



Curso OpenLDAP

22/05/2005
Versión 2

Jose Manuel Suárez
jmsuarez@goa.es

<http://www.goa.es>

Copyright 2005 Jose Manuel Suárez

Permiso para copiar, distribuir y/o modificar este documento bajo los términos de la Licencia de Documentación Libre GNU (GNUFDL), Versión 1.2 o cualquier otra versión posterior publicada por la Free Software Foundation; con las secciones invariantes siendo CURSO DE OPENLDAP, por Jose Manuel Suárez. Una copia de la licencia puede ser encontrada en <http://www.gnu.org/licenses/fdl.html>.

INDICE DEL CURSO

1.- DESCRIPCIÓN DE LDAP.....	4
1.1. Descripción	4
1.2. Ventajas en el uso de LDAP.....	5
1.4. ¿Cuándo resulta interesante usar LDAP?.....	7
1.5.- Diferencias con una base de datos relacional.....	8
1.7.- Servidores LDAP disponibles en el mercado	13
2. ADMINISTRACIÓN DE LDAP	15
2.1.- Introducción a la estructura de árbol.....	15
2.2. Definición de términos	17
2.2.1 Entradas.....	17
2.2.2 Atributos	18
2.2.3 Tipos de Atributos.....	19
2.2.4 LDIF	20
2.2.5 Objetos	21
2.3. Integración con otros sistemas	24
2.4. Casos de éxito y fracaso en la implementación de LDAP	24
3. FUNCIONAMIENTO DE OPENLDAP	26
3.1. Presentación de OpenLDAP	26
3.2. Requisitos.....	27
3.3 Contenido de OpenLDAP.....	27
4. ADMINISTRACIÓN DE OPENLDAP	29
4.1. Cálculo del dimensionamiento del servidor/servidores.....	29
4.2. Nomenclatura.....	30
4.3. Descarga el software	30
4.4. Compilación e instalación	30
Dependencias de OpenLDAP:.....	30

Explicación de los comandos	32
4.5. Comandos del cliente OpenLDAP	33
4.5.1. ldapsearch.....	33
4.5.2 ldapmodify.....	34
4.5.3 ldapdelete	34
4.6 Configuración	35
4.6.1 Configuración de slapd.conf	35
4.6.2 Arranque del servidor.....	35
4.6.3 Primeras entradas del directorio	35
4.6.4 Creación de la estructura de árbol.....	36
4.6.5. Primeras entradas	37
4.6.6 Búsquedas.....	39
4.6.7 Modificación de atributos.....	39
4.6.8 Borrado de atributos.....	40
4.6.9 Inclusión de atributos.....	40
4.6.10 Modificación de un DN.....	41
4.6.11 Borrado de una entrada	41
4.7 Indices	41
4.8 Sintaxis de los filtros de búsqueda.....	42
5.- CONTROL DE ACCESOS.....	47
6. REPLICACIÓN DE DIRECTORIOS.....	52
7. BACKUP Y DISASTER RECOVERY DE BBDD EN LDAP Y OPENLDAP	54
8. MONITORIZACIÓN	55
9. INTEGRACIÓN CON LENGUAJES DE PROGRAMACIÓN	55
9.1 PHP.....	55
9.2 Perl.....	56
9.3 Python.....	57
11. OPTIMIZANDO EL RENDIMIENTO DE OPENLDAP.....	58
12. SEGURIDAD AVANZADA	61
13. INTERFACES GRÁFICOS.....	61

1.- Descripción de LDAP

1.1. Descripción

LDAP ("Lightweight Directory Acces Protocol", en español Protocolo Ligerero de Acceso a Directorios) es un protocolo de tipo cliente-servidor para acceder a un servicio de directorio.

Se usó inicialmente como un Front-end o interfaz final para **x.500**, pero también puede usarse con servidores de directorio únicos y con otros tipos de servidores de directorio.

¿Qué es un directorio?

Un directorio es **una base de datos**, pero en general **contiene información más descriptiva** y más **basada en atributos**.

La información contenida en un directorio normalmente **se lee mucho más de lo que se escribe**. Como consecuencia los directorios no implementan normalmente los complicados esquemas para transacciones o esquemas de reducción que las bases de datos utilizan para llevar a cabo actualizaciones complejas de grandes volúmenes de datos, Las actualizaciones en un directorio son usualmente cambios sencillos de todo o nada, si es que permiten algo.

Los directorios están para proporcionar una respuesta rápida a operaciones de búsqueda o consulta.

Pueden tener **capacidad de replicar información** de forma amplia, **con el fin de aumentar la disponibilidad y fiabilidad**, y a la vez **reducir tiempo de respuesta**. Cuando se duplica la información de un directorio, pueden aceptarse inconsistencias temporales entre la información que hay en las réplicas, siempre que finalmente exista una sincronización.

Hay muchas formas de proporcionar un servicio de directorio. Los diferentes métodos permiten almacenar en el directorio diferentes tipos de información, establecer requisitos diferentes para hacer referencias a la información, consultarla y actualizarla, la forma en que protege al directorio de accesos no autorizados. Algunos servicios de directorios son locales, proporcionando servicios a un contexto restringido. Otros servicios son globales, proporcionando servicio en un contexto mucho más amplio.

¿Un directorio LDAP es una base de datos?

El sistema gestor de una base de datos (Database Management System ó DBMS) de Sybase, Oracle, Informix ó Microsoft SQL Server es usado para procesar peticiones (queries) ó actualizaciones a una base de datos relacional. Estas bases de datos pueden recibir cientos o miles de órdenes de inserción, modificación o borrado por segundo.

Un servidor LDAP es usado para procesar peticiones (queries) a un directorio LDAP. Pero LDAP procesa las órdenes de borrado y actualización de un modo muy lento.

En otras palabras, **LDAP es un tipo de base de datos, pero no es una base de datos relacional**. No está diseñada para procesar cientos o miles de cambios por minuto como los sistemas relacionales, sino para **realizar lecturas de datos de forma muy eficiente**.

Funcionamiento de LDAP

El servicio de directorio LDAP se basa en un **modelo cliente-servidor**.

Uno o más servidores LDAP contienen los datos que conforman el árbol de directorio LDAP o base de datos troncal, el cliente LDAP se conecta con el servidor LDAP y le hace una consulta. El servidor contesta con la respuesta correspondiente, o bien con una indicación de donde puede el cliente hallar más información. No importa con que servidor LDAP se conecte el cliente ya que siempre observará la misma vista del directorio; el nombre que se le presenta a un servidor LDAP hace referencia a la misma entrada a la que haría referencia en otro servidor LDAP.

1.2. Ventajas en el uso de LDAP

Un directorio LDAP destaca sobre los demás tipos de bases de datos por las siguientes características:

- es muy **rápido** en la lectura de registros
- permite **replicar** el servidor de forma muy sencilla y económica
- muchas aplicaciones de todo tipo tienen **interfaces** de conexión a LDAP y se pueden integrar fácilmente
- Dispone de un **modelo de nombres globales** que asegura que todas las entradas son únicas
- Usa un **sistema jerárquico** de almacenamiento de información.

- Permite **múltiples directorios independientes**
- Funciona sobre **TCP/IP** y **SSL/TLS**
- La mayoría de aplicaciones disponen de **soporte** para LDAP
- La mayoría de servidores LDAP son fáciles de instalar, mantener y optimizar.

1.3. Usos empresariales

Dadas las características de LDAP sus usos más comunes son:

- **Directorios de información.** Por ejemplo bases de datos de empleados organizados por departamentos (siguiendo la estructura organizativa de la empresa) ó cualquier tipo de páginas amarillas.
- **Sistemas de autenticación/autorización centralizada.** Grandes sistemas donde se guarda gran cantidad de registros y se requiere un uso constante de los mismos. Por ejemplo:
 - o **Active Directory Server de Microsoft**, para gestionar todas las cuentas de acceso a una red corporativa y mantener centralizada la gestión del acceso a los recursos.
 - o **Sistemas de autenticación para páginas Web**, algunos de los gestores de contenidos más conocidos disponen de sistemas de autenticación a través de LDAP.
 - o **Sistemas de control de entradas a edificios, oficinas....**
- **Sistemas de correo electrónico.** Grandes sistemas formados por más de un servidor que accedan a un repositorio de datos común.
- **Sistemas de alojamiento de páginas web y FTP**, con el repositorio de datos de usuario compartido.
- **Grandes sistemas de autenticación basados en RADIUS**, para el control de accesos de los usuarios a una red de conexión o ISP.
- **Servidores de certificados públicos y llaves de seguridad**

- **Autenticación única ó “single sign-on”** para la personalización de aplicaciones.
- Perfiles de usuarios centralizados, para permitir itinerancia ó “roaming”
- Libretas de direcciones compartidas.

