Computer Network Protocols Assignment
LDAP: The next-generation directory?

Torben Nehmer
torben@nehmer.net

August 25, 2001

Contents

Chapter 1

What is LDAP - an overview

Corporates need directory services on almost every thinkable place. Questions like “What’s the e-mail address of the project leader?” or “Where is the next colour printer?” take up a lots of time in daily live. Especially with all these new information services on todays large networks, such directories became more and more important. Users want centralised accounts and similar system structures independent of the workstation they are currently using.

All this problems do have solutions. The point is, that these solutions are all specialised for one specific problem — users and administrators have to learn how to use many different programs. In addition this leads to the problem of interoperability and interchangeability. Almost any corporate LAN has it’s own directory service, external users from other corporations can’t use these most of the time because the systems are not compatible to each other.

A few years ago there was no single TCP/IP based directory standard, that could be easily implemented and used. Of course, there is the ISO/OSI X.500 protocol standard, a very heavyweighted directory service. It may be suitable for large-scale directories, but is not only difficult to implement but also uses a lots of resources, since its specification require the use of a full OSI protocol stack.

LDAP, the Lightweight Directory Access Protocol, is designed to avoid these weaknesses. Though it is unlikely that a single specification will become a global standard, LDAP has the capability to bring the various existing directories together in a useful manner. The goal of this document is to analyse this standard. This first part will show the more general aspects of LDAP and its history. The second part will go into the protocol itself and show a brief overview about the design and operation of an LDAP service. The third part will draw the conclusion and point out the advantages and disadvantages of LDAP.

1.1 A brief introduction into directory services

To understand LDAP, we first need to review some of the concepts of directories in general and X.500 in special. The most important part of each directory is its namespace. A namespace is a method of referencing each entry in a directory. A phone book for example uses a flat hierarchical namespace consisting roughly speaking of the city and the name of the person. Each namespace defines, which attributes are required to uniquely identify an entry. This *reference* can then be used to retrieve or modify an entry out of this namespace. Such a set could consist of the person's name, the address and the phone number.

The X.500 service also uses a similar kind of hierarchical namespace to reference and retrieve entries. The basic problem here is the fact, that such namespace require complex management schemes. The X.500 model can save almost anything in its database because it is primarily concerned with the structure of the entries, not the way the information is presented to the end user. This is a major limitation that makes X.500 very difficult to use.

1.1.1 Entries, Objects and Object classes

Every entry in a X.500 Directory Information Tree (DIT) is defined as a set of attributes. These attributes are composed of a type and one or more values. These entries are also called objects, since they normally represent an "object" of real life. To represent the type of an entry, each entry requires an associated object class. The object class defines the permitted attributes of an entry.

17 basic object classes are defined in the X.500 standard. In addition X.500 is extensible, so that an implementor can create other, self-defined object classes. The 17 basic object classes include:

- Country
- Locality
- Organisation
- Organisational Unit
- Person

An object is defined by a set of attributes. Again X.500 defines a set of standard attributes for use with a directory. Some of this basic 40 attribute types include:

- Common Name (CN)
- Organisation Name (O)
- Organisational Unit Name (OU)
- Locality Name (L)
- Street Address (SA)

- State or Province Name (S)
- Country (C)

1.2 The History of LDAP

As stated above, the X.500 protocol is built above a complete ISO/OSI protocol stack, which makes its implementation rather complicated. The main reason for X.500's complexity is the fact, that you can save anything in its database. This flexibility not only requires a heavyweight client, that can deal with almost any type of data, but also resulted in a huge network load. This directory access protocol (DAP) made most of the X.500 directory user agents (DUA) almost unusable.

1.2.1 LDAP version 1

This is the point, where LDAP came in. The emphasis in LDAP was – and is – on the *L*, which stands for *lightweight*. The main idea behind LDAP was to create a new protocol for accessing X.500 directories. This protocol should be suitable for desktop computers and the unpredictable bandwidth of the Internet.

