

INTRODUCTION TO LDAP

(Lightweight Directory Access Protocol)

PART I

Introduction

If you've ever worked in a small business, you've no doubt come to realize that a good operations manager can make all the difference in the world. The operations manager can be counted on to know at all times what is happening within the business, where important files are located, who to contact for various business services, and who can take care of a particular task or problem. Although perhaps not completely versed in all the technical details of the business, this person is usually very resourceful, and it's usually well understood that he or she is the one to talk to when you really need results.

From a software point of view, this is exactly what the Lightweight Directory Access Protocol (LDAP) does. This technology is the key to creating better enterprise networks. It's a simple protocol that acts as a storehouse of information for applications, but is also important enough to become a central part of modern operating system services. It binds together system information distributed across multiple computers, with system services and client applications that make it simpler to access user preferences, application configuration data, and security configuration data, and to locate services on the network.

Lapd Applications And Use

LDAP was designed to contain small records of information in a hierarchical structure. This structure appears much like the directory tree of a filesystem, with individual nodes containing attributes and connecting to other subtrees. However, unlike the multimegabyte files in most user directories, the nodes in an LDAP tree are usually pretty small.

Unlike existing database systems, LDAP is not designed to hold many hundreds of thousands of entries. It might be best to think of LDAP as a hierarchically organized lightweight database. An LDAP server may use a small embedded database to contain its information for faster access, but it's nothing like the large commercial databases such as Oracle, Sybase, DB/2 or SQL Server.

To date, there are still only a handful of native LDAP servers. These include the OpenLDAP server (Linux), Innosoft's Distributed Directory Server (Linux), Netscape Directory Server (coming to Linux), Sun Microsystems's Directory Services (Solaris), IBM's DSSeries LDAP Directory (AIX), and the University of Michigan's SLAPD (various forms of Unix). There are a number of other directory service systems that also support LDAP queries. These include Novell's NetWare Directory Services (NDS) 3.0, Microsoft's Active Directory (AD), and Lotus Domino. These primarily use proprietary APIs, but also provide interfaces for LDAP communications. Microsoft's AD, for example, uses LDAP natively but also "extends" the protocol.

Use

There are several ways to look at LDAP. It is A data retrieval protocol -- used directly as an application server to retrieve information from a directory. For example, an LDAP server can contain a list of personnel contact information, like a white pages phone book.

An application service protocol - used by different applications to retrieve the information they require. For example, a user creates a query to be sent to a search engine, the query is matched against an LDAP server, and this points to the place where the actual data is located. Another example of this use would be a DNS server that structures its records internally in an LDAP hierarchy.

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

An interapplication data exchange interface - used by one application to exchange data with another, or as a gateway between two incompatible applications. For example, a Lotus Notes database can store a record into an LDAP server so that it can be retrieved by an Microsoft Outlook client. In this sense, it could also be used as a simple interface for creating vendor-neutral database queries.

A system service protocol - used by an operating system to communicate information between its different resources or components. For example, an LDAP server can contain the access rights of user accounts that are referenced by the login system, the filesystem, and the application execution environment.

To give you a detailed example of how LDAP is involved in system services, take the case of user login authentication. The LDAP service can host information about user accounts and preferences for a group of machines. The login authentication service on each system is directed to query the LDAP servers each time a user attempts to log in. The authentication service checks the results of the query to validate the user. If the user is authorized, it then locates and loads his or her home directory and other personal preferences. If this sounds familiar to you, don't be surprised. Sun's Yellow Pages, Network Information Service (NIS), and NIS+ are all precursors to this method of user authentication. However, these technologies are mostly used only for network-distributed user and host address information. LDAP does a lot more by generalizing the service to support other applications as well.

Rather than creating complex APIs for each type of information service (e.g., a user authentication service, a network host information service, a network filesystem locator service, etc.), LDAP provides a handful of common APIs (see Table 1) to do this. The applications, of course, have to be written to use these APIs properly. Still, it provides the basic service of locating information. LDAP can thus be used to store information for other system services, such as DNS, DHCP, mail, Kerberos, etc.

LDAP Command Description

- Search - Search the directory for matching directory objects.
- Compare - Compare one directory object to a set of data.
- Add - Add a new directory object.
- Modify - Modify a particular directory object.
- Delete - Delete a particular directory object.
- Rename (Modify DN) - Rename or modify the distinguished name of a directory object.
- Bind - Start a session with an LDAP server.
- Unbind - End a session with an LDAP server.
- Abandon - Abandon an operation previously sent to an LDAP server.
- Extended - Extended operations command.

LDAP Communications

There are two sides to LDAP: client-to-server communications and server-to-server communications. Basic client-to-server communications allows user applications to contact an LDAP server to create, retrieve, modify, and delete data with the standard LDAP commands.

Server-to-server communications define how multiple servers on a network share the contents of an LDAP directory information tree and how they update and replicate information between themselves.

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

Currently, the client-to-server communications side is well defined. The basic command set of LDAP is enough to provide all that client applications should need. With the addition of the extended command operation in LDAP version 3, there is even room to add more operations, should the need arise. You can also expect that some vendors will create proprietary extended operations to fit the needs of their environments.

Unfortunately, the server-to-server communications side is still in the Internet draft process. Some vendors, such as Microsoft, have already implemented some of these drafts, along with their own extensions. The drafts should be formalized into full standards between August and December of 1999. A separate protocol, known as the LDAP Duplication Protocol (LDUP), will serve this purpose.

LDAP Architecture

The basic structure of LDAP is a simple tree of information. Starting at a root node, it contains a hierarchical view of all its data and provides a tree-based search system for them (this is one of the most common ways to search data). The actual data contents can vary, depending on how you use the LDAP server. For example, you could have a set of LDAP servers focused solely on servicing login information requests for the thousands of users you support.

Another server could contain host address information used by your DNS and DHCP servers. On the other hand, if you don't have that large of a network, you could even combine the information into a single tree.

The tree itself is called the directory information tree (DIT). Subtrees of the DIT contain all the information on that LDAP server. Every node in the tree is known as an entry, or directory service entry (DSE). These entries contain the actual records that describe different real and abstract objects in the computing environment, such as users, computers, preferences, etc. Some entries can be aliases leading to other parts of the directory, if the real contents already exist somewhere else. The root node of the tree doesn't really exist and can't be accessed directly. There is a special entry called the root directory specific entry, or rootDSE, that contains a description of the whole tree, its layout, and its contents, but this really isn't the root of the tree itself. Each entry contains a set of properties, or attributes, in which data values are stored. Simply said, each entry is a datastructure of variables.

