LDAP Directory Services

Directory Service

A Directory Service is nothing but a software system that responds to requests for information about entities, e.g. people in an organization.
X.500 and Network Information Service (NIS) are examples of directory services.

Need of Directory Service

Enterprise Computing Environments have a need to store information in a centralized data store so that it can be added to, deleted, modified, and queried by users and applications. The information stored could be user accounts, e-mail addresses, digital certificates, component object names, network names, printers, groups and so on. There is a need to access this information both from within the enterprise and from the Internet. The amount of information stored varies greatly with the customer. This data store has come to be known as a *Directory Service*.

Directory services not only allow you to locate and access these resources, but also let you manage the relationships among them.

For our own use, we all maintain personal address directory where we store addresses, telephone nos. and other information in a format that is most suitable for us. But when we talk about maintaining a global directory service on Internet or in any organization,

The Directory Service must be:

- *Flexible* enough to store a range of information types
- *Secure* when accessing from both the Internet and intranet
- *Scalable* from a small business to the largest enterprise
- *Extensible* as business needs change
- *Accessible* via an open, standards-based protocol

Using an open protocol enables the information in the Directory Service to be accessible from clients from different vendors. Directory Services from different vendors communicating using an open protocol can exchange information with each other to create aggregated directories.

X.500 Directory

X.500 is a standard for a Directory Service by the International Telecommunications Union (ITU). X.500, the OSI directory standard, defines a comprehensive Directory Service, including an information model, namespace, functional model, and authentication framework. X.500 also defines the Directory Access Protocol (DAP) used by clients to access the directory. DAP is a full OSI protocol that contains extensive functionality, much of which is not used by most applications.

DAP is significantly more complicated than the more prevalent TCP/IP stack implementations and requires more code and computing horsepower to run. The size and complexity of DAP makes it difficult to run on thin clients, such as the PC and Macintosh where TCP/IP functionality often comes with the machine. DAP stack implementations are cumbersome to administer, thus limiting the acceptance of X.500. Whereas, LDAP is a subset of X.500 Directory Access Protocol (DAP) and can easily be embedded in lightweight applications (both client and server) such as email, web browsers, and groupware.

Initiation of LDAP

As "LDAP" suggests, it started life as an access method, not a directory. By establishing a protocol, LDAP standardizes how an application can "talk" to a directory. LDAP was originally designed to be a lightweight front end to X.500 directories at the University of Michigan. But later the work turned to develop a standalone LDAP-based directory that could be accessed by any LDAP client and this work resulted in LDAPv1. Further work in this direction resulted in LDAPv2 and then LDAPv3 with more features.

LDAPv1 protocol was designed to provide access to the Directory while not incurring the resource requirements of the Directory Access Protocol (DAP). It was specifically targeted at simple management applications and browser applications that provide simple read/write interactive access to the Directory, and was intended to be a complement to the DAP itself.

The LDAP version1 Specification was published in March of 1994. The LDAP version 2 Specification was published as rfc 1777 by the Access Searching and Indexing of Directories (ASID) working group in the IETF in March of 1995. In April of 1996, 40 companies including Microsoft, Netscape, and Novell separately announced support for LDAP protocol in their Directory Services products in order that they may in turn operate with each other and integrate with the Internet. And finally in December 1997, the LDAP version 3 Specifications was published as rfc 2251.

What is LDAP?

- LDAP is the Internet standard for directory lookups, just as the Simple Mail Transfer Protocol (SMTP) is the Internet standard for delivering email and the Hypertext Transfer Protocol (HTTP) is the Internet standard for delivering documents. It creates a standard way for applications to request and manage directory information.

- The Lightweight Directory Access Protocol (LDAP) is a protocol for accessing online directory services. It runs directly over TCP, and can be used to access a standalone LDAP directory service or to access a directory service that is back-ended by X.500.

The LDAP standard defines...

- A network protocol for accessing information in the directory.
- An information model defining the form and character of the information.
- A namespace defining how information is referenced and organized.
- An emerging distributed operation model defining how data may be distributed and referenced (v3).
- Both the protocol itself and the information model are *extensible.*

LDAP Overview

LDAP defines the following components:

- A Data Model—which defines the syntax of the data in the directory.
- An organizational model—which defines how the data is organized in the directory
- The Functional Model—which defines the operations for querying and modifying the directory.
- A Security Model—which defines how the information in the directory is accessed in a secure manner.
- The Topological Model – which defines how the directory service integrates with other directory services to form a global directory service on the Internet.

