications and services to discover which services exist in a Grid infrastructure and further information about their structure and state.

Usage

An LDAP server may return referrals to other servers for requests that it cannot fulfill itself. This requires a naming structure for LDAP entries so one can find a server holding a given DN or distinguished name, a concept defined in the X.500 Directory and also used in LDAP. Another way of locating LDAP servers for an organization is a DNS server resource record (SRV).

An organization with the domain example.org may use the top level LDAP DN dc=example,dc=org (where dc means domain component). If the LDAP server is also named ldap.example.org, the organization's top level LDAP URL becomes ldap://ldap.example.org/dc=example,dc=org.

Primarily two common styles of naming are used in both X.500 [2008] and LDAPv3. These are documented in the ITU specifications and IETF RFCs. The original form takes the top level object as the country object, such as c=US, c=FR. The domain component model uses the model described above. An example of country based naming could be L=Locality, ou=Some Organizational Unit, o=Some Organization, c=FR, or in the US: CN=Common Name, L=Locality, ou=Some Organizational Unit, o=Some Organization, st=CA, c=US.

LDAP Application Program Interface

The LDAP Application Program Interface, described by RFC 1823, is an Informational RFC that specifies an application programming interface in the C programming language for version 2 of the Lightweight Directory Access Protocol. Version 2 of LDAP is historic. Commonly available LDAP C APIs do not strictly adhere to this specification.

List of LDAP software:

The following is a list of software programs that can communicate with and/or host directory services via the Lightweight Directory Access Protocol (LDAP)

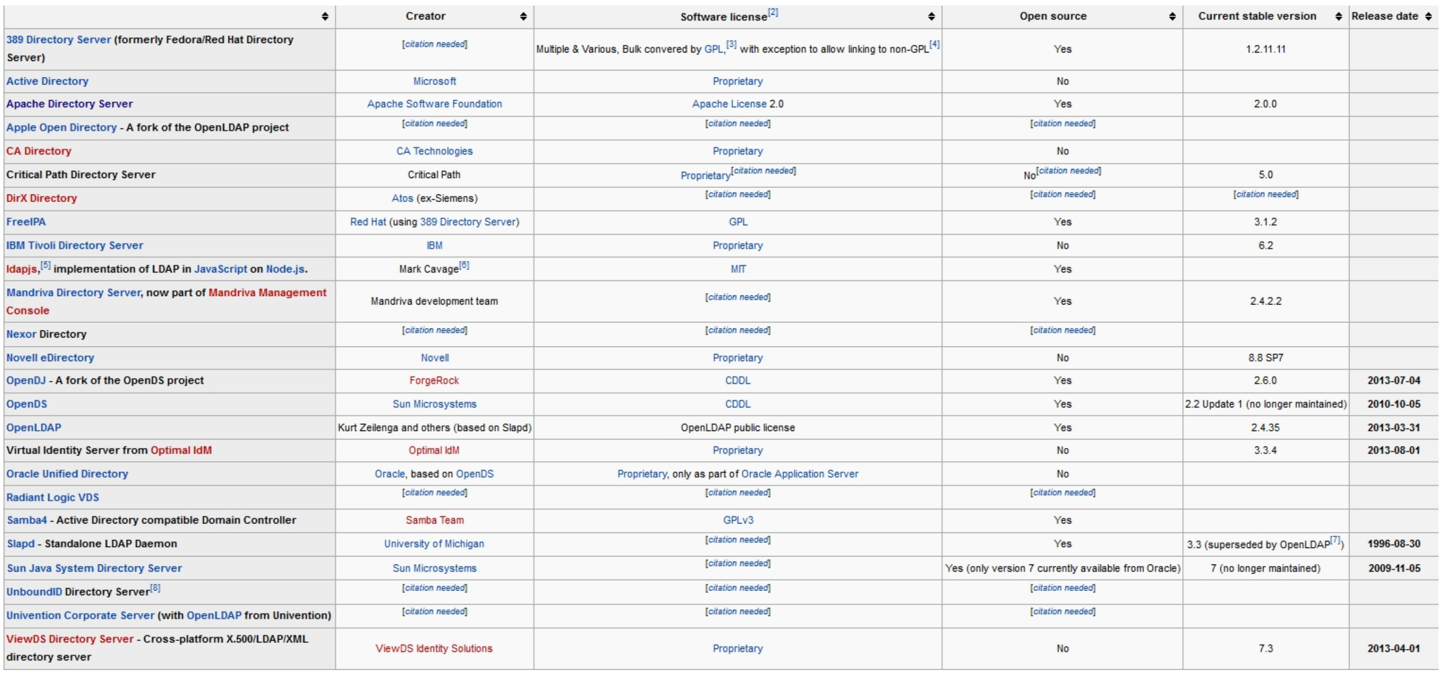
Cross-platform:

- Apache Directory Server/Studio - an LDAP browser and directory client for Linux, Mac OS X, and Microsoft Windows, and as a plug-in for the Eclipse development environment.
- FusionDirectory - a web application under license GNU General Public License developed in PHP for managing LDAP directory and associated services.
- JXplorer - a Java-based browser that runs in any operating environment.
- LDAP Account Manager - a PHP based webfrontend for managing various account types in an LDAP directory.
- phpLDAPadmin - a web-based LDAP administration tool for creating and editing LDAP entries in any LDAP server.