1.4. ¿Cuándo resulta interesante usar LDAP?

Como hemos visto LDAP es una base de datos optimizada para entornos donde se realizan **muchas lecturas de datos y pocas modificaciones o borrados**.

Por lo tanto es muy importante saber elegir dónde es conveniente usarlo. No será conveniente como base de datos para sitios que realicen constantes modificaciones de datos (por ejemplo en entornos de e-commerce)

Normalmente el tipo de preguntas que debes hacerte para saber si LDAP es conveniente para tus aplicaciones son:

- ¿Me gustaría que los datos fueran disponibles desde distintos tipos de plataforma?
- ¿necesito acceso a estos datos desde un número muy elevado de servidores y/o aplicaciones?
- Los datos que almaceno ¿son actualizados muchas veces?, o por el contrario ¿son sólo actualizados unas pocas veces?
- ¿tiene sentido almacenar este tipo de datos en una base de datos relacional? Si no tiene sentido, ¿puedo almacenar todos los datos necesarios en un solo registro?

Pongamos algunos **ejemplos**:

Sistema de correo electrónico

Cada usuario se identifica por su dirección de correo electrónico, los atributos que se guardan de cada usuario son su contraseña, su límite de almacenamiento (quota), la ruta del disco duro donde se almacenan los mensajes (buzón) y posiblemente atributos adicionales para activar sistemas anti-spam o anti-virus.

Como se puede ver este sistema LDAP recibirá cientos de consultas cada día (una por cada email recibido y una cada vez que el usuario se conecta mediante POP3 o webmail). No obstante el número de modificaciones diarias es muy bajo, ya que solo se puede cambiar la contraseña o dar de baja al usuario, operaciones ambas que no se realizan de forma frecuente.

Sistema de autenticación a una red

Cada usuario se identifica por un nombre de usuario y los atributos asignados son la contraseña, los permisos de acceso, los grupos de trabajo a los que pertenece, la fecha de caducidad de la contraseña...

Este sistema recibirá una consulta cada vez que el usuario acceda a la red y una más cada vez que acceda a los recursos del grupo de trabajo (directorios compartidos, impresoras...) para comprobar los permisos del usuario.

Frente a estos cientos de consultas solo unas pocas veces se cambia la contraseña de un usuario o se le incluye en un nuevo grupo de trabajo.

1.5.- Diferencias con una base de datos relacional

Las características de una base de datos relacional (RDBMS o Relation Database Management Systems) son:

- Realizan **operaciones de escritura intensivas**: las bases de datos relacionales están preparadas para hacer un uso constante de operaciones orientadas a transacciones, que implican la modificación o borrado constante de los datos almacenados.
- **Esquema específico para cada aplicación**: las bases de datos relacionales son creadas para cada aplicación específica, siendo complicado adaptar los esquemas a nuevas aplicaciones.
- **Modelo de datos complejo**: permiten manejar complejos modelos de datos que requieren muchas tablas, foreign keys, operaciones de unión (join) complejas...
- **Integridad de datos**: todos sus componentes están desarrollados para mantener la consistencia de la información en todo momento. Esto incluye operaciones de rollback,

integridad referencial y operaciones orientadas a transacciones.

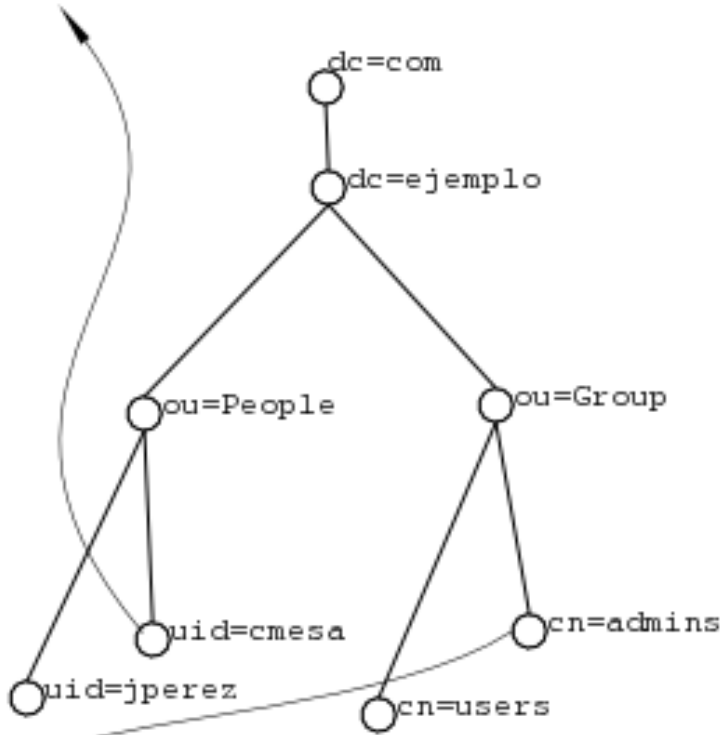
- Además las **transacciones** se efectúan siempre **aisladas** de otras transacciones. De tal forma que si dos transacciones están ejecutándose de forma concurrente los efectos de la transacción A son invisibles a la transacción B y viceversa, hasta que ambas transacciones han sido completadas.
- Disponen de **operaciones de roll-back (vuelta atrás)**. Hasta el final de la transacción ninguna de las acciones llevadas a cabo pasa a un estado final. Si el sistema falla antes de finalizar una transacción todos los cambios realizados son eliminados (roll-back)

Las características de un servidor LDAP son:

- **Operaciones de lectura muy rápidas.** Debido a la naturaleza de los datos almacenados en los directorios las lecturas son más comunes que las escrituras.
- **Datos relativamente estáticos.** Los datos almacenados en los directorios no suelen actualizarse con mucha frecuencia.
- **Entorno distribuido,** fácil replicación
- **Estructura jerárquica.** Los directorios almacenan la información de forma jerárquica de forma nativa.
- **Orientadas a objetos.** El directorio representa a elementos y a objetos. Los objetos son creados como entradas, que representan a una colección de atributos.
- **Esquema Standard.** Los directorios utilizan un sistema standard que pueden usar fácilmente diversas aplicaciones.
- **Atributos multi-valor.** Los atributos pueden almacenar un valor único o varios.
- **Replicación multi-master.** Muchos de los servidores LDAP permiten que se realicen escrituras o actualizaciones en múltiples servidores.

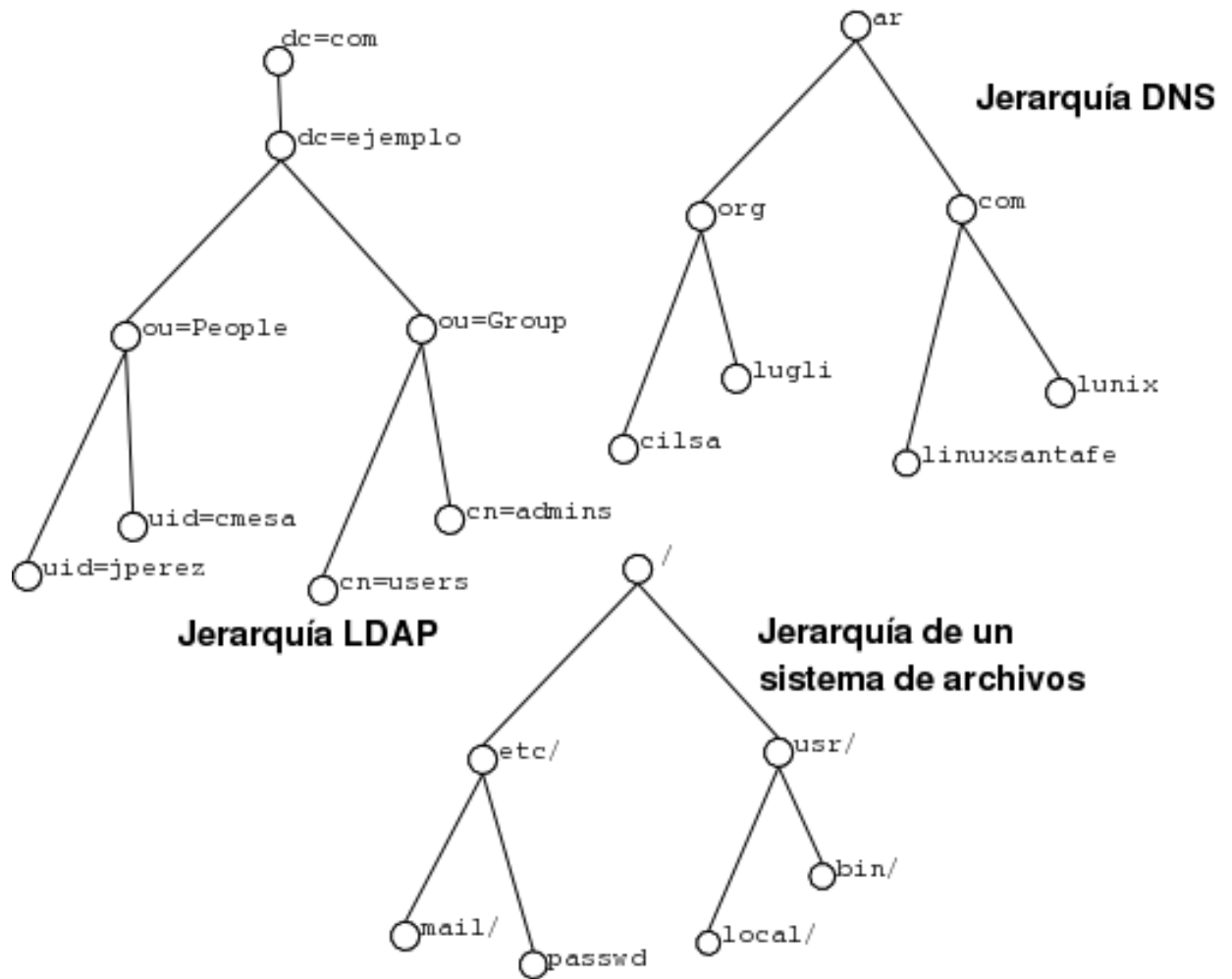
EJEMPLO 1. Cómo se construyen los DN's de las entradas