For example, a tree defining the crew of the starship Red Dwarf could start with a hierarchical description of the ranks of the crew, starting from Captain at the top, and going down to Assistant Technician's Mate at the bottom. At each rank, there would be a single person as well as links to those subordinate to that rank. Each person (entry) would have attributes such as his or her name, serial number, office location, e-mail address, etc. Each attribute would in turn have a specific value; e.g., the name attribute could have the value of Arnold Rimmer, while the e-mail address attribute would have the value of rimmer@reddwarf.example. As you can see, the actual data contained can be fairly small. Of course, you could have other attributes, like a 3D representation of the person, which could be a very large piece of data; typically, however, the LDAP node would simply contain a pointer to the place where the actual contents of such a large value would be located.

To refer to each entry in the DIT uniquely, you must use a distinguished name (DN). Normally, this is the very first attribute of the entry. The DN presents the full name of the entry within the entire tree. For example, the DN for our Mr. Rimmer could be:

```
uid=rimmer, ou=technicians, dc=reddwarf, dc=example
```

INTRODUCTION TO LDAP (Lightweight Directory Access Protocol)

Having to specify the DN every single time to access an entry could become cumbersome, especially when you're just looking at a small subset of the tree. If you already know which subset you're examining, you need only specify the name relative to this position. Each of these are thus relative distinguished names (RDNs). For example, relative to reddwarf.example there can be several names:

```
uid=rimmer, ou=technicians
uid=lister, ou=technicians
uid=scutter1, ou=robots
uid=scutter2, ou=robots
uid=kryten, ou=androids
uid=cat,
ou=resultsof3millionyearsofcatinbreedingandevolution
```

The LDIF Record

The LDAP Data Interchange Format (LDIF) is commonly used to define records within the DIT. This is typically a text file describing the various elements within a record and how it is organized. Starting with the DN, the example below shows the user rimmer at the domain reddwarf.example ("example" is a new top-level domain, just like com, net, org, and edu).

```
dn: uid=rimmer, ou=technicians, dc=reddwarf,
dc=example
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Arnold J. Rimmer
cn: The Git
sn: Rimmer
mail: rimmer@reddwarf.example
description: He's just a git
homeLocation: bottom bunk
dislikes: gazpacho soup
objectClass: organization
c: No country
o: Red Dwarf
objectClass: account
uid: rimmer
userPassword: {crypt}aij2kdownm.30da2
uidNumber: 1004
gidNumber: 1001
loginShell: /bin/false
homeDirectory: /home/rimmer
```

Most vendors use proprietary database systems internal to their LDAP server to maintain individual records and only allow updates using API calls. A database makes records faster to access or update. The LDIF is more for human readability, and it approximates the syntax used by the LDAP software within the command operations. LDIF also allows data to be converted between LDAP servers that may use different data formats.

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

The basic syntax is simple, an attribute name followed by a colon and then the attribute value (e.g., cn: Arnold Rimmer). The DN is the first attribute beginning with a dn: followed by the full DN attributes separated by commas (e.g., dn: uid=rimmer, ou=technicians, dc=reddwarf, dc=example).

Long lines are simply wrapped to the next line, with at least one blank space before the text begins. This is to simplify editing. The actual data is stored as a continuous value. Binary data, such as images, signatures, or audio is stored as base 64-encoded data which can be represented in text form.

LDAP Operations

The basic update operations to a directory -- add, delete, modify, and rename – restructure the DIT and its entries. Add and delete are fairly self-evident; they add a new entry or delete an entry based on a DN or RDN. Modify changes the list of attributes, or their values, in an existing entry. The operations rename or modify are used to either move an existing entry to a different location in the directory or to change its name. They work very much like the mv shell command in Unix.

Search and compare send a set of parameters to the server that request objects that match the operation. A fully declared search operation involves eight parameters:

- The tree or subtree you want to search
- How many levels of the tree to search
- How aliases to other parts of the tree should be followed
- The number of entries to be returned
- The time limit for the search
- Whether only the attribute types and not its values are needed
- A filter operation to be applied to the search list
- The list of attributes to be returned

The search filter maps a regular expression string to one or more entries within the tree. There are a number of different search filter types for string, numeric, and Booleans. For strings, you can compare attributes by existence (presence of), lexicographic/alphabetic equality and approximation (equals to, approximates, greater than, less than), and substring matching (substring of). Numeric values can be compared for equality (equal to, less than, greater than), and existence (presence of). Boolean operations (and, not, or) work with any data type.

There is no real copy operation in LDAP, since every entry has to have a unique DN. To approximate a copy, you have to search for an existing entry, then add it to a different section using the same name, or into the same section under a different name.

The LDAP session is fairly basic in nature. It begins when a client issues a bind operation to the server. The server may require an authentication step at this point. Once the connection between the two parties has been established, the client can send any number of LDAP update or interrogation commands. Each command has a sequence number so that the client will recognize it when multiple responses are sent. A single interrogation command, such as a search, can return one or hundreds of records, depending upon the filter operations sent. The clients sending update operations only receive an acknowledgement or an error condition from the server when the operations are complete.

Know where it's at LDAP implements a common platform-neutral structure to information services of the sort needed by operating systems and applications. Although similar services have existed before,

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

none are as widely accepted or implemented across as many platforms and applications as LDAP. By combining the information from various different sources used by the operating system, such as accounting, password, and preference files, LDAP makes it easier to manage system information and to reduce the duplication of such data.

This sums up the workings of the LDAP directory system. It's a fairly simple protocol, but has significant implications for the enterprise. Starting from the basic concept, the directory can span hundreds of servers and millions of entries. The biggest difficulty with LDAP is creating a proper design for the contents and topology of the servers. As NetWare sysadmins will attest, there is a lot of trial-and-error involved in designing a directory for a large corporate environment. Directory services have even resulted in reorganizations of companies themselves, in order to better suit a more flowing hierarchical model. In Part 2, we'll investigate the process of designing directory services in a hope to reduce this trial-and-error process.

PART II

Any NetWare administrator who has had to make the move to NetWare Directory Services (NDS) will tell you that designing a central directory service for your enterprise is no easy task. Although NDS has been around for years, it still takes a lot of planning to get it right. You need knowledge, organizational skills, and a certain degree of persistence to get a directory service working the way it should.

The basic problem, remarkably, isn't a technical one, but rather that a directory service outlines the organizational hierarchy of a company's computing and human resources (HR) infrastructures. Getting the computing resources organized is less taxing than trying to draw boundaries between the various users and their divisions within the company. What's more, it (unfortunately) becomes an almost political endeavor.

Anytime you're charged with arranging human resources -- even if only to ensure that user accounts can access printers across the network -- you have to get the approval of almost every decision maker involved. What's more, this directory is about users, so you'll have to get each user to contribute information (phone numbers, fax numbers, room locations, and so on). This can be time consuming.