Initially, LDAP clients interacted via TCP/IP with a single LDAP server. This LDAP server then interacted with one or more X.500 directories on behalf of the client. The LDAP server still had to use the ISO/OSI protocol stack, but this proxy-like concept restricted the load (both system and network) to a small region of a network. The first specification as of RFC 1487[1] did not provide communication between LDAP servers.

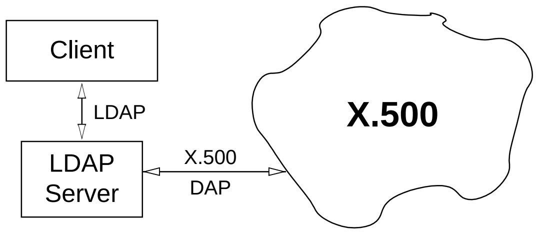

Figure 1.1: LDAP as a X.500 gateway

1.2.2 LDAP version 2

LDAP v1 had a big problem. You had to use a single LDAP server/gateway to access a large X.500 system behind it. This wasn't really efficient, because the LDAP server needed a lot of computing power to do this. So LDAP v2 came into existence in March 1995. LDAP v2 is described in RFC 1777[2].

To get rid of this problem, LDAP v2 supported stand alone servers with their own directory. Though an standalone LDAP directory is not as complex as a X.500 directory, the LDAP subset of X.500 was enough most of the time. In the rare cases, where it was not, you could still use LDAP as a gateway to X.500.

In addition to this, more than one LDAP server can now interoperate with each other. This was achieved with a master/slave concept, where LDAP servers could either be a master in a DIT, with the capability of updating subtree directories on slave servers, which manage an internal part of the DIT. This was good enough for intranet sized directories. The difficulty with this master server is, that it cannot be both master and slave at the same time. This represented a major hurdle for both flexibility and scalability of LDAP-style directories.

In addition to this, there were more limitations that prevented LDAP v2 to become a new IETF ratified Internet standard:

- no extension mechanism
- limited security
- no support for internationalisation
- no referral mechanism

1.2.3 LDAP version 3

Most of these points were taken into account when LDAP v3 has been defined in RFC 2251[4]. Many enhancements were dedicated to the Internet: Support for extended character sets, referrals to other LDAP servers and a extension mechanism to incorporate future improvements with far less problems.

LDAP v3 still lacks a proper authentication mechanism that limits interoperability and security if you require write access to the directory. But as LDAP is designed as an access protocol for a directory with the emphasis on querying rather than updating the repository, this is not necessarily a strong point against LDAP. Basically LDAP uses a clear text password authentication, that can be encrypted using the SASL [1] standard as an extended authentication protocol. Another alternative is the encryption of the connection itself using the Secure Sockets Layer protocol.

One of the most important improvements of v3 is the ability to return referrals to another LDAP server. This is the basic mechanism that makes the interconnection of more then one LDAP server possible. To enable this interaction on a large (i.e. global) scale you need internationalisation support. LDAP v3 achieves this through the use of the ISO 10646 character set.

To be prepared for future changes, LDAP v3 also supports an extension mechanism to incorporate future changes and new requirements into the existing protocol.

[1]Simple Authentication and Security Layer, defined in RFC 2222[3]. It is a basic authentication and security layer which supports encrypts authentication (like Kerberos).

1.3 Where can LDAP (theoretically) be deployed?

LDAP is can be used in many different constellations. Mostly it is used to access X.500 directories. It is easy to adopt this gateway approach to other directory services, providing an unified interface many different directory services. All relevant companies have already declared their support for LDAP as an alternative way to access their directories.

There is also the local standalone LDAP server, which is used for internal information distribution. This is especially interesting as a replacements for the Network Information Service. With this approach you could replace most proprietary systems and bring a heterogeneous network (e.g. WinXX and Unix∗) together. RFC 2307[7] defines some candidates for this approach:

- User and Group information (/etc/passwd, /etc/shadow, /etc/group)
- IP services (/etc/services)
- RPC portmapper information
- Information about distributed filesystems
- IP address information (/etc/hosts, possibly DNS)

This would integrate LDAP into an network wide authentication and authorisation system where a single user/password pair could be used to control access to every resource on a heterogeneous network.