DN: 'uid=cmesa,ou=People,dc=ejemplo,dc=com'

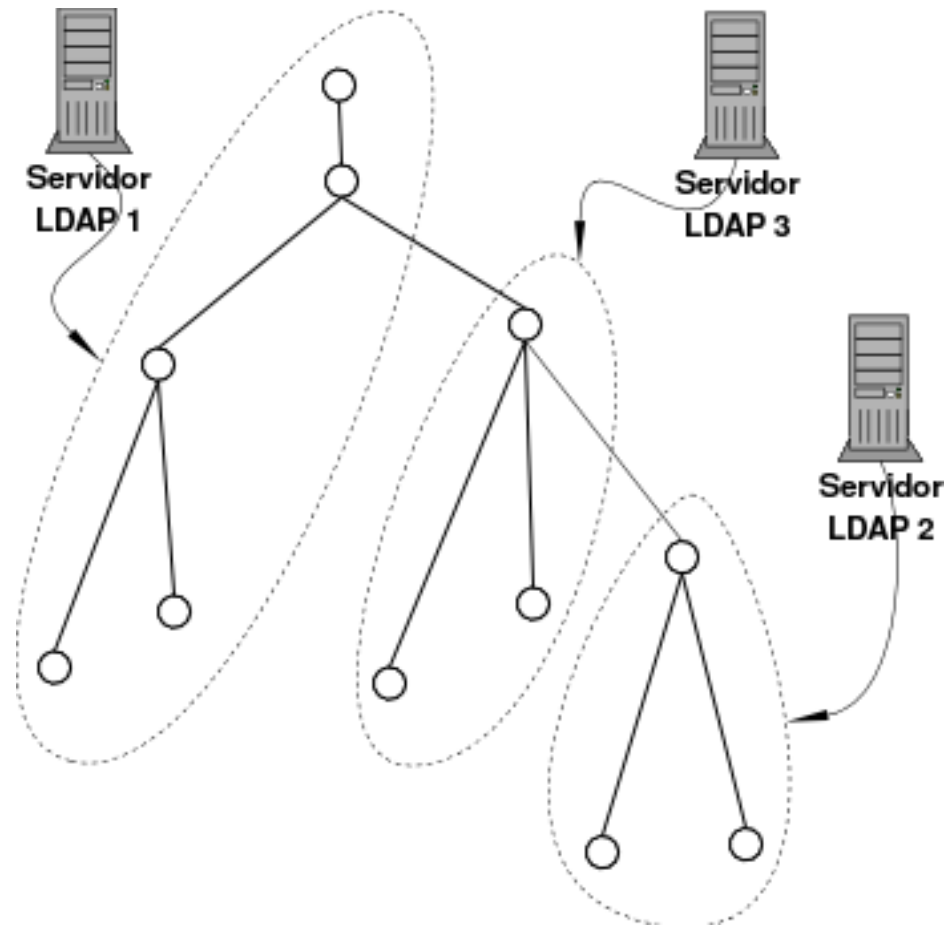


DN: 'cn=admins,ou=Group,dc=ejemplo,dc=com'

EJEMPLO 2. Jerarquías en árbol



EJEMPLO 3. Delegación del árbol LDAP



1.6.- Historia

LDAP aparece con el estándar de los directorios de servicios. La versión original fue desarrollada por la Universidad de Michigan.

La primera versión no se usó y fue en 1995 cuando se publicaron los RFC (Request For Comments) de la versión LDAPv2.

Los RFC para la versión LDAPv3 fueron publicados en 1997. La versión 3 incluía características como las listas de acceso (control access lists) y replicación de directorios.

LDAP RFCs (Request for Comments)

Los RFCs asociados con LDAP son:

RFC1777 - Lightweight Directory Access Protocol. (Obsoletes RFC1487)
RFC1778 - The String Representation of Standard Attribute Syntaxes
RFC1779 - A String Representation of Distinguished Names.(Obsoletes RFC1485)
RFC1823 - The LDAP Application Program Interface
RFC1960 - A String Representation of LDAP Search Filters (Obsoletes RFC1558)
RFC 2251 - Lightweight Directory Access Protocol (v3)
RFC 2252 - LDAPv3 Attribute Syntax Definitions
RFC 2253 - UTF-8 String Representation of Distinguished Names
RFC 2254 - The String Representation of LDAP Search Filters
RFC 2255 - The LDAP URL Format
RFC 2256 - A Summary of the X.500(96) User Schema for use with LDAPv3
RFC 2829 - Authentication Methods for LDAP.
RFC 2830 - Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security.

RFCs relacionados

RFC1274 - The COSINE and Internet X.500 Schema

RFC1279 - X.500 and Domains

RFC1308 - Executive Introduction to Directory Services Using the X.500 Protocol

RFC1309 - Technical Overview of Directory Services Using the X.500 Protocol

RFC1617 - Naming and Structuring Guidelines for X.500 Directory Pilots (Obsoletes RFC1384)

RFC1684 - Introduction to White Pages services based on X.500

RFC2079 - Definition of an X.500 Attribute Type and an Object Class to Hold Uniform

1.7.- Servidores LDAP disponibles en el mercado

Los más usados son:

- **OpenLDAP** – <http://www.openldap.org>
- **Sun SunONE 5.2** – http://www.sun.com/software/products/directory_srvr/home_directory.html
- **Siemens DirX Server 6.0** – <http://www.siemens.com/directory>
- **Syntegra Intrastore Server 2000** - http://www.syntegra.com/us/directory_messaging/
- **Computer Associates eTrust Directory 3.6** - <http://www3.ca.com/Solutions/Product.asp?ID=160>
- **Novell NDS Corporate Edition 8.7.1** - <http://www.novell.com/coolsolutions/nds/>
- **Microsoft ADS - Windows 2000 server edition** - <http://www.microsoft.com/windows2000/technologies/directory/ad/default.asp>
- **Oracle Internet Directory**

Comparativa

Características	OpenLDAP	SunONE	DirX Server	Intrastore	eTrust	NDS	ADS
K4 (UMich) bind	Si	no	?	?	?	no	no
LDAPv2 protocol	Si	Si	?	?	?	?	?
LDAPv3 protocol	Si	Si	Si	Si	Si	?	Si
K5 bind	Si	no	?	?	?	no	Si
SASL GSSAPI (Krb5) Auth	Si	Si	?	?	?	?	Si
Scalable > 200K entries	Si	Si	Si	?	Si	?	?
Solaris 8	Si	Si	Si	Si	Si	Si	no
Search limit tied to binddn	Si	?	?	?	?	?	?
Multi-Master	Si	Si	?	?	?	?	?
Supports "<text>*(space)<text>" search	Si	?	?	?	?	?	?
Soporte técnico	Si*	Si	Si	Si	Si	Si	Si

* Se realiza a través de listas de correo, se puede contratar soporte técnico con empresas como Symas Corp., Mind NV, or Inter7.