Getting Started

Implementing useful directory services involves more than just plugging information into the proper slots. You must first decide what kind of information each user will have to provide, how the information will be represented electronically, what kind of security each type of entry will require, how the information should be spread across multiple servers, and how to modify, access, or manage the information from the network. All of this must be decided before you start collecting or harvesting information.

Since this is a technical magazine and not one for HR administrators, we'll mostly focus on the technical elements of the process. The Lightweight Directory Access Protocol (LDAP) doesn't magically handle all of these issues for you. It's just a simple protocol that retrieves, adds, or modifies information on a directory server. Anytime you create a resource-access system, you must also create policies regulating user access, and determine where it fits into your overall system and how it will be managed. Since much of this isn't defined within LDAP, we have to come up with our own methods.

INTRODUCTION TO LDAP

(Lightweight Directory Access Protocol)

LDAP Models

The LDAP infrastructure is based on the following six models, which describe how it's designed and how it operates:

1. The naming model describes how the tree of data is laid out. Organizing entries by their respective position in the enterprise infrastructure allows you to build a tree-like representation of your entire organization. Each entry can correspond to any type of real or abstract object -- a file, a user account, a printer, a DNS resource record, a user's desktop environment preference, etc. Unlike other tree hierarchies used in computing, like the filesystem tree, the domain name service tree, and so on, each node in the LDAP tree can be of any type and can have any number of child nodes of different varieties. The naming model defines how each entry can be accessed within the context of the entire directory (or relative to other directories). Depending on where the entry is placed in the naming model, you derive either the distinguished name or the relative distinguished name (for more on this, see Part 1 of this series – the URL is in the Resources section below).
2. The information model defines the attributes of each entry and the datatypes of these attributes. Thus, for each entry (i.e., a user account), there must be a definition of how the account name is represented (usually a string of characters, although even this can vary). Each entry must also have a definition of formats for user directory locations, profile information, and so on. The information model defines datatypes (strings, binary data, Boolean values, integers, floating point values, etc.) and places constraints on what constitutes a valid range of information for each attribute.
3. The functional model defines how data is accessed from the directory system. This model basically specifies how the LDAP commands can be used. The commands listed in Table 1 in Part 1 of this series can be grouped into interrogation operations (search and compare), update operations (add, delete, modify, and rename), and session control operations (bind, unbind, and abandon).
4. The security model defines how the directory is secured. This security could be established by subtree, by entry, or even by attribute within an entry. This also takes into account how LDAP sessions are authenticated. Under LDAP version 3, there is a common authentication framework known as the Simple Authentication and Security Layer (SASL). This framework is independent of the actual authentication scheme used, so it works with Secure Sockets Layer (SSL), Transaction Layer Security, Secure MIME, Kerberos, and other security systems and protocols. SASL is built into the bind operation of LDAPv3 when establishing a new connection.

By default, LDAP doesn't specify an access control system to define individual access rights. Since the access control methodology differs depending on the underlying operating system, LDAP leaves it to the implementation of the server software. Microsoft's Active Directory, Netscape's Directory Server, and NDS all use access control lists to provide security down to the directory objects and their attributes.

The replication model is specific to multiserver environments. It defines how the DIT is subdivided or replicated across servers. This isn't a function of LDAP, but many directory products implement a proprietary mechanism of their own design. A standardized protocol called the LDAP Duplication/Update Protocol (LDUP) is still being worked on by the Internet Engineering Task Force. LDUP will allow LDAP to scale to larger organizations with tens of thousands of employees, devices,

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

and applications. Currently, directory vendors have resorted to creating their own versions of LDUP using the Extended command operation in LDAPv3.

The management model defines how the tree is managed, either as a whole or in parts. The directory space can be fairly large in an enterprise network spanning tens or even hundreds of thousands of objects. Such scale can be next to impossible to manage from a single point. Subdivision of management tasks across various locations and IT groups should logically follow the structure of the tree. It's the responsibility of sysadmins in this kind of scenario to assist in keeping track of log files and events, to add or edit access control and security permissions, to troubleshoot application or directory errors, and to manually change or add entries to the tree when necessary.

Some directory service packages, like Microsoft's Active Directory and Novell's NDS, allow you to assign specific jobs to directory administrators so that they don't interfere with each other's subtrees.

These models define how your directory behaves for all parties, users, administrators, and software developers. The users group is most directly affected by the naming, information, and security models. Administrators are primarily concerned with the naming, security, replication, and management models. Developers need to understand the naming, information, functional, and security models. Those who develop tools for directory server management may also find the replication and management models to be relevant.

Planning

Any modern directory service requires careful planning before execution. To begin, you'll have to create a directory task force. Its members should be selected mostly from the IT department, with one or two members drawn from the HR department. The task force is responsible for seeing the project through from beginning to end, and will advise you on policy decisions. Make sure you have approval from the executive ranks to carry out the project at the level for which you're planning.

You may find as you go through the project that secondary objectives begin to appear. This often happens when you talk to those who will be affected by the project. Always take the input as advice and as requests for service, and always evaluate such input carefully before including it in your project. As long as the goals and objectives of the project are kept clear and the changes are minimal, adjustments can be made. But beware of trying to satisfy all the people all the time. It's impossible.

Be forewarned that a directory deployment project can easily get out of hand if you don't take precautions, specifically with regard to who has control of the decision-making process. Someone above you may decide it would be nifty to attach your project to another of theirs, especially if it involves a software development or company reorganization effort. Write down the formal goals of the project and keep this list at eye level. If the project runs on for months, reassert its goals every few weeks so that everyone stays in step. Don't be completely inflexible, but don't be a total rubber band either. After all, rubber bands tend to snap when stretched too thin.

Some companies prefer to gather employee opinions about what should go into the enterprise directory. Although this does make employees feel more involved in the process, it usually results in a diverse mass of ideas that can be too much to handle. It's better to create a representative action group -- consisting of system administrators, human resources staff, legal staff, directory programmers, and middle managers -- whose job it is to determine what will go into the directory. Keep in mind that the larger the group, the longer it will take to come to a decision -- but also that the decision reached in that case will be representative of the opinions of the general population.

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

Goals of your directory service Before you begin, you have to decide on the goals and the scope of the directory service. The goals define how the directory is going to be used. The scope defines who will use the directory. Your task force will be faced with a number of questions, but a very basic set will keep popping up:

What or whose information will be stored in the directory? Which users will be allowed to use the directory? How will users be allowed to use the information in the directory? How and where can users access the information in the directory? How and when can the information in the directory be changed, and who will be authorized to make changes? Who will be responsible for the security and management of the information in the directory? If similar systems exist, what impact will the new directory have on their use?

The primary goal of the directory may be technical (i.e., creating a single enterprise login system), or administrative (i.e., an employee scheduling and calendaring system), or even informative (i.e., an open list of employees and their contact information). However, the questions above still stand, regardless of the directory's goal.