The Data Model

The data model is centered around *entries*, which are composed of *attributes*. Each attribute has a *type* and one or more distinct *values*. This defines what kind of information is allowed in the values. LDAP data elements are string types.

Which attributes are required and allowed in an entry are controlled by a special *objectClass* attribute in every entry. The values of this attribute identify the type of entry (for example, *person*, *country*, *organization*, and so on). The type of entry determines which attributes are required, and which are optional. For example, the object class *person* requires the *surname* and *commonName* attributes, but *description*, and others are optional.

cn = Test User, o=IDRBT, c=IN

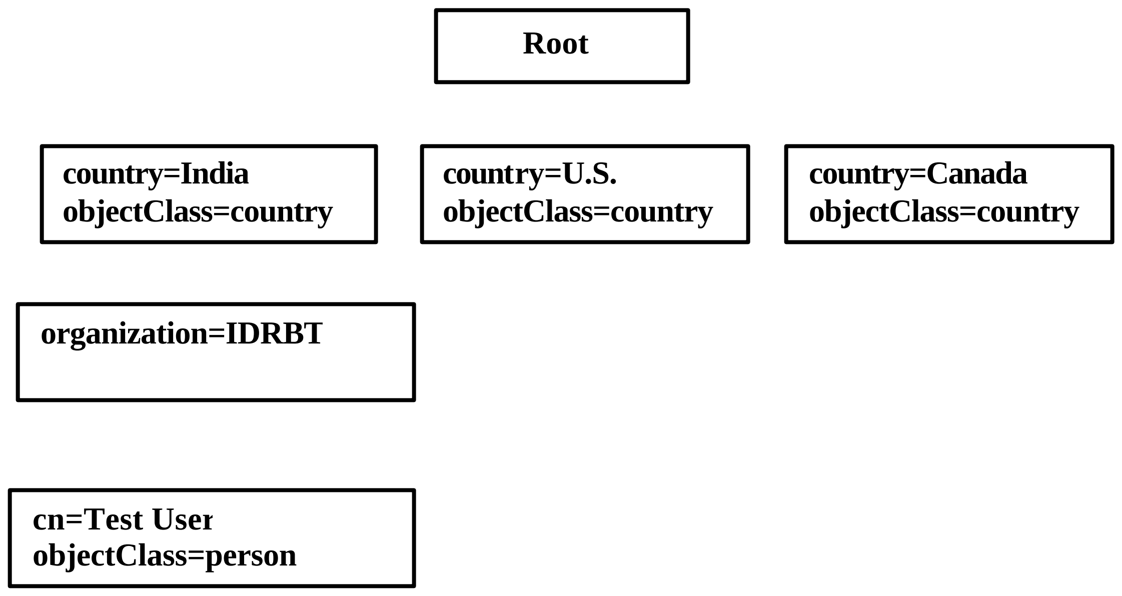

Figure 1

The Organization Model

Entries are organized in a Directory Information Tree. Each entry except the root has a parent entry. Each entry has a fully qualified name, the Distinguished Name (DN). Each component of the DN is called a Relative Distinguished Name

(RDN). The Distinguished Name for any entry is constructed by concatenating the Relative Distinguished Names of the entry's ancestors.

Figure 1 depicts the relationship between entries, attributes, and values and shows how entries are arranged into a tree.

LDAP uses string encodings for representing distinguished names. In the above figure the Relative Distinguished Name for the entry *Test User* is “cn=Test User” and a Distinguished Name of “cn=Test User, o=IDRBT,c=IN”.

The Functional Model

The functional model in LDAP defines the operations for querying and modifying the directory. It defines the operations for:

- Adding an entry to the directory
- Deleting an entry from the directory
- Modifying an already existing entry
- Changing the name of an existing entry
- Querying for an entry in some portion of the directory based on a criteria specified by a query filter.