2. Administración de LDAP

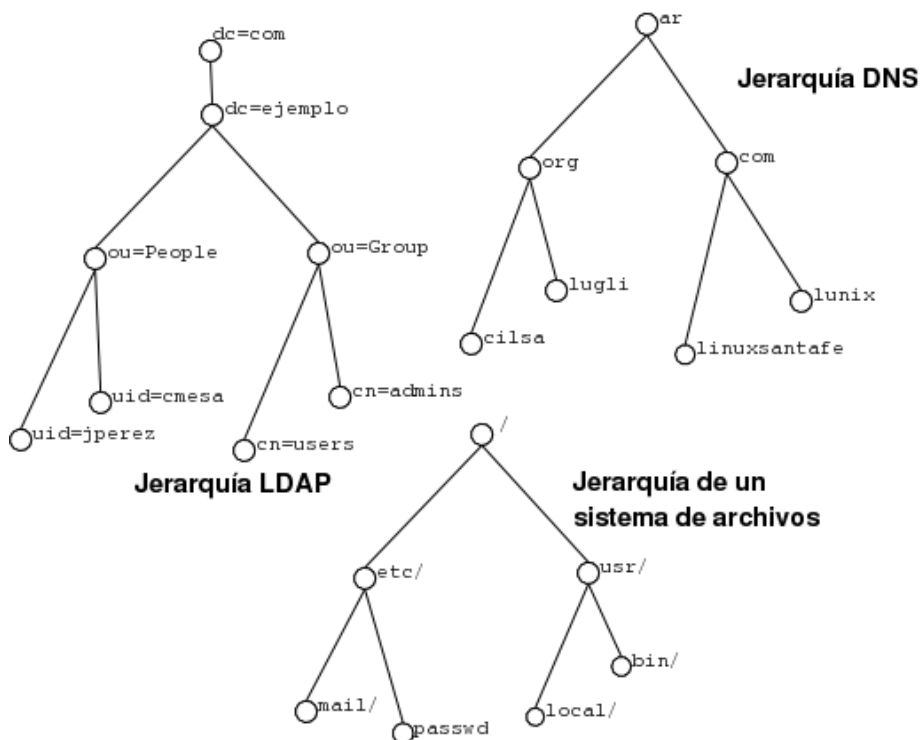
2.1.- Introducción a la estructura de árbol

Tradicionalmente se han usado las estructuras de árbol para jerarquizar la información contenida en un medio. El ejemplo más claro es la estructura de carpetas (directorios) de un sistema operativo. Esta organización nos permite ordenar la información en subdirectorios que contienen información muy específica.

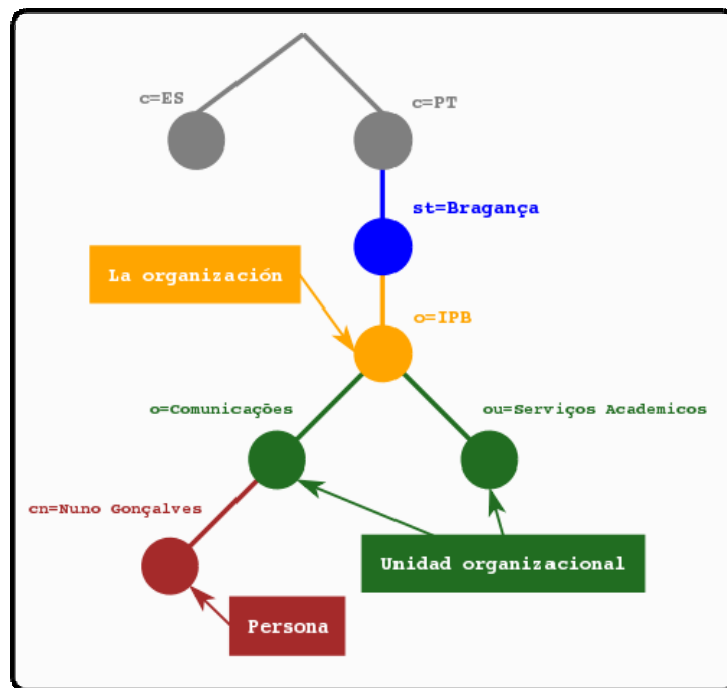
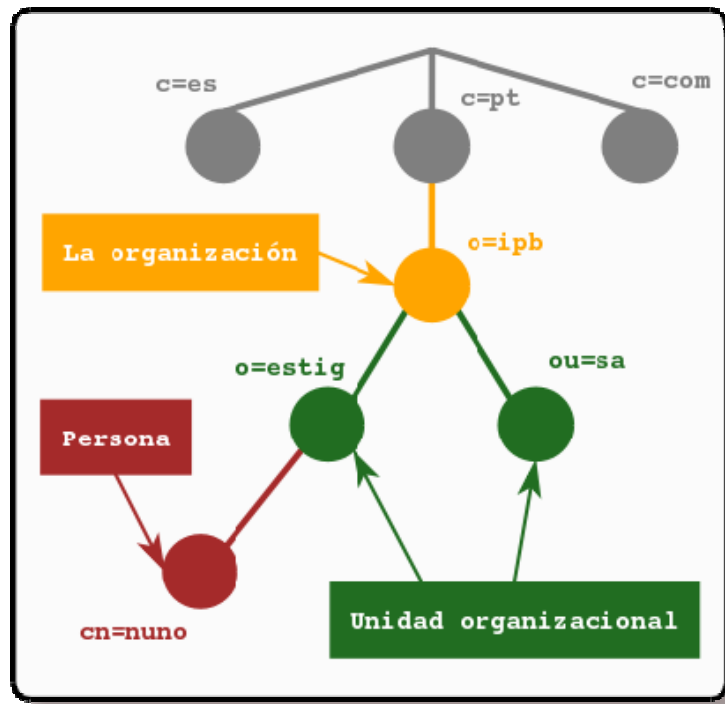
Otro ejemplo muy común son los servidores DNS que nos permiten acceder a distintos servicios concretos que representan un dominio, por ejemplo

www.empresa.com – servidor www principal de la empresa
www.admin.empresa.com – servidor de administración
mail.empresa.com – servidor de mail de la empresa
us.mail.empresa.com – servidor secundario de correo en USA
es.mail.empresa.com – servidor secundario de correo en España

EJEMPLO 2. Jerarquías en árbol



EJEMPLO 4. Dos ejemplos de jerarquías en árbol



2.2. Definición de términos

2.2.1 Entradas

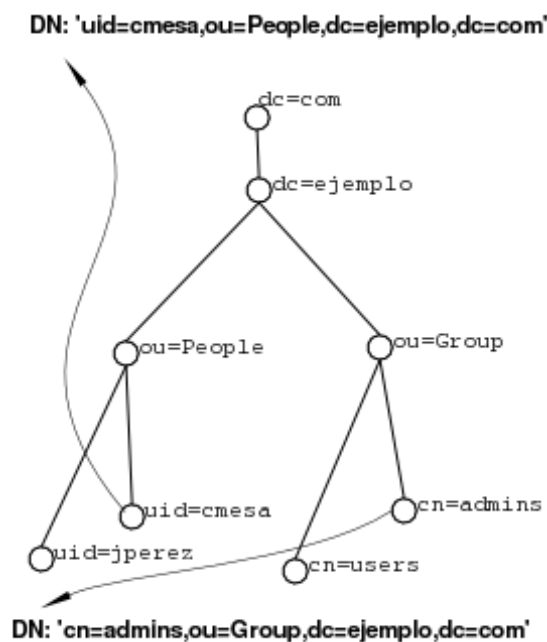
El modelo de información de LDAP está basado en **entradas**.

Una entrada es una colección de atributos que tienen un único y global **Nombre Distintivo (DN)**

El DN se utiliza para referirse a una entrada sin ambigüedades. Cada atributo de una entrada posee un tipo y uno o más valores. Los tipos son normalmente palabras nemotécnicas, como “cn” para common name, o “mail” para una dirección de correo.

La sintaxis de los atributos depende del tipo de atributo. Por ejemplo, un atributo cn puede contener el valor “Jose Manuel Suarez”. Un atributo email puede contener un valor “jmsuarez@ejemplo.com”. El atributo jpegPhoto ha de contener una fotografía en formato JPEG.

EJEMPLO 1. Cómo se construyen los DN's de las entradas



2.2.2 Atributos

Los datos del directorio se representan mediante **pares de atributo y su valor**.

Por ejemplo el atributo `commonName`, o `cn` (nombre de pila), se usa para almacenar el nombre de una persona. Puede representarse en el directorio a una persona llamada José Suarez mediante:

- `cn: José Suarez`

Cada persona que se introduzca en el directorio se define mediante la colección de atributos que hay en la clase de objetos `person`.

Otros atributos:

- `givenname: José`
- `surname: Suarez`
- `mail: jmsuarez@ejemplo.com`

Los **atributos requeridos** son aquellos que deben estar presentes en las entradas que se utilicen en la clase de objetos. Todas las entradas precisas de los atributos permitidos son aquellos que pueden estar presentes en las entradas que utilicen la clase de objetos.