Collecting Information The roughest part of creating a directory service must surely be the collection and collating of all the information that needs to be put into the directory. This isn't always difficult to do, but it can be quite time-consuming if you have lots of users. Your ability to collect any form of data that crosses boundaries between groups or departments in your company is also affected by politics.

Creating a directory service will most likely involve launching a census program on your organization. Although you may have the go-ahead from the top brass to put together a new directory, they may not realize how invasive it can be. All along the way, you're going to encounter obstacles. Individuals will question why you need the information, the legal and human resources departments will scan every questionnaire you create or collect, and some people won't respond even to repeated requests for information -- for a variety of reasons.

Doing a company census to build a directory service is like handpicking ripe peaches. You do it carefully and individually, because the peaches can bruise easily.

One good method is to get the cooperation of the middle managers and supervisors for every group along the way, and explain to them the need for the census and directory service. The middle managers and supervisors will have to field most of the questions from the employees directly, so by educating them you can win valuable allies -- and reduce the hundreds of calls and questions from individuals that you will otherwise have to address yourself!

Once the format for the name space is decided and you have a list of attributes, you can create template-based forms that users or census-givers can fill out. To aid in census taking, you should consider building a Web interface to a temporary database that will store most of the per-user attributes in a single file. This can be processed later, before it is formally entered into the directory. You might also consider hiring temporary staff to perform the census taking.

After you've gathered the information, but before you can begin to make the directory, you'll have to have the following items on hand:

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

- An asset list of all the computers that will use the directory, with brief details of hardware and software configurations for each machine.
- A network topology map showing where the directory client machines will be physically located in the various rooms, buildings, and locations. (A directory client can be an end user's desktop machine, a workgroup server, or any other computer that accesses the directory service for information.)
- A directory software map of all the applications that will use the directory, how they access it, what particular directory services they require, and where they're located on the map.
- A list of directory programming projects indicating all the software developer groups working on projects that interact with the direct services, and where they're located on the map.
- A hierarchy of your organization showing how different workgroups are arranged into departments, and how departments are organized into divisions.
- A list of directory information candidates indicating all users who will have some of their information stored in the directory for any purpose, and -- if possible -- where they're located on the map.
- A list of end users indicating all those who will access the directory service on a regular or semiregular basis to retrieve information, and where they are located on the map.
- A list of administrators who will be responsible for managing the directory services.

Details of Data Collection Items

For your project to be successful, you will have to retrieve quite a volume of information from many different sources. Each item will entail a certain amount of detail that will be required in later stages. To make things easier, start with just a name or identifying tag for each item – a client computer, a programmer, an end user, a job title -- but also create an entry indicating where you can get more information about each item for later use. Eventually, you'll need to look at the details of these other attributes for each user in order to create the full database of directory entries. But to get started, you should be able to work with user names alone.

For the purpose of building the directory, all you need in the asset list is basic information about the computer, such as the vendor model, the amount of RAM, the operating system installed, and a list of all installed software that uses the directory services (see below). Many companies already keep such a list, maintained by either the IT department or the purchasing department. You will probably have to enter the information about installed or to-be-installed directory client software yourself.

To develop a network topology map, you can look to network management packages like Computer Associates's Unicenter TNG, Sun's Solstice Enterprise Manager, or Tivoli Systems's Enterprise Manager. Any one of these packages can help you collect data on the clients on your various LANs and how the networks themselves interconnect. Another tool, Visio Professional, is designed to help you draw accurate, presentable diagrams of your network layout, although most of the above network management packages can be used to do the same thing. Unfortunately, none of these software packages are cheap.

INTRODUCTION TO LDAP (Lightweight Directory Access Protocol)

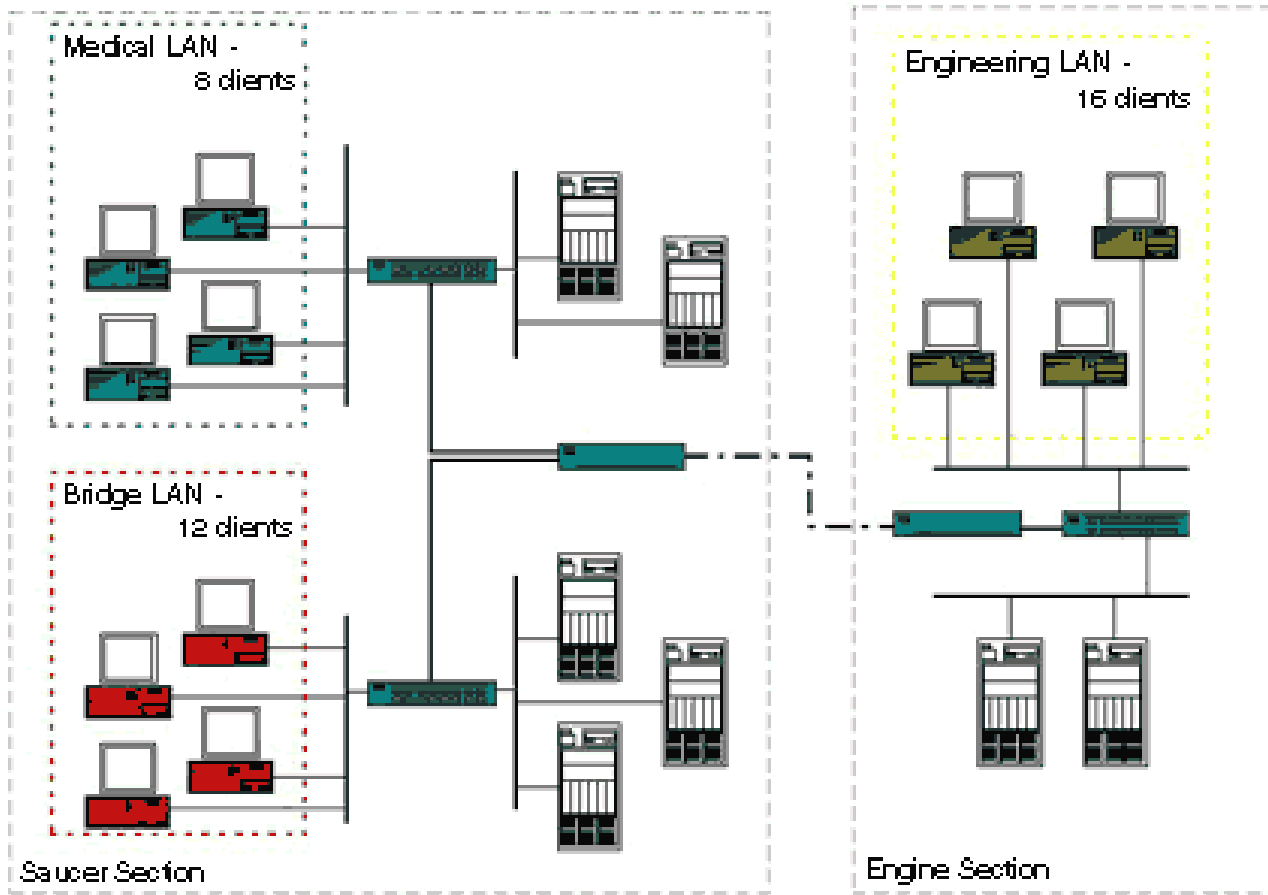


Figure 1
A network topology map

The software map is a logical view of how directory client and server applications communicate. If you're planning directory services for the first time, draw a generic layout of what you perceive will be the future system. If you have existing directory services, you should indicate the platforms on which they run, the version of the directory software, how directory servers are replicated, and how they are managed, and how users access and run the directory client software.