The Security Model

LDAPv2 defines an authentication model based on clear text passwords. LDAPv3 defines an extensible model based on the *Simple Authentication and Security Layer (SASL)*. SASL uses a layered architecture for using different security providers. The Generic Security Service Application Program Interface (GSSAPI) is used as one of the principal providers of SASL. GSSAPI is a security interface, which defines a common interoperable security system for the Internet. LDAPv3 defines the packet formats of the SASL requests and responses between the LDAP client and server. It supports both security authentication and encryption using different SASL and GSSAPI mechanisms.

In addition to SASL, LDAPv3 also supports secure connections using the *Secure Sockets Layer (SSL)* protocol. SSL enables authenticated and encrypted communication between clients and servers. LDAP SSL connections use port 636, whereas connections using SASL use port 389.

The Topological Model

A major part of LDAP is that you can build a global directory structure using LDAP. It is essentially a directory Web in much the same way that HTTP and HTML are used to define and implement the global hypertext Web. One or more LDAP servers together make up the directory tree. An LDAP client connects to an LDAP server and makes a request. If the information is not available locally, the server attempts to connect to another LDAP server that can fulfill the request. LDAP uses this referral capability to implement a global directory structure of independent LDAP servers that appear to a client to be a single LDAP server.

What kind of data can an LDAP-based directory hold?

You can put just about anything you want into the directory...

- Text
- Photo
- URLs
- Pointers (..say to an user entry)
- Binary data
- Public Key Certificates
- CRLs

NOTE: Though, there may be implementation-dependent limitations on the amount of data of a given type you can store.

Authentication Methods for LDAP

Security Threats

Basic threats to an LDAP directory service include:

1. Unauthorized access to data via data-fetching operations,
2. Unauthorized access to reusable client authentication information by monitoring others' access,
3. Unauthorized access to data by monitoring others' access,
4. Unauthorized modification of data,
5. Unauthorized modification of configuration,
6. Unauthorized or excessive use of resources (denial of service), and
7. Spoofing of directory: Tricking a client into believing that information came from the directory when in fact it did not, either by modifying data in transit or misdirecting the client's connection.

Security Mechanisms

The LDAP protocol suite can be protected with the following security mechanisms:

- Client authentication by means of the SASL mechanism set, possibly backed by the TLS credentials exchange mechanism,
- Client authorization by means of access control based on the requestor's authenticated identity,
- Data integrity protection by means of the TLS protocol or data-integrity SASL mechanisms,
- Protection against snooping by means of the TLS protocol or data-encrypting SASL mechanisms,
- Resource limitation by means of administrative limits on service controls, and
- Server authentication by means of the TLS protocol or SASL mechanism.

Access Control Policy

An access control policy is a set of rules, defining the protection of resources, generally in terms of the capabilities of persons or other entities accessing those resources. A common expression of an access control policy is an access control list. Using Access Control List (ACL), permissions can be set for

- The entire directory,
- A subset of the directory,
- Specific entries in the directory,
- A specific set of entry attributes,
- Configuration tasks for any server (such as directory server),
- A specific user,
- All users belonging to a specific group,
- All users of the directory, or
- For a specific location such as an IP address or a DNS name.

Access Control Factors

When a request is being processed by a server, it may be associated with a wide variety of security-related factors. The server uses these factors to determine whether and how to process the request. These are called access control factors (ACFs). They might include *source IP address*, *encryption strength*, *the type of operation being requested*, *time of day*, etc. Some factors may be specific to the request itself, others may be associated with the connection via which the request is transmitted, and the factors like *time of day* may be "environmental". Access control policies are expressed in terms of access control factors.

Anonymous authentication

Directory operations, which modify entries or which access protected attributes or entries, generally require client authentication. Clients, which do not intend to perform any of these operations typically use anonymous authentication.

Password-based authentication

LDAP implementations support authentication with the "simple" password choice when the connection is protected against eavesdropping using TLS.

Key Aspects of LDAPv1

Protocol elements are carried directly over TCP/IP, bypassing much of the session/presentation overhead.

Protocol data elements are encoded as ordinary strings (e.g., Distinguished Names).

A lightweight BER encoding is used to encode all protocol elements.

What's new in LDAP version 3?