Por ejemplo, en la clase de objetos `person`, se requieren los atributos `cn` y `sn`. Los atributos `description` (descripción), `telephoneNumber` (número de teléfono), `seealso` (véase también), y `userpassword` (contraseña del usuario) se permiten pero no son obligatorios.

Por ejemplo:

```
requires
    objectClass,
    sn,
    cn
    organizationalPerson
allows
    description,
    telephoneNumber,
    seealso,
    userpassword
```

Cada atributo tiene la **definición de sintaxis** que le corresponde. La definición de sintaxis describe el tipo de información que proporciona ese atributo:

1. **bin** binario

2. **ces** cadena con mayúsculas y minúsculas exactas (las mayúsculas y minúsculas son significativas durante las comparaciones)
3. **cis** cadena con mayúsculas y minúsculas ignoradas (las mayúsculas y minúsculas no son significativas durante las comparaciones)
4. **tel** cadena de número de teléfono (como cis, pero durante las comparaciones se ignoran los espacios en blanco y los guiones "_")
5. **dn** "distinguished name" (nombre distintivo)
6. **boolean** cierto/falso, si/no, on/off

Por ejemplo:

attribute	associatedname	dn
attribute	audio	bin
attribute	dn	dn
attribute	documentauthor	dn
attribute	facsimiletelephonenumber	tel
attribute	homephone	tel
attribute	jpegphoto	bin
attribute	labeledurl	ces

2.2.3 Tipos de Atributos

Una definición de tipo de atributo especifica la **sintaxis de un atributo** y cómo se ordenan y comparan los atributos de ese tipo.

Los tipos de atributos en el directorio forman un árbol de clases. Por ejemplo, el tipo de atributo "commonName" es una subclase del tipo de atributo "name".

Hay atributos obligatorios y opcionales listados en la siguiente tabla:

Identificador de Atributo	Descripción del Valor de Atributo
NUMERICOID (obligatorio)	Identificador de Objeto Único (OID)
NAME	Nombre del Atributo
DESC	Descripción del Atributo
OBSOLETE	"true" si es obsoleto; "false" o ausente si no lo es
SUP	Nombre del tipo de atributo superior del que se deriva el tipo de atributo
EQUALITY	Nombre ó OID de la regla de correspondencia si la igualdad de correspondencia está permitida; ausente si no lo está
ORDERING	Nombre o OID de la regla de correspondencia si está permitida la ordenación; ausente si no lo está.
SUBSTRING	Nombre o OID de la regla de correspondencia si está permitida la

	correspondencia de sub-string ausente si no lo está.
SYNTAX OID	numérico de la sintaxis de los valores de este tipo
SINGLE-VALUE	"true" si el atributo no es multi-valor; "false" o ausente si lo es
COLLECTIVE	"true" si el atributo es colectivo; "false" o ausente si no lo es
NO-USER-MODIFICATION	"true" si el atributo no es modificable por el usuario; "false" o ausente si lo es
USAGE	Descripción del uso del atributo

Estos atributos corresponden a la definición de "AttributeTypeDescription" en la RFC 2252.

Un ejemplo de definición de tipo de atributo:

```
attributetype ( 0.9.2342.19200300.100.1.6 NAME 'roomNumber'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )
```

2.2.4 LDIF

Para importar y exportar información de directorio entre servidores de directorios basados en LDAP, o para describir una serie de cambios que han de aplicarse al directorio, se usa en general el fichero de formato conocido como **LDIF (formato de intercambio de LDAP)**.

Un fichero LDIF almacena información en jerarquías de entradas orientadas a objeto. Todos los servidores LDAP que incluyen una utilidad para convertir ficheros LDIF a formato orientadas a objeto. Normalmente es un fichero ASCII.

EJEMPLO:

Un fichero LDIF corriente tiene este aspecto:

Ejemplo 4. Formato LDIF para cuenta de usuario.

```
dn: uid=jmsuarez,ou=People,dc=empresa,dc=com
uid: jmsuarez
cn: Jose Manuel Suarez
objectclass: account
objectclass: posixAccount
objectclass: top
loginshell: /bin/bash
```

```
uidnumber: 512
gidnumber: 300
homedirectory: /home/jmsuarez
gecos: Jose Manuel Suarez,,,,
userpassword: {crypt}LPnaOoUYN57Netaac
```

Como se puede apreciar, cada entrada está identificada por un nombre distintivo:

DN (“distinguished name”, nombre distintivo) esta compuesto por el nombre de la entrada en cuestión, más la ruta de nombres que permiten rastrear la entrada hacia atrás hasta la parte superior de la jerarquía del directorio.

2.2.5 Objetos

En LDAP, **una clase de objetos define la colección de atributos que pueden usarse para definir una entrada**. El estándar LDAP proporciona estos tipos básicos para las clases de objetos:

1. Grupos en el directorio, entre ellos listas no ordenadas de objetos individuales o de grupos de objetos.
2. Emplazamientos, como por ejemplo el nombre del país y su descripción.
3. Organizaciones que están en el directorio.
4. Personas que están en el directorio.

Una entrada determinada puede pertenecer a más de una clase de objetos.

Por ejemplo, la entrada para personas se define mediante la clase de objetos *person*, pero también puede definirse mediante atributos en las clases de objetos *inetOrgPerson*, *groupOfNames* y *organization*. La estructura de clases de objetos del servidor determina la lista total de atributos requeridos y permitidos para una entrada concreta.

Atributos LDAP y clases de objetos usados más frecuentemente

Función	Objectclass	Atributos	Descripción	Valor por defecto	
User accounts	Top		Default		
		Ou	Organizational Unit	Users	
	Person			Owner is a person	
		uid	unix login name	Foo	
		cn	Common Name	Foo Bar	
		sn	Surname	Bar	
	Account			Owner has an account	
	posixaccount			Owner has a Unix account	
		uidNumber	Uid	513	
		gidNumber	Gid	100	
		homedirectory	Home directory	/home/users/foo	
		userpassword	unix password	S3cr3t	
	sambaaccount			Owner has a samba account	
		ntuid	Unknown	uid	
		rid	Unknown	uidnumber	
		Impassword	Lanman password hash	Unused	
		ntpasswd	NT password hash	Unused	
		loginshell	Users shell	/bin/pleurop	
	Machine accounts	Top		Default	
			ou	Organizational Unit	Machines
posixaccount				Owner has a unix account	
		uid	login name	speed\$	
		uidnumber	unix uid	514	
		gidnumber	Gid	100	
		homedirectory	Home directory	Unused	

Podemos **crear nuestros propios objectclass** con los atributos que vayamos a necesitar.

Por ejemplo, vamos a crear un objectclass para un servidor de correo

Función	Objectclass	Atributos	Descripción	Valor por defecto
Cuentas de correo electrónico	Smail	mailID	Identificador de correo (dirección de correo electrónico)	
		mailActive	Un bit que indica si la cuenta está activa o inactiva	1
		mailName	Nombre y apellidos del usuario	
		mailPassword	Contraseña	
		mailQuota	Capacidad del buzón	100
		mailPath	Ruta de almacenamiento	/data/\$mailID

Esta clase de objetos la definiríamos como:

```
objectclass ( 2.16.840.1.113730.3.2.2
  NAME 'Smail'
  DESC 'Cuentas de correo electrónico'
  SUP top STRUCTURAL
  MUST ( mailID $ mailPath )
  MAY ( maiActive $ mailName $ mailPassword $ mailQuota )
 )
```

2.3. Integración con otros sistemas

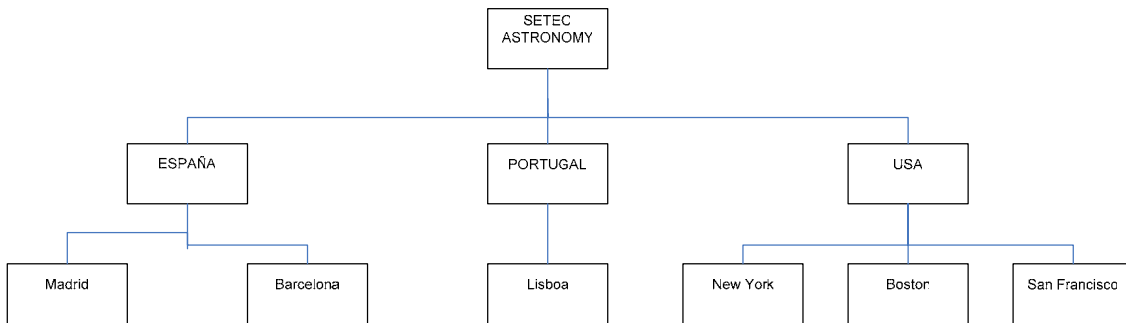
Una vez que hayamos configurado e instalado LDAP lo podemos usar como repositorio de datos para multitud de aplicaciones que disponen de soporte

- Radius
- Samba
- DNS
- Mail Transfer Agents
- Libretas de direcciones
- Servidores FTP
- Servidores HTTP
- Servidores de certificados de seguridad

2.4. Casos de éxito y fracaso en la implementación de LDAP

EJERCICIO 1

A partir del siguiente diagrama de una empresa ficticia cree una estructura LDAP en la que se pueda englobar a todos los empleados de una empresa para poder autenticarlos en la red. Cada sede dispone de tres departamentos, ventas (sales), marketing (marketing) y dirección (management)

**EJERCICIO 2**

Para el esquema anterior elija tres DNs para tres usuarios de diferentes países, sabiendo que cada usuario se identifica por un atributo "uid".