INTRODUCTION TO LDAP (Lightweight Directory Access Protocol)

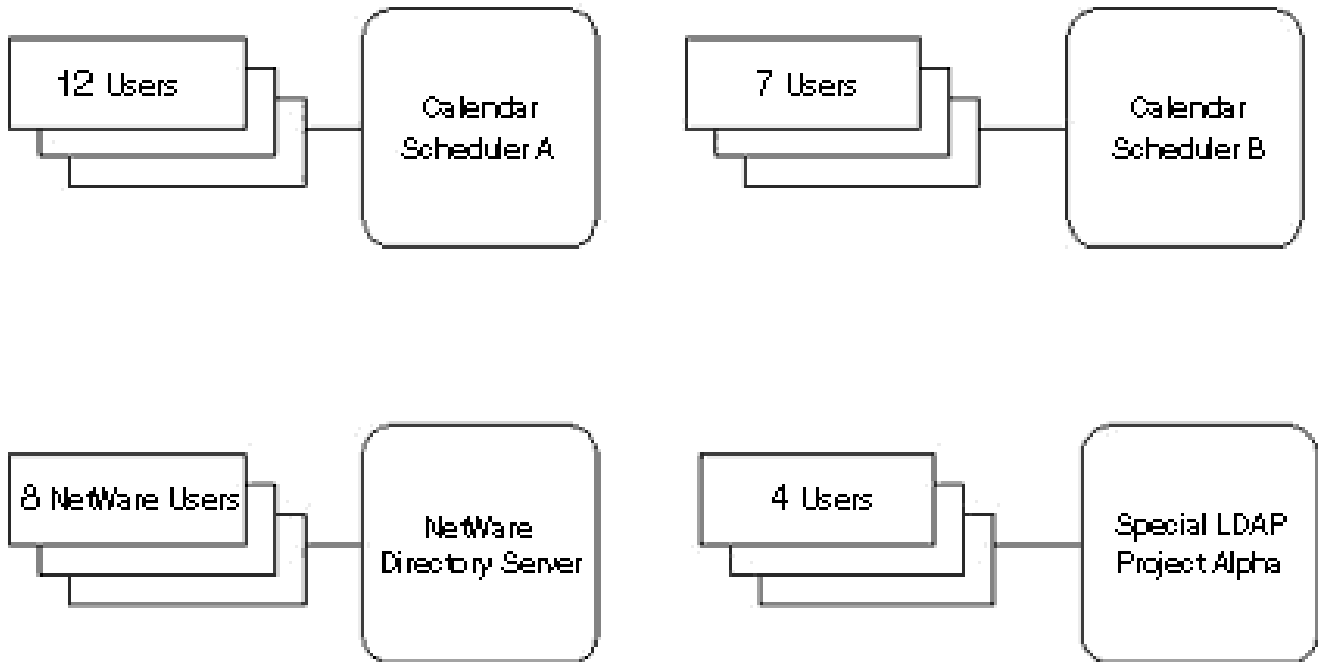


Figure 2
A directory software map

Directory programmers have needs that are different from those of regular users. For the most part, a development project should have its own private directory server. The directory server shouldn't be living with company data during the development stage; the development effort can generate errors or even problems. By giving the directory its own server at this stage, you'll limit problems to a testbed.

The organizational hierarchy of your company can most likely be culled from the HR department, or the staff of the company COO. It could take a lot of time to develop this information on an individual employee level if you have thousands of employees. In such a case, you can probably simply indicate workgroups of people at the lowest level. Visio also has a personnel organization template that will allow you to generate a visual hierarchy much more quickly.

INTRODUCTION TO LDAP (Lightweight Directory Access Protocol)