Intelligent referrals: Servers can refer a query to other servers, depending on what is being asked. As a result of that, users can perform Internet-wide address book lookups. In addition, they enjoy the illusion of a single directory even if directory data is scattered across multiple servers (regardless of who owns those servers).

Support for international character sets such as UTF-8 encoding and language attribute tags. Customers can deploy directories using their native languages. As a benefit of this, applications can display multiple languages in the same window. Netscape Directory Server 4.0 correctly sorts more than 38 languages and permits developers to add additional sorting capabilities.

Enhanced security features such as LDAP over SSL and Simple Authentication and Security Layer (SASL) framework. Strong authentication and encryption protects directory data.

Dynamically extensible schema: Schema can be published in the directory and managed through LDAP operations. As a result of this, applications can easily write private data to the directory, making the directory a perfect place to put user preferences, configuration data, and other shared data.

Protocol extensibility: New operations can be added to the protocol without rewriting it. As a benefit of this, users can enjoy new capabilities such as type-down addressing using standards-based products.

LDAPv3 Core Specifications

The LDAPv3 core specifications contain the following:

- Protocol specification
- Attribute syntax definitions
- UTF-8 String representation of Distinguished Names
- String representation of search filters
- LDAP URL format
- X.500 schema definitions

Protocol specification: Key aspects of this version of LDAP are:

- All protocol elements of LDAPv2 are supported. The protocol is carried directly over TCP or other transport, bypassing much of the session/presentation overhead of X.500 DAP.
- Protocol data elements can be encoded as ordinary strings (e.g., Distinguished Names).
- Referrals to other servers may be returned.
- SASL mechanisms may be used with LDAP to provide association security services.
- Attribute values and Distinguished Names have been internationalized through the use of the ISO 10646 character set.
- The protocol can be extended to support new operations, and controls may be used to extend existing operations.
- Schema is published in the directory for use by clients.

Attribute syntax definitions

Explained in Data Model.

UTF-8 String representation of Distinguished Names

The X.500 Directory uses distinguished names as the primary keys to entries in the directory. Distinguished Names are encoded in ASN.1 in the X.500 Directory protocols. In the LDAP, a string representation of distinguished names is transferred. This specification defines the string format for representing names, which is designed to give a clean representation of commonly used distinguished names, while being able to represent any distinguished name.

String representation of search filters

LDAP defines a network representation of a search filter transmitted to an LDAP server. Some applications may find it useful to have a common way of representing these search filters in a human-readable string format.

LDAP URL format

LDAP URL format describes an LDAP search operation to retrieve information from an LDAP directory. It begins with the ldap:// protocol prefix (or ldaps://, if the server is communicating over an SSL connection) and specifies a search request sent to an LDAP server.

LDAP URLs have the following syntax:

ldap[s]://<hostname>:<port>/<base_dn>?<attributes>?<scope>?<filter>
where:

<hostname>	Name (or IP address in dotted format) of the LDAP server (for example, ldap.idrbt.com or 192.202.185.90).
<port>	Port number of the LDAP server If no port is specified, the standard LDAP port (389) or LDAPS port (636) is used.
<base_dn>	Distinguished name (DN) of an entry in the directory. This DN identifies the entry that is starting point of the search. If this component is empty, the search starts at the root of the directory tree.
<attributes>	The attributes to be returned. To specify more than one attribute, use commas to delimit the attributes (for example, "cn,mail,telephoneNumber"). If no attributes are specified in the URL, all attributes are returned.

<scope> The scope of the search, which can be one of these values:

• *base* retrieves information only about the distinguished name (<base_dn>) specified in the URL.

• *one* retrieves information about entries one level below the distinguished name (<base_dn>) specified in the URL. The base entry is not included in this scope.

• *sub* retrieves information about entries at all levels below the distinguished name (<base_dn>) specified in the URL. The base entry is included in this scope. If no scope is specified, the server performs a base search.

<filter> Search filter, to apply to the entries within the specified scope of the search. If no filter is specified, the server uses the filter (objectClass=*).