Microsoft Windows:

- Active Directory Explorer - a free LDAP client tool from Microsoft. http://en.wikipedia.org/wiki/Active_Directory_Explorer
- LDAP Admin - a free, open source LDAP directory browser and editor. http://en.wikipedia.org/wiki/Ldap_admin

Server software:



	Creator	Software license[2]	Open source	Current stable version	Release date
389 Directory Server (formerly Fedora/Red Hat Directory Server)	[citation needed]	Multiple & Various, Bulk convered by GPL,[3] with exception to allow linking to non-GPL[4]	Yes	1.2.11.11	
Active Directory	Microsoft	Proprietary	No		
Apache Directory Server	Apache Software Foundation	Apache License 2.0	Yes	2.0.0	
Apple Open Directory - A fork of the OpenLDAP project	[citation needed]	[citation needed]	[citation needed]		
CA Directory	CA Technologies	Proprietary	No		
Critical Path Directory Server	Critical Path	Proprietary[citation needed]	No[citation needed]	5.0	
DirX Directory	Atos (ex-Siemens)	[citation needed]	[citation needed]	[citation needed]	
FreeIPA	Red Hat (using 389 Directory Server)	GPL	Yes	3.1.2	
IBM Tivoli Directory Server	IBM	Proprietary	No	6.2	
ldapjs,[5] implementation of LDAP in JavaScript on Node.js.	Mark Cavage[6]	MIT	Yes		
Mandriva Directory Server, now part of Mandriva Management Console	Mandriva development team	[citation needed]	Yes	2.4.2.2	
Nexor Directory	[citation needed]	[citation needed]	[citation needed]		
Novell eDirectory	Novell	Proprietary	No	8.8 SP7	
OpenDJ - A fork of the OpenDS project	ForgeRock	CDDL	Yes	2.6.0	**2013-07-04**
OpenDS	Sun Microsystems	CDDL	Yes	2.2 Update 1 (no longer maintained)	**2010-10-05**
OpenLDAP	Kurt Zeilenga and others (based on Slapd)	OpenLDAP public license	Yes	2.4.35	**2013-03-31**
Virtual Identity Server from Optimal IdM	Optimal IdM	Proprietary	No	3.3.4	**2013-08-01**
Oracle Unified Directory	Oracle, based on OpenDS	Proprietary, only as part of Oracle Application Server	No		
Radiant Logic VDS	[citation needed]	[citation needed]	[citation needed]		
Samba4 - Active Directory compatible Domain Controller	Samba Team	GPLv3	Yes		
Slapd - Standalone LDAP Daemon	University of Michigan	[citation needed]	Yes	3.3 (superseded by OpenLDAP[7])	**1996-08-30**
Sun Java System Directory Server	Sun Microsystems	[citation needed]	Yes (only version 7 currently available from Oracle)	7 (no longer maintained)	**2009-11-05**
UnboundID Directory Server[8]	[citation needed]	[citation needed]	[citation needed]		
Univention Corporate Server (with OpenLDAP from Univention)	[citation needed]	[citation needed]	[citation needed]		
ViewDS Directory Server - Cross-platform X.500/LDAP/XML directory server	ViewDS Identity Solutions	Proprietary	No	7.3	**2013-04-01**

Info links:

http://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol

http://en.wikipedia.org/wiki/LDAP_Application_Program_Interface

http://tools.ietf.org/html/rfc1823

http://www.ietf.org/rfc/rfc1777.txt

http://en.wikipedia.org/wiki/List_of_LDAP_software

http://en.wikipedia.org/wiki/Ldap_admin

http://msdn.microsoft.com/en-us/library/windows/desktop/aa367008%28v=vs.85%29.aspx

http://msdn.microsoft.com/en-us/library/windows/desktop/aa367036%28v=vs.85%29.aspx

http://weblogic-wonders.com/weblogic/ldap-server/

https://h20392.www2.hp.com/portal/swdepot/displayProductInfo.do?productNumber=J4269AA