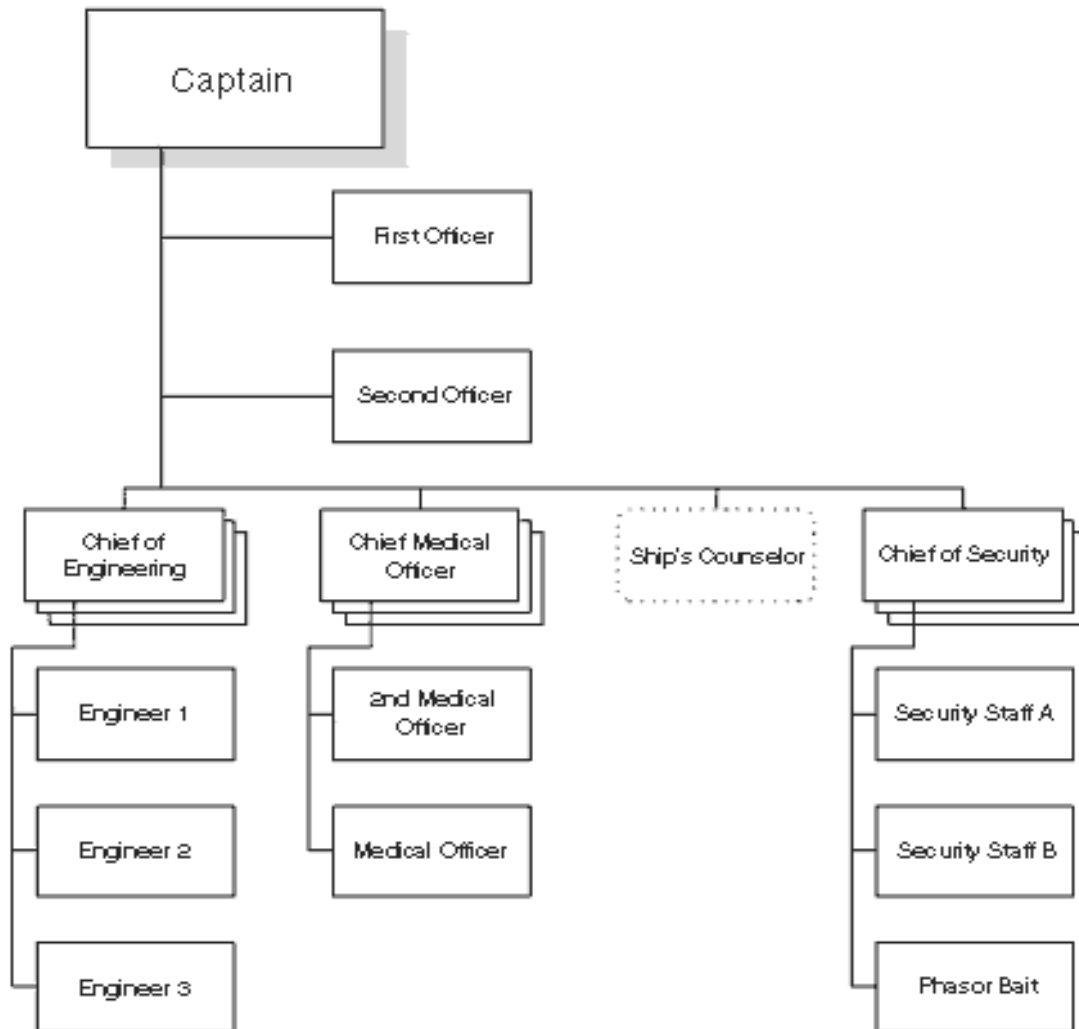


Figure 3
An organizational hierarchy

For the list of directory information candidates, you have to be specific, since each candidate will be a separate entry in the directory. Human resource database packages like those available from PeopleSoft or Oracle can provide you with the list of directory information candidates, or perhaps the employee hierarchy of the organization. At this point of deployment, you do not need the full details of each candidate. Rather, what you should find out is where you can get this information. Once the directory is complete, you may either have to reenter this information into your directory, or tie it to a subset of the HR database that contains the data.

Not everyone in your organization will use the directory. You may even have part of your directory available through a public Web site for the public to use as a contacts list. A greater concern is the list of end users. You should already have prepared the map of the client software, which involves identifying who uses the software and what kind of training they have or need. For each employee,

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

this list must identify the computer(s) used, the directory-related software used, and the user's ability to utilize the software properly.

On to Stage 2

Information gathering will take up the majority of time for your directory deployment project. If your company employs tens of thousands of employees, the required information could even take upwards of a year to accumulate, if not longer! You may find out surprising things about your company when you look at the information, but keep in mind that famous military motto: Ours is not to wonder why; ours is but to do and die.

As you collect the data, you'll also get opinions from individuals and groups on what to include in the directory. Take this information into account, but as indicated before, stick to the plan. In the next part of this series, we'll take a look at how to assemble this collection of data and make sense of it from a directory point of view.

PART III

Now that we've laid the groundwork on how LDAP works and what is required to go about building an enterprise directory, it comes down to putting the pieces together properly. In Part 2, we outlined the eight pieces you'll need for the directory, as follows:

- An asset list
- A network topology map
- A directory software map
- A list of directory programming projects
- An organizational hierarchy
- A list of information candidates
- A list of end users
- A list of administrators

In Part 2, we also presented a few diagrams for an example scenario. We'll reuse that scenario in this article.

Existing directory services If you have existing directory services, it's likely that when you expand the directory to the enterprise they will have to be either revamped or replaced entirely with other directory server products. This can cause quite a bit of disruption, so your existing servers should be included in your directory software map.

You'll recall the seven stages of deployment we outlined in Part 2, but here's the breakdown once again:

- Census
- Interrelation
- Mapping
- Information distribution and replication
- Administrative delegation
- Marketing
- Maintenance

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

The census can be a quite long and involved part of the project compared to the other stages, but is the very basis of the directory project. It is, therefore, essential that you at least begin the census (as outlined in Part 2) before you embark on the remaining stages of deployment.

Here, we pick up where we left off, with the stage following the census.

Interrelation

Interrelation is the process of creating a namespace or naming model based on the information you've gathered about your company and users. Using the organizational hierarchy and the list of directory information candidates, you can map out a detailed plan of who and what is to be contained in your directory, and all the possible distinguished names in your directory.

The designs most companies use are fairly similar. At the very top is the domain for the entire company, which constitutes the directory space. One model creates subsets under this top domain based on identifiable object types such as users, computers, network resources, etc., and then contains them within the various divisions, departments, and workgroups of the organization. Another model follows the hierarchical view of the company, starting with divisions, departments, and workgroups, which are then broken down to encompass individual users, computers, and network resources. In either case, you can see that there are two dimensions to the space: the organizational subdivision and the directory object.

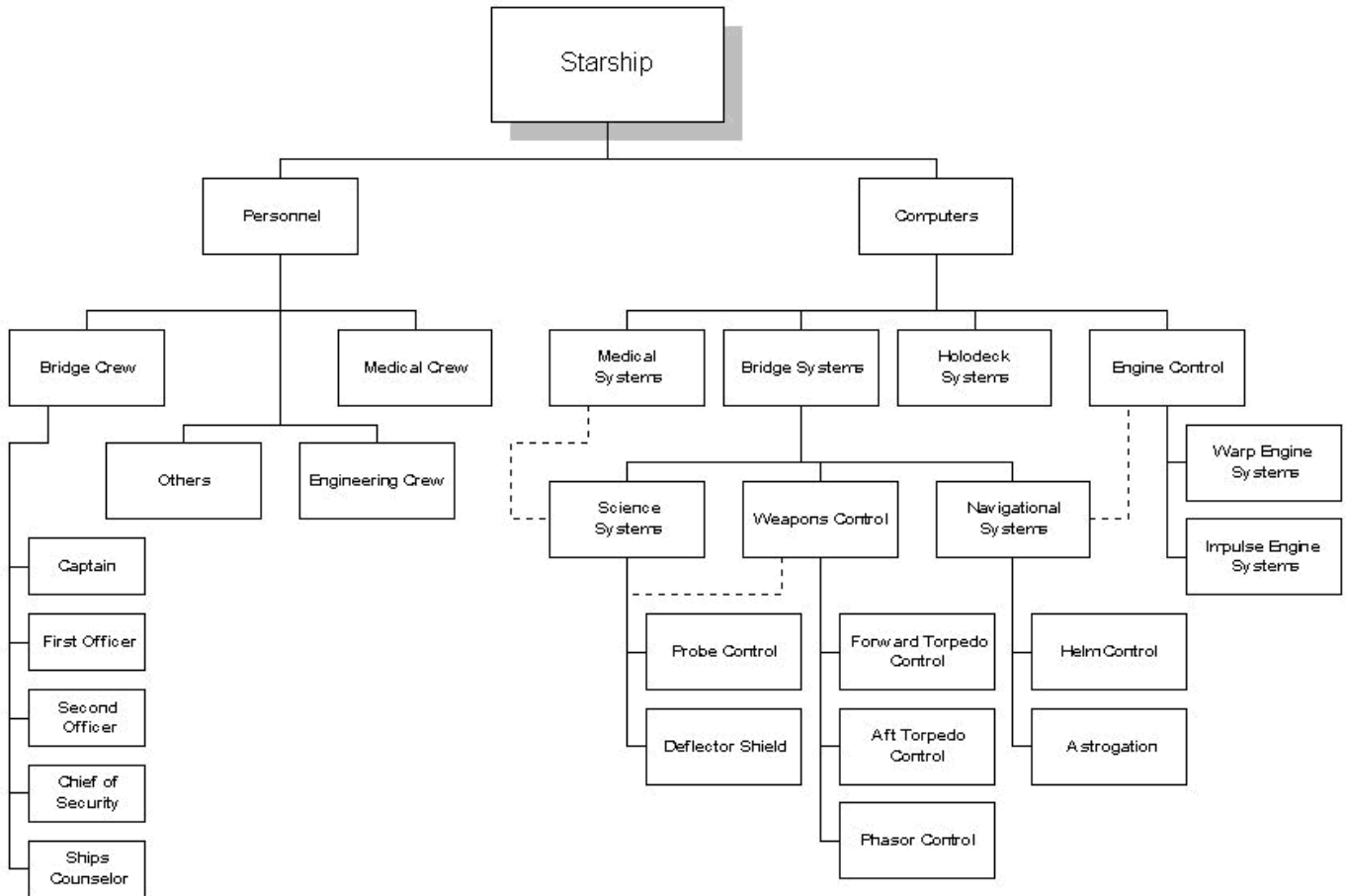
Figure 1 illustrates a design ordered by directory object type (Personnel, Computers), and then by organizational units (Bridge Crew, Medical Crew, Engineering Crew). In this example, all the individuals (not shown) in the organization are included in the directory; i.e., they're all directory candidates.