As you can see from this short introduction, LDAP is a very versatile protocol. If the current trend continues, it might very well be the de facto standard to query a directory service.

Chapter 2

A brief introduction into the LDAP protocol

This chapter is divided into two parts. The first part will focus on the hierarchical structure of LDAP while the second half is concerned with the operations that can be performed on this tree. The emphasis will be on searching the tree.

2.1 The LDAP data model

LDAP's origin as an simplified access protocol for X.500 has greatly influenced the structure of the protocol. It uses the same namespace concept like X.500 to reference entries. The main difference is, that LDAP primarily uses strings to represent attributes. Extensions to LDAP also allow the storage of photos, sound streams or public encryption keys (e.g. PGP Keys).

Basically LDAP assumes that the Directory Information Tree is provided by one or more servers. The tree itself is made up of entries of different structure, where each entry has one or more attributes. Each entry in the tree must have an unique identifier. First there is the relative distinguished name (RDN) which identifies an entry uniquely among its siblings. If you concatenate this RDN with every RDN of the parents up to the root node you get the entry's distinguished name (DN), which uniquely identifies an entry in the tree.

Consider the example directory tree in figure 2.1. Within the DMU node the RDN of the faculties would be their OU (Organisational Unit) attribute. To uniquely identify the Computing Science faculty of DMU you need to concatenate every RDN until you get to the root node: `C=UK, O=DMU, OU=Computing Science`.

2.1.1 Attributes of Entries

Each entry in the directory consists of a set of attributes. One ore more values can be associated with a given attribute. Each attribute has a type which is

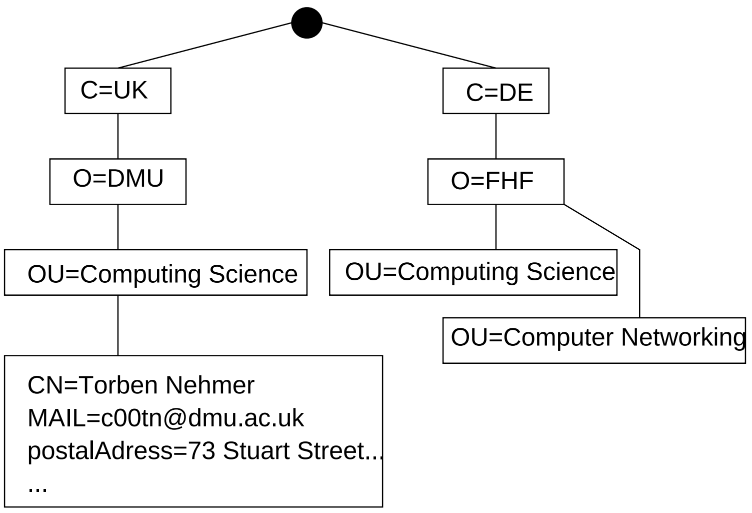

Figure 2.1: A Sample Directory Tree

identified both by a short descriptive name (like "Organisational Unit") and an object identifier (OID). Much like variable types in a programming language, the attribute type defines the type and number of values that can be associated with an attribute. In addition you have the ability to define syntax rules for these values. This can be used to enforce certain integrity constraints, analogous to the rules in a database management system.

For example you could define the attribute `homepage` for all organisational units in the DIT. There can be more than one value for this attribute and they are all IA5-ASCII strings that may be compared case insensitive.

2.1.2 Schemas and data representation

Each DIT has an associates schema. A schema is a colletion of attribute type definitions, object class definitions (see below) and other informations like rules for comparison operations (needed to filter data) or basic permissions (whether to allow add or modify operations). RFC 2252[5] defines a schema for use with LDAP.

LDAP uses octet strings to represent attribute values. RFC 2252 also defines the rules by which the LDAP attribute values are represented as octet strings for transmission.

2.1.3 Object classes

It is mandatory that each entry has an object class attribute. This attribute specifies which other attribute are allowed in this entry by referencing to an attribute set in the current schema. A client can modify, add or remove values of these attributes, but it cannot remove the object class. Servers can even restrict clients to modify the object class to ensure the integrity of the DIT.