For example:

To specify a sub tree search starting from "o=IDRBT" that returns all attributes for entries matching "(cn=)", use the following

ldap://172.16.200.50:370/o=IDRBT??sub?(cn=Test User)

The two consecutive question marks—??—indicate that no attributes have been specified. Since no specific attributes are identified in the URL, all attributes are returned in the search.

Benefits / Advantages

LDAP has several benefits that give this standard an advantage over other directory server protocols:

- With LDAP, Internet clients, servers, and applications have a common method for accessing and using directory information.

- LDAP provides a simple search operation that reduces the amount of input and output processing. This is important for Internet implementations that favor more simplistic processing methods for slow Internet connections.
- By using TCP/IP connections, LDAP can be implemented relatively easily on most platforms.
- Directory servers developed under LDAP can support many users and their associated address information.
- With dynamic directory services, LDAP can track more volatile information that changes frequently and must be updated on a regular basis. This is a requirement for dynamic call directories that need to be continually updated to show the most current directory of users.
- It makes modest memory and CPU demands relative to DAP.
- It excludes esoteric X.500 features typical organizations do not need.
- It enjoys widespread industry support.
- It uses a lightweight string encoding to carry protocol data instead of the highly structured and costly X.500 data encoding (which use complicated ASN.1 syntax).
- For OpenLDAP, a free reference implementation is available from the University of Michigan.

Various LDAP Directory Services

Netscape Directory Server: Netscape Directory Server is based on an open-systems server protocol called the Lightweight Directory Access Protocol (LDAP). The directory server is a robust, scalable server designed to manage an enterprise-wide directory of users and resources.

Novell Directory Services: LDAP Services for NDS allow LDAP clients to access the information, which is stored in Novell Directory Services (NDS). LDAP Services for NDS v3 is based on the IETF RFC 2251 LDAP v3 protocol. This means that LDAP clients can perform a v3 bind as described in the LDAP v3 specification. It also means that LDAP Services for NDS can understand the LDAP v3 protocol and can respond to requests that discover available features (authentication mechanisms, controls, schema), demand or request controls, and support extended requests. LDAP Services for NDS v3 gives you the freedom to specify exactly the information you would like to expose to an LDAP client.

OpenLDAP: OpenLDAP is an open source implementation of the Lightweight Directory Access Protocol. The software is a suite of LDAP clients, servers, utilities and developer tools. The suite includes:

- slapd - stand-alone LDAP server
- slurpd - stand-alone LDAP replication server
- libraries implementing the LDAP protocol, and
- utilities, tools, and sample clients.

The OpenLDAP Project is currently developing an LDAPv3 implementation based upon RFC 2251. The OpenLDAP Project is a collaborative effort to develop a robust, commercial-grade, fully featured, and open source LDAP suite of applications and development tools. The project is managed by a worldwide community of volunteers that use the Internet to communicate, plan, and develop the OpenLDAP Suite and its related documentation.

Perl-LDAP: The perl-ldap distribution is a collection of perl modules, which provide an object-orientated interface to LDAP servers.

The perl-ldap distribution has several advantages

- By using the perl object interface the perl-ldap modules provide programmers with an interface, which allows complex searches of LDAP directories with only a small amount of code.

- All the perl-ldap modules are written entirely in perl, which means that the library is truly cross-platform compatible. No C or XS extension are used so no C compiler is needed to install the modules.

Linux Directory Services: Work is going on a project to integrate LDAP and SSL to provide a secure next-generation network directory services architecture to replace the aging Network Information Service (NIS). LDAP will combine several systems that normally have to be maintained separately, such as NT authentication, UNIX authentication, MTA routing information, services/protocols/hosts information, network address books, etc.

Summary

- LDAP is an extensible, vendor-independent, network *protocol* standard -- it supports hardware, software, and network heterogeneity, thus it is *platform-independent.*

- An LDAP-based directory supports *any* type of data

- The LDAP protocol directly supports various forms of *strong security* (authentication, privacy, and integrity) technology

- Because LDAP is an open standard protocol, writing gateways between it and other protocols or systems is relatively straightforward.

- These gateways currently exist...
    - LDAP to X.500 and X.500 to LDAP
    - HTTP to LDAP
    - WHOIS++ to LDAP
    - FINGER to LDAP
    - Email to LDAP

- People are working on...
    - ODBC to LDAP