Mapping, information distribution and replication The second step is to create a map of the relevant computers that will use the directory services. If all your users will have access to the directory to look up phone numbers and other information, the map could well contain all the client computers and many of the servers in your network. Using the network map, software map, and list of directory programming projects, you can draw up a new directory services topology map specifically indicating the computers participating in the LDAP directory.

The map in Figure 2 illustrates the four different directory groups spread across the three different LANs in the organization. In our example, the Medical LAN runs an existing NDS directory for the productivity applications of its members. The Engineering LAN has two of its own directory groups, one for the group calendar and work schedule and the other for a special application development project that works with its own directory. Finally, the Bridge LAN reveals the major portion of the Commanding Officers directory group, indicating their schedules, meeting times, and allotted tasks. This group, however, also has members (e.g., the chief medical officer and chief of engineering) who access the information from the Medical and Engineering LANs.

From this directory services topology map, you can figure out where directory servers should fall into place -- as it will always be close to the locus of the highest population of clients and applications.

INTRODUCTION TO LDAP (Lightweight Directory Access Protocol)



There are four directory servers also represented on this map. The NDS directory server for the Medical LAN, called Europa, appears in red stripes. The directory server for the Engineering LAN, Charon, actually handles three different directory groups: the two directories within the Engineering LAN (the Engineering Schedule and Special Project Alpha) and a cached replicated image of the Command Officers directory group. The Bridge LAN comprises two servers. The directory server Titan is the primary server for the Commanding Officers group. Server Luna contains replicated images of all the information in the directory servers for Commanding Officers group, the Engineering group, and the Medical group.

INTRODUCTION TO LDAP (Lightweight Directory Access Protocol)

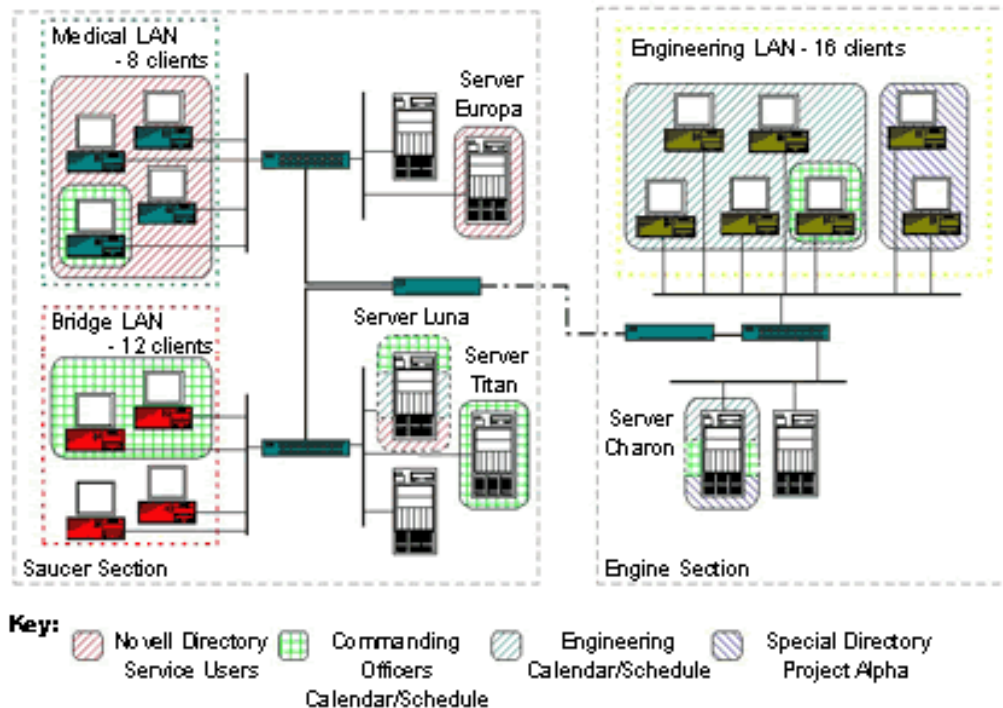


Figure 2
Directory services topology map

The directory servers are placed where they can best serve the local group. In addition, there are replication services for those located remotely. Finally, one server (Luna) contains replicated information of all the other servers, with the exception of the information of Special Project Alpha. Project Alpha is a development project, so its team is likely to test, reboot, or restructure the server often. For that matter, the project really doesn't need to be accessed from anywhere else.

The network map and software map can also combine to illustrate the traffic patterns for your directory. This doesn't, however, provide a precise picture of the amount of traffic the servers will encounter once they're running. Even when taken in combination with the list of end users and the predicted amount of requests they'll generate, you will be able to make an educated guess at best. You'll probably require a few months of monitoring actual network activity to figure out where to properly locate servers on the network to service the highest number of requests and ensure the information travels the shortest possible distance.