Object classes may also form a hierarchical structure. It is possibly to derive one class from another. If a user creates an object, it automatically inherits all attributes of any superclasses from which the object class may be derived. The user then has to supply the mandatory values not only for the selected object class, but also for its superclasses.

2.1.4 Operational attributes

Operational attributes are for administration of the directory itself. Queries do not return them unless the user explicitly asks for them. These attributes are not listed in the schema used by the server. They are only known to the server and are solely used for administrative purposes. Examples of operational attributes are the name of the creator of the entry or the timestamp of the creation.

2.2 The LDAP operations

LDAP operations can be mapped to a strict subset of X.500 operations. However, this mapping is not a one to one mapping between LDAP and DAP. A single LDAP operation to a gateway server can result in a whole series of DAP operations. This also leads to the fact, that it is not possible to estimate the required time to answer a request. Therefore LDAP is designed as an asynchronous protocol. It is guaranteed that every request will be answered, but the sequence of the responses can be in any order.

2.2.1 Basic LDAP operations

There are ten basic LDAP operations. Except of Bind and Unbind, all of those are either applied to a local LDAP-style directory or are performed on a remote directory on behalf of the client.

Bind/Unbind operation Bind establishes a connection to a LDAP server. It allows authentication information to be appended. SASL negotiation may be incorporated into this request. Each LDAP connection has to be initialised with Bind. Analogous Unbind terminates the connection to the LDAP server.

Search operation As the name suggests, Search is used to retrieve certain entries out of the DIT. It can be used to read attributes from a single entry, a complete entry or whole subtrees below a given entry of the DIT. This operation is covered in more detail below.

Modify operation Modify is used to change an entry in the DIT. It is possible to specify more than one modification request in the operation. This operation has the semantics of a transaction. If an error occurs, none of the requests was executed, if no error occurs, every request was executed.

Modify is subject to all limitations of the schema. It is not possible to modify a part of the DN or to remove any mandatory attributes.

Modify DN operation This operation allows the client to modify the least significant part of the DN, the RDN, of a given entry in the DIT.

Add operation Add allows the client to request the addition of an entry into the DIT. The new entry's DN must not exist in the DIT prior to the Add operation.

Delete operation The Delete operation requests the removal of an entry in the DIT.

Compare operation This allows the client to compare an assertion with an entry in the DIT.

Abandon operation Due to the asynchronous nature of the LDAP protocol interactions, this operation can be used to abort a pending operation.

Extended operation This type of interaction represents the LDAP extension mechanism. Generally speaking, this operation encapsulates another, custom operation, in a generic way. Such new operation (like digitally signed requests) can then be defined in future RFCs.

2.2.2 Searching the directory tree

LDAP search result can be filtered in various ways. First you can reduce the result set by applying condition to specific fields, i.e. all students in 4th year or so. Second you can reduce the amount of data you get by requesting only those attributes you are interested in.

In order to get LDAP integrated into the World Wide Web, RFC 2255[6] defines the LDAP URL format, which for example can be used to access a LDAP directory with a regular web browser as frontend. A full explanation of this URL format is far beyond the scope of this document. Instead I will show the general idea with a few examples.

Consider the following URL in the context of the example DIT in figure 2.1:

```
ldap://ldap.dmu.ac.uk/o=DMU,c=UK
```

The first part `ldap://` specifies the schema that you want to use. The default schema for LDAP is `ldap`, defined in RFC 2252[5]. It is also used to tell the client browser to switch to LDAP mode. The next part tells the server to connect to `ldap.dmu.ac.uk`. The last part specifies the DN of the entry you wish to retrieve. In this case it refers to the root record of the DMU.