EJERCICIO 3

Cree un archivo LDIF ficticio para estos tres usuarios que ha creado donde se incluyan los siguientes atributos

uid:
cn:
sn:
objectclass:
objectclass:
objectclass:
loginshell:
uidnumber:
gidnumber:
homedirectory:
userpassword:

3. Funcionamiento de OPENLDAP

3.1. Presentación de OpenLDAP

El proyecto OpenLDAP nació como la continuación de la versión 3.3 del servidor LDAP de la Universidad de Michigan cuando dejaron de desarrollarlo.

OpenLDAP es un servidor LDAP que se distribuye bajo licencia GNU (OpenSource), que permite que el software se pueda usar de forma gratuita tanto de forma educativa como profesional. Además disponemos del código fuente para poder realizar nuestras propias modificaciones.

Se puede descargar de forma gratuita en la siguiente dirección

<http://www.openldap.org/software/download/>

A la hora de descargarte OpenLDAP verás que hay varias versiones disponibles:

- **OpenLDAP Release.** Las últimas versiones de OpenLDAP para uso general. OpenLDAP-2.2.23 es la última versión disponible.
- **OpenLDAP *Stable* Release.** Es la última versión que ha sido intensamente probada y suele ser la más fiable de las versiones disponibles.
- **OpenLDAP Test Releases.** Ocasionalmente los programadores de OpenLDAP hacen disponible una versión beta o gamma. Estas versiones son sólo para pruebas y no son para uso general.

En este momento OpenLDAP-2.2.23, es la versión considerada más estable.

Las versiones OpenLDAP 2.x funcionan con la versión 3 de LDAP ([RFC 3377](#)). LDAPv3 es el estándar actual para todos los servidores LDAP.

Los paquetes que incluyen las distribuciones de OpenLDAP son:

- servidor LDAP ([slapd](#))
- servidor de replicación LDAP ([slurpd](#))
- Software Development Kit ([ldap](#))
- Utilidades, herramientas, ejemplos...

Toda la documentación sobre el producto puede consultarse en

<http://www.openldap.org/doc/>

El coordinador del proyecto OpenLDAP se llama **Kurt D. Zeilenga** y es fácil contactar con él a través de las listas de correo.

Además de desarrollar OpenLDAP Kurt trabaja en IBM donde es Ingeniero de investigación de Servicios de Directorio y desarrollador de IBM Linux Technology Center.

3.2. Requisitos

- **Sistema operativo.** OpenLDAP funciona en los siguientes sistemas operativos:

Apple Mac OS X
Linux: Debian, RedHat, Suse, Fedora, Mandrake...
FreeBSD
IBM AIX
Microsoft Windows 2000/NT
NetBSD
Solaris

3.3 Contenido de OpenLDAP

El paquete **OpenLDAP** contiene los archivos binarios: **Idapadd**, **Idapcompare**, **Idapdelete**, **Idapmodify**, **Idapmodrdn**, **Idappasswd**, **Idapsearch**, **Idapwhoami**, **slapadd**, **slapcat**, **slapd**, **slapindex**, **slappasswd**, **slurpd**, **liblber** y **libldap**.

Descripciones:

Idapadd abre una conexión a un servidor LDAP, enlaza y añade entradas.

Idapcompare abre una conexión a un servidor LDAP, enlaza y hace una comparación usando los parámetros especificados.

Idapdelete abre una conexión a un servidor LDAP, enlaza y borra una o mas entradas.

Idapmodify abre una conexión a un servidor LDAP, enlaza y modifica entradas.

Idapmodrdn abre una conexión a un servidor LDAP, enlaza y modifica el RDN de las entradas.

ldappasswd es una herramienta para establecer la contraseña de un usuario LDAP.

ldapsearch abre una conexión a un servidor LDAP, enlaza y hace una búsqueda usando los parámetros especificados.

ldapwhoami abre una conexión a un servidor LDAP, enlaza y realiza una operación whoami.

slapadd se usa para añadir entradas especificadas en el formato Intercambio de Directorio de LDAP (LDIF) en una base de datos slapd.

slapcat Se usa para generar una salida LDAP LDIF basada en el contenido de una base de datos slapd.

slapd es el servidor LDAP independiente.

slapindex se usa para regenerar índices slapd basados en el contenido actual de una base de datos.

slapasswd es una utilidad de contraseñas OpenLDAP.

slurpd es el servidor réplica LDAP independiente.

liblber y libldap

Estas librerías dan soporte a los programas LDAP y suministran funcionalidad a otros programas que interactúan con LDAP.

4. Administración de OPENLDAP

4.1. Cálculo del dimensionamiento del servidor/servidores

Elige bien tu plataforma hardware:

- **Procesador:** Normalmente servidores multiprocesador.
- **Discos:** Para OpenLDAP lo más óptimo es que uses un disco duro para el sistema operativo (preferiblemente en RAID) y un disco separado para la base de datos (normalmente sin RAID)
Para elegir un disco duro adecuado puedes consultar <http://www.kegel.com/drives/>
Esta es la optimización más importante.
- **Tamaño de la memoria:** Dependerá del número de entradas que quieras almacenar y del número de atributos que use cada entrada. También de las pruebas de carga que realices y sus resultados. Normalmente necesitarás entre 2 GB y 4 GB.

Instalación del sistema operativo

- Elegir una instalación simple, sólo con los complementos imprescindibles.
- Actualizar el sistema operativo con los últimos parches o service packs (ej.: sunsolve.sun.com, redhat.com, windowsupdate.microsoft.com....)
- Elegir un sistema de archivos adecuado, normalmente:
 - o Ext3 para Linux
 - o UFS con LOGGING para SolarisEs muy importante activar el 'logging' en UFS porque incrementa el rendimiento de forma muy apreciable.
- Parar todos los servicios y demonios que no se vayan a usar.
- Securitizar el servidor
- Optimizar los parámetros del sistema operativo (hay diversos métodos de hacerlo que no se incluyen en este curso)
- Optimizar la configuración de la pila TCP.

4.2. Nomenclatura

Antes de instalar el servidor elige una nomenclatura de directorios para todos tus trabajos (debes pensar siempre en las actualizaciones posteriores a la instalación actual)

Un ejemplo es usar un directorio como /opt/apps

/opt/source/openldap-2.2.23 directorio con el código fuente

/opt/apps/openldap-2.2.23 directorio para tu aplicación

/opt/apps/openldap es un link a la aplicación

/opt/data/openldap es el directorio para la base de datos

/opt/backup es el backup diario

Con una nomenclatura como esta es muy fácil implementar actualizaciones de la aplicación.

- Elige una nomenclatura para todos los objetos, atributos, usuarios....

4.3. Descarga el software

Descarga la última versión estable de OpenLDAP (en este momento la 2.2.23) desde <http://www.openldap.org/>

Verifica la firma MD5 del paquete que te has descargado usando el siguiente comando:

```
[root@dep tmp]# md5sum openldap-2.2.23.tgz
```

Ahora verifica que la firma es exactamente la misma que la contenida en un archivo llamado openldap-2.2.23.md5 que te puedes descargar desde el servidor FTP de OpenLDAP.

Desconfía de servidores FTP que no sean los oficiales y de paquetes cuya firma MD5 no coincida con los de las páginas oficiales de OpenLDAP.

4.4. Compilación e instalación

Dependencias de OpenLDAP:

Requerida: Berkeley DB

Opcionales: OpenSSL, readline, GDBM, Cyrus y Heimdal

Sitúate en el directorio donde hayas descomprimido el código fuente y ejecuta los comandos:

```
./configure  
make  
make depend  
su make install
```

Conseguiremos que nuestro software esté mucho más optimizado si desactivamos antes de compilar las opciones que no vamos a usar y activamos otras específicas, por ejemplo.

```
./configure \  
--disable-debug \  
--disable-ipv6 \  
--enable-crypt \  
--without-tls \  
--with-threads
```

Esto le prepara a OpenLDAP para configurarse de una forma especial:

- Sin soporte para debugging.
- Sin soporte para IPv6.
- Activa el soporte para passwords cifrados (con crypt).
- Desactiva la encriptación TLS/SSL.
- Activa el soporte para threads.