Delegating Directory Administration

All directories are information repositories and must therefore have a dedicated administrator. Like a librarian, the person responsible for the directory must ensure its information is correct and properly catalogued. He or she is also responsible for controlling the modification of directory information.

Since a directory space can be fairly huge -- holding millions of entries -- this is a lot of ground to cover for any one person. Furthermore, the information can be relevant to different locations and different departments. You must therefore find a way to share this responsibility among a number of sysadmins, either locally or distributed around your company.

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

How you delegate control of the directory is dependent on the directory software you use. Some products offer a fine degree of control over what each administrator is allowed to do, while others offer a general group of admin accounts responsible for all areas of the directory. Some products provide an easy-to-learn Web interface, while others require the admin to understand cryptic commands.

Whatever the degree of control available to you, you should subdivide the directory tree and assign specific sysadmins to take responsibility of the subtrees. This creates regions of responsibility, and each sysadmin can handle troubleshooting on a regional basis. The subdivisions can be along the lines of personnel groupings, computers and networks, individual directory servers, or according to commonly used directory applications.

Marketing 101 for your LDAP service Once the directory system is up and running and the technical administration details have been worked out, you have to get people actively using the system. This is more of an administrative step than a technical one. Nonetheless, one part of marketing the directory is developing the documentation on how to use it.

This documentation should be targeted at users and the directory-related applications they use. After you install the directory system, you may have to reconfigure the applications to access the directory contents. If this changes the users' applications, they must be briefed on how to use the modified system.

The good thing about directories is that they often fade into the background of applications. Since the applications that use the directory information often present it in their own interface, the user need only know how to use that application. In fact, they might not even know about the directory system behind it at all.

Maintenance and optimization Maintenance is an ongoing responsibility for the directory, and it goes hand-in-hand with technical administration. All the usual server sysadmin tasks are part of the job: backups, system security, log file management, hardware management, and data management. In addition, depending on the uses of your directory, the information will periodically fall out of date and have to be modified by the proper personnel. You should have a policy dictating who is allowed to modify directory information, and how and when they can do so. There is simply nothing more useless than an out-of-date directory, and you don't want all of your hard work to come to naught!

Maintenance involves not only keeping the directory contents safe and current, but also periodically checking the usage and traffic patterns of the servers to determine how well the system is working. Using these traffic patterns you can work to optimize the directory system and the client-to-server and server-to-server communications.

Looking at the traffic, you should attempt to classify the type of requests your directory service gets. You may choose to classify by LDAP operation type, by the groupings in your directory hierarchy, or by the origin of the request (the client machine that generated the request). In any case, the process is much the same.

The first method: Operation type In classification by operation type on a per-server basis, the most basic class is the simple query for data based on a complete or nearly-complete distinguished name. Following this in popularity is the more complex compare query, in which a number of search

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

parameters are used to match one or more data items. From there, we get into the add, delete, and modify operations.

If you have only one server, these operations aren't as significant as you may think. When you have multiple replicated servers at separate sites -- which require such operations to be propagated -- however, employing these operations becomes relatively more expensive. Most costly are the rename or move operations, which can radically restructure your entire directory, and thus affect the namespace of many servers. A simple move of one end-point data item isn't the problem here. It's when you have move operations that relocate a whole subtree that you run into trouble. In this case, the entire directory may be affected. If and when that happens, the use of the add, modify, delete, and move operations together may slow the whole system down considerably.

To avoid that and to help balance the load, you can build a weighting function for your servers. To do so, start by classifying the requests by their operation type (e.g., move, rename, search) and log the number of requests and executed operations for each type. Some operations may be cancelled before they're executed, so these two values, the number of requests and the number of executed operations, can differ.

Next, associate a simple numerical weight or factor with each class. Take a periodic log in which you multiply the weight of each class with the total number of requests of that type, and make a table of the values by class. The total load on the server is the sum of these values. This is a relative number, and it should be used only for comparison between servers.

Based on the data you've recorded from these calculations (performed over several time periods for greater accuracy), you can monitor the overall generation of requests and their distribution by server. This can help you to decide if you need to move some of the information around or if you need to change the replication rules and times.

One or many?

It may turn out that your organization is simply too large or too diverse to contemplate building a single enterprisewide directory service. There may be too many group-specific information attributes that simply do not make sense to everyone in the company. In fact, a single enterprisewide directory often ends up containing only the lowest common denominator of information across the company.

Even worse is when you have a multinational or multidivisional company that maintains very different organizational structures within each location or division. This business model exacerbates the already problematic implementation of directory services at the enterprise level. And, unfortunately, it can be hard to convince the company to reorganize simply to ease the process of putting together an organized directory!

One more thing to look out for when trying to make a single directory service is that different groups and departments may be running different types of directory server products. As indicated earlier, this is a compatibility problem -- and a hard one to fix, given that each department is typically loathe to switch operating systems. You might have no choice but to run several different types of directory server products.

Administration of a single large directory service can be a nightmare. The most common result of such administration is that the information in the directory becomes outdated. Although several directory

INTRODUCTION TO LDAP **(Lightweight Directory Access Protocol)**

services allow you to delegate the authority over subtrees of the directory to individual sysadmins, this isn't a standard feature of LDAP -- which is, after all, just a protocol.

You may end up questioning whether a single directory model is beneficial to you. You may consider setting up several separate directory services for different divisions instead. Or do both -- create a single directory for the organization and several small directories for individual departments. There is no simple if-then-else answer to this problem. Settling on a solution requires an analysis of your IT directory management group's ability to handle the various combinations of issues.

Depending on the size of your organization, plan on taking a few weeks to a few months to architect your directory service. Like any good project manager, you should define milestones in the major steps of deployment, including the census, the name architecture, the server distribution, and the management plan. Each milestone marks an accomplishment by your IT management group (e.g., completing the census of everyone in the sales department). Milestones will help keep your team's spirits up, and will help you track the progress of the directory deployment.

Deploying LDAP will give you a technical and political headache. On the technical side, distributing and integrating servers across a company is a problem of distributed information management. It also entails dealing with compatibility issues, should your company have an existing base of multiplatform directory servers. On the political side, the issue is one of census-taking, a mostly unpopular but necessary aspect of organizational maintenance.

A methodical approach to directory deployment will get you results. For, in the end, the implementation process isn't so much a test of your ability to handle technical complexity as it's a test of your diligence and perseverance.

Once you have the directory service going, the benefits should arise fairly quickly. And once they get the hang of it, users will grow to appreciate the integration of their schedules, word processors, contact databases, and other applications. As this tool saves the time they once spent hunting down and copying information from several disparate sources, you'll find yourself shaking hands and exchanging smiles more often.