This URL specification does not require you to specify the ldap host name, you can just leave it blank and let the client choose an appropriate LDAP server. The address of this server could than be distributed with DHCP. The URL would then look like this:

```
ldap:///o=DMU,c=UK
```

Let's assume you are interested only in the postalAddress attribute of this entry. You just append the attribute list delimited by a question mark:

```
ldap:///o=DMU,c=UK?postalAddress
```

Consider you know that Torben Nehmer is a student at DMU and you'd like his postalAddress. You need to perform subtree search of DMU over all entries but you only want to receive those entries whose common name is equal to "Torben Nehmer":

```
ldap:///o=DMU,c=UK?postalAddress?sub?(cn=Torben%20Nehmer)
```

We started with the query above but extended it with `?sub` to a complete subtree search. In addition the third parameter `?(cn=Torben%20Nehmer)` reduces the result to only those entries that have the desired value in the attribute CN.

As you can see, the format of this query is quite intuitive. As long as you have an idea how the directory is structured you can easily formulate such queries. Alternatively a graphical front ends customised by application developers can make this knowledge not necessary.

Chapter 3

Conclusion

LDAP is certainly not the ultimate directory access protocol. Though LDAP now supports standalone LDAP servers, its real strength is acting as a gateway to other, proprietary services. LDAP operations are easy to implement in a client and since it uses the TCP/IP stack its integration in an existing network is easy. Since all important companies (Microsoft, IBM, Lotus, Novell and more) have declared to support LDAP in their directory services it has become the missing link between all these services. The University of Michigan has developed an LDAP daemon which is licensed under the GPL. This implementation can act both as a X.500 gateway and as a standalone server. This daemon can for example be used for small scale standalone LDAP directories, making a cheap directory service available.

The new extension mechanism of LDAP makes it possible to adapt the protocol to new requirements. Though this further increases the flexibility of LDAP it also brings the risk of single vendors creating new "standards". This could lead to similar trouble as there currently is with HTML where both Microsoft and Netscape have their own "standard". The best way in my opinion is to stick with official recommendations on LDAP schemes (as defined in various RFCs).

However LDAP has some limitations. The greatest problem is the limited authentication capabilities LDAP offers. Even with the support for SASL or the use of SSL as an independent encryption layer, the lack of more sophisticated access controls and transaction based authentication will for now prevent LDAP to establish itself as general way to modify directories. In addition LDAP only maps a subset of all X.500 operations. This can hamper employment in special cases where this flexibility is required.

LDAP also lacks a standardised replication system. Netscape is working on replication extensions, but these are not an official part of the LDAP standard. The LDAP daemon of the University of Michigan also has an replication daemon. Both of those approaches are not interoperable. This point is not important if you use LDAP solely for querying a directory. But if you use LDAP in a standalone configuration it can be very important.

Should you use LDAP? It depends. As a pure access protocol it is great. Fast, easy to implement and flexible enough to cover the needs of most directories. If you require update capabilities you have to consider the lack of security in LDAP and you may be better off using some proprietary directory. LDAP is still in a state of change. So the situation could change in the future.

Bibliography

[1] Network Working Group. Rfc 1487: X.500 lightweight directory access protocol. WWW, Usenet, 7 93. http://www.ietf.org/rfc/rfc1487.txt.

[2] Network Working Group. Rfc 1777: Leightweight directory access protocol. WWW, Usenet, 5 95. http://www.ietf.org/rfc/rfc1777.txt.

[3] Network Working Group. Rfc 2222: Simple authentication and security layer. WWW, Usenet, 10 97. http://www.ietf.org/rfc/rfc2222.txt.

[4] Network Working Group. Rfc 2251: Lightweight directory access protocol (v3). WWW, Usenet, 12 97. http://www.ietf.org/rfc/rfc2251.txt.

[5] Network Working Group. Rfc 2252: Lightweight directory access protocol (v3): Attribute syntax definitions. WWW, Usenet, 12 97. http://www.ietf.org/rfc/rfc2252.txt.

[6] Network Working Group. Rfc 2255: The ldap url format. WWW, Usenet, 12 97. http://www.ietf.org/rfc/rfc2255.txt.

[7] Network Working Group. Rfc 2307: An approach for using ldap as a network information service. WWW, Usenet, 03 98. http://www.ietf.org/rfc/rfc2307.txt.