Ejemplo:

Instala OpenLDAP ejecutando los siguientes comandos:

```
./configure --prefix=/usr --libexecdir=/usr/sbin \  
--sysconfdir=/etc --localstatedir=/var/lib \  
--disable-debug --enable-ldbm &&  
make depend &&  
make &&  
make test &&  
make install
```

Explicación de los parámetros

`--sysconfdir=/etc` : Establece la ruta al fichero de configuración para evitar el directorio por defecto `/usr/etc`.

`--libexecdir=/usr/sbin` : Pone los ejecutables del servidor en `/usr/sbin` en vez de en `/usr/libexec`.

`--enable-ldb` : Construye **slapd** usando como interfaz primaria de bases de datos Berkeley DB o GNU Database Manager.

`--disable-debug` : Desactiva el código de depuración.

make test : Verifica que el paquete se ha compilado correctamente

4.5. Comandos del cliente OpenLDAP

4.5.1. Ldapsearch

La herramienta de línea de comando `ldapsearch` busca entradas específicas en el directorio.

La sintaxis de `ldapsearch` es:

ldapsearch [*opciones*] *filter* [*parametros*]

Se entiende por filtro la condición que se debe cumplir para la búsqueda de entradas.

Parámetros obligatorios	Descripción
-b <i>basedn</i>	Especifica el DN base para las búsquedas
-s <i>scope</i>	Alcance de la búsqueda: base, one ó sub.

Parámetros opcionales	Descripción
-A	Solo muestra los nombres de los atributos (no los valores)
-a <i>deref</i>	Referencias a los alias: never, always, search, or find
-B	Permite imprimir valores no-ASCII
-D <i>binddn</i>	Cuando se autentica con un directorio, permite especificar la entrada <i>binddn</i> . Usar con la opción -w <i>password</i> .
-d <i>debug level</i>	Nivel de debug
-E " <i>character_set</i> "	Especifica la página de codificación de caracteres
-f <i>file</i>	Ejecuta la sentencia de búsquedas archivadas en el archivo <i>file</i>
-h <i>ldaphost</i>	Conecta al servidor LDAP en la dirección <i>ldaphost</i> . El valor por defecto es <i>localhost</i> .
-L	Muestra la entradas en formato LDIF
-l <i>timelimit</i>	Timeout en segundos antes de abandonar una búsqueda
-p <i>ldapport</i>	Conecta al servidor en el puerto TCP especificado en <i>ldapport</i> . Por defecto conecta en el puerto 389.
-S <i>attr</i>	Ordena los resultados por el atributo <i>attr</i>
-v	Modo extendido
-w <i>passwd</i>	Especifica la contraseña para hacer el bind (para autenticación simple)
-z <i>sizelimit</i>	Especifica el número máximo de entradas que pueden ser mostradas.

Se puede usar el comodín * para las búsquedas pero no es aconsejable si se espera un número muy alto de atributos.

4.5.2 Idapmodify

La herramienta de línea de comando Idapmodify permite cambiar, añadir o borrar atributos.

La sintaxis de Idapmodify es:

```
Idapmodify [opciones] -f archivo
```

4.5.3 Idapdelete

La herramienta de línea de comando Idapdelete permite borrar entradas.

La sintaxis de Idapdelete es:

```
Idapdelete [opciones] "DN_de_la_entrada"
```

El siguiente ejemplo usa el puerto 389 en un host llamado server1

```
Idapdelete -p 389 -h server1 ou=jmsuarez,ou=mad,ou=es,dc=setec,dc=com"
```

4.6 Configuración

4.6.1 Configuración de slapd.conf

Edita el archivo slapd.conf (normalmente instalado en `/usr/local/etc/openldap/slapd.conf`) que contiene la definición de la base de datos

```
database      bdb
suffix        "dc=<MY-DOMAIN>,dc=<COM>"
rootdn        "cn=Manager,dc=<MY-DOMAIN>,dc=<COM>"
rootpw        secret
```

Sustituye MY-DOMAIN y COM con los nombre adecuados que hayas elegido, en nuestro ejemplo

```
database      bdb
suffix        "dc=setec,dc=com"
rootdn        "cn=root,dc=setec,dc=com"
rootpw        secret
```

4.6.2 Arranque del servidor

Inicializa el servidor SLAPD

En Unix: **su root -c /usr/local/libexec/slapd**
En Windows: **./slapd -d 1**

Para verificar que el servidor está funcionando y ha sido configurado correctamente puedes realizar una búsqueda usando el comando **ldapsearch**:

```
ldapsearch -x -b " -s base '(objectclass=*)' namingContexts
```

Este comando nos debería devolver el siguiente resultado.

```
dn:
namingContexts: dc=setec,dc=com
```

4.6.3 Primeras entradas del directorio

Mediante el uso del comando **ldapadd** se añaden las primeras entradas. Para ello es necesario crear un archivo en formato LDIF.

Usa tu editor de texto favorito para crear un archivo LDIF (ldif1.ldif) que contenga la información inicial de tu directorio:

```
dn: dc=setec,dc=com
objectclass: dcObject
objectclass: organization
o: Setec Astronomy
dc: setec

dn: cn=root,dc=setec,dc=com
objectclass: organizationalRole
cn: root
```

A continuación usa ldapadd para añadir estas entradas al directorio

```
ldapadd -x -D "cn=root,dc=setec,dc=com" -W -f ldif1.ldif
```

Comprueba si la operación se ha realizado correctamente con el comando

```
ldapsearch -x -b 'dc=setec,dc=com' '(objectclass=*)'
```

Este comando buscará y mostrará todas las entradas en la base de datos que se encuentren en el directorio setec.

4.6.4 Creación de la estructura de árbol

Añadimos la estructura de nuestro árbol de directorio.

Creemos un archivo LDIF con el siguiente contenido:

```
dn: ou=es,dc=setec,dc=com
ou: es
description: Sede en Espana
objectclass: organizationalunit

dn: ou=pt,dc=setec,dc=com
ou: pt
description: Sede en Portugal
objectclass: organizationalunit

dn: ou=us,dc=setec,dc=com
ou: us
description: Sede en USA
objectclass: organizationalunit

dn: ou=mad,ou=es,dc=setec,dc=com
ou: mad
description: Sede en Madrid (Spain)
objectclass: organizationalunit
```

```
dn: ou=bcn,ou=es,dc=setec,dc=com
ou: bcn
description: Sede en Barcelona (Spain)
objectclass: organizationalunit
```

```
dn: ou=lis,ou=pt,dc=setec,dc=com
ou: lis
description: Sede en Lisboa (Portugal)
objectclass: organizationalunit
```

```
dn: ou=ny,ou=us,dc=setec,dc=com
ou: ny
description: Sede en New York (USA)
objectclass: organizationalunit
```

```
dn: ou=sfo,ou=us,dc=setec,dc=com
ou: sfo
description: Sede en San Francisco (USA)
objectclass: organizationalunit
```

```
dn: ou=bos,ou=us,dc=setec,dc=com
ou: bos
description: Sede en Boston (USA)
objectclass: organizationalunit
```

Lo salvamos como Idif2.ldif y lo cargamos en el servidor LDAP con el siguiente comando:

```
ldapadd -x -D "cn=root,dc=setec,dc=com" -W -f Idif2.ldif
```

y comprobamos que la operación se ha realizado correctamente con el comando:

```
ldapsearch -x -b 'dc=setec,dc=com' '(objectclass=*)'
```

4.6.5. Primeras entradas

Una vez creada toda la estructura de directorio vamos a introducir los primeros usuarios.

Para ello volvemos a crear un fichero LDIF con el contenido:

```
dn: uid=jmsuarez,ou=mad,ou=es,dc=setec,dc=com
uid: jmsuarez
cn: Jose Manuel
sn: Manuel
objectclass: top
objectclass: person
objectclass: posixaccount
loginshell: /bin/bash
uidnumber: 99
gidnumber: 99
```

```
homedirectory: /home/jmsuarez
userpassword: secret1
```

```
dn: uid=msilva,ou=lis,ou=pt,dc=setec,dc=com
uid: msilva
cn: Mauro
sn: Silva
objectclass: top
objectclass: person
objectclass: posixaccount
loginshell: /bin/bash
uidnumber: 100
gidnumber: 100
homedirectory: /home/msilva
userpassword: secret2
```

```
dn: uid=jsmith,ou=ny,ou=us,dc=setec,dc=com
uid: jsmith
cn: John
sn: Smith
objectclass: top
objectclass: person
objectclass: posixaccount
loginshell: /bin/bash
uidnumber: 102
gidnumber: 102
homedirectory: /home/jsmith
userpassword: secret3
```

Lo salvamos como `ldif3.ldif` y lo cargamos en el servidor LDAP con el siguiente comando:

```
ldapadd -x -D "cn=root,dc=setec,dc=com" -W -f ldif3.ldif
```

Podemos comprobar la operación con los comandos:

```
ldapsearch -x -b 'dc=setec,dc=com' uid=jmsuarez
ldapsearch -x -b 'dc=setec,dc=com' uid=msilva
ldapsearch -x -b 'dc=setec,dc=com' uid=jsmith
```

4.6.6 Búsquedas

A partir de este momento ya podemos realizar búsquedas en nuestro directorio.

Pongamos algunos **ejemplos**:

- o Esta búsqueda mostrará todos las entradas del directorio a partir de la rama “dc=setec,dc=com”

```
ldapsearch -x -b 'dc=setec,dc=com' '(objectclass=*)'
```

- o Con esta búsqueda obtendremos los datos del usuario jmsuarez

```
ldapsearch -x -b 'dc=setec,dc=com' uid=jmsuarez
```

- o Con esta otra buscaremos al usuario jmsuarez en la rama “ou=mad,ou=es,dc=setec,dc=com”

```
ldapsearch -x -b 'ou=mad,ou=es,dc=setec,dc=com' uid=jmsuarez
```

- o Y con esta otra en la rama “ou=us,dc=setec,dc=com” (no mostrará ningún resultado)

```
ldapsearch -x -b 'ou=us,dc=setec,dc=com' uid=jmsuarez
```

4.6.7 Modificación de atributos

Si queremos realizar una modificación de un atributo de una entrada lo haremos creando un archivo LDIF y usando el comando **ldapmodify**.

Por ejemplo, queremos cambiar el apellido (sn) a la entrada “uid=jmsuarez”, crearemos un archivo LDIF (ldif4.ldif) con el siguiente contenido:

```
dn:uid=jmsuarez,ou=mad,ou=es,dc=setec,dc=com
changetype:modify
replace: sn
sn: San Martin
```

Y a continuación ejecutaremos el comando

```
ldapmodify -x -D "cn=root,dc=setec,dc=com" -W -f ldif4.ldif
```

Podemos comprobar la operación con el comando:

```
ldapsearch -x -b 'dc=setec,dc=com' uid=jmsuarez
```

4.6.8 Borrado de atributos

Si queremos borrar un atributo de una entrada lo haremos creando un archivo LDIF y usando el comando **ldapmodify**.

Por ejemplo, borrar el atributo apellido (sn) de la entrada "uid=jmsuarez", crearemos un archivo LDIF (ldif4.ldif) con el siguiente contenido:

```
dn:uid=jmsuarez,ou=mad,ou=es,dc=setec,dc=com
changetype:delete
delete: sn
```

Y a continuación ejecutaremos el comando

```
ldapmodify -x -D "cn=root,dc=setec,dc=com" -W -f ldif4.ldif
```

Podemos comprobar la operación con el comando:

```
ldapsearch -x -b 'dc=setec,dc=com' uid=jmsuarez
```

4.6.9 Inclusión de atributos

Si queremos añadir un atributo de una entrada lo haremos creando un archivo LDIF y usando el comando **ldapmodify**.

Por ejemplo, añadir el atributo apellido (sn) a la entrada "uid=jmsuarez", crearemos un archivo LDIF (ldif4.ldif) con el siguiente contenido:

```
dn:uid=jmsuarez,ou=mad,ou=es,dc=setec,dc=com
changetype:modify
add: sn
sn: Suarez
```

Y a continuación ejecutaremos el comando

```
ldapmodify -x -D "cn=root,dc=setec,dc=com" -W -f ldif4.ldif
```

Podemos comprobar la operación con el comando:

```
ldapsearch -x -b 'dc=setec,dc=com' uid=jmsuarez
```

Y a continuación ejecutaremos el comando

```
ldapmodify -x -D "cn=root,dc=setec,dc=com" -W -f ldif4.ldif
```

Podemos comprobar la operación con el comando:

```
ldapsearch -x -b 'dc=setec,dc=com' uid=jmsuarez
```

4.6.10 Modificación de un DN

Para modificar el DN de una entrada creamos un archivo LDIF (ldif6.ldif) con el siguiente contenido

```
dn: uid=msilva,ou=lis,ou=pt,dc=setec,dc=com
changetype:modrdn
newrdn: uid=msilva2,ou=lis,ou=pt,dc=setec,dc=com
```

Y a continuación ejecutaremos el comando

```
ldapmodify -x -D "cn=root,dc=setec,dc=com" -W -f ldif6.ldif
```

Podemos comprobar la operación con el comando:

```
ldapsearch -x -b 'dc=setec,dc=com' 'uid=msilva*'
```

4.6.11 Borrado de una entrada

Si queremos borrar una entrada con todos sus atributos usaremos el comando **ldapdelete** y a continuación el DN que queremos eliminar:

```
ldapdelete -x -D "cn=root,dc=setec,dc=com" -W "uid=jmsuarez , ou=mad, ou=es, dc=setec, dc=com"
```

4.7 Índices

Con el objetivo de mejorar el rendimiento de las búsquedas dentro del directorio LDAP, se pueden definir una serie de índices al archivo de configuración del demonio slapd.

En OpenLDAP los índices se ponen por el tipo de consulta que vas a hacer. La sintaxis de los índices es:

```
index lista_att tipo_indice
```

lista_att = lista de atributos separados por coma
tipo_indice = segun el tipo de consulta que vas a hacer:

- **pres:** si la consulta para ese atributo es atributo=algo
- **approx:** si la consulta es atributo~=algo (algo asi como el like de SQL)
- **eq:** la consulta es exacta atributo=algo
- **sub:** la consulta tiene wildcards atributo=al*

Si tienes muchas consultas del tipo cn=nombre te conviene definir el índice como:

```
index cn eq
```

Por ejemplo:

```
index objectClass eq
index cn pres,sub,eq
index sn pres,sub,eq
index uid pres,sub,eq
```

Cada vez que se modifican o añaden los índices hay que regenerarlos con el comando

```
# /usr/sbin/slapindex -vf /etc/ldap/slapd.conf
```

Y reiniciar el servidor OpenLDAP.

4.8 Sintaxis de los filtros de búsqueda

Los filtros de búsqueda te permiten definir criterios de búsqueda y realizar búsquedas más eficientes y efectivas.

Los filtros de búsqueda de LDAP están definidos en el RFC2254.

La siguiente tabla muestra algunos de los filtros de búsqueda:

Filtro de búsqueda	Descripción
(objectClass=*)	All objects.
(&(objectCategory=person)(objectClass=user)(!cn=andy))	All user objects but andy.
(sn=sm*)	All objects with a surname that starts with sm.
(&(objectCategory=person)(objectClass=contact)(!(sn=Smith)(sn=Johnson)))	All contacts with a surname equal to Smith or Johnson.

Estos filtros de búsqueda siguen la sintaxis:

<filtro>=(<atributo><operador><valor>)

O:

(<operador><filtro1><filtro2>)

Operadores

La siguiente tabla muestra los operadores de los filtros de búsqueda más usados:

Operador Lógico	Descripción
=	Igual a
~=	Aproximadamente igual a
<=	Menor o igual que
>=	Mayor o igual que
&	AND
	OR
!	NOT

Wildcards

Se pueden usar 'wildcards' y condiciones a un filtro de búsqueda de LDAP. Los siguientes ejemplos muestran subcadenas que pueden utilizarse para buscar en nuestro directorio:

Obtener todas las entradas:
(objectClass=*)

Obtener las entradas que contienen la cadena "bob" en el nombre (common name o cn):
(cn=*bob*)

Obtener las entradas con un nombre mayor o igual (léxicamente) que "bob":
(cn>='bob')

Obtener todos los usuarios que tienen definido el atributo "email":
(&(objectClass=user)(email=*))

Obtener todas las entradas de usuarios que tengan definido el atributo "email" y con un apellido igual a smith:
(&(sn=smith)(objectClass=user)(email=*))

Obtener todas las entradas con un common name que empiece por andy, steve o margaret:
(&(objectClass=user) | (cn=andy*)(cn=steve)(cn=margaret))

Obtener todas las entradas sin el atributo "email":
(!(email=*))

La definición formal de los filtros de búsqueda (según el RFC1960) es:

```

<filter> ::= '(' <filtercomp> ')'
<filtercomp> ::= <and> | <or> | <not> | <item>
<and> ::= '&' <filterlist>
<or> ::= '|' <filterlist>
<not> ::= '!' <filter>
<filterlist> ::= <filter> | <filter> <filterlist>
<item> ::= <simple> | <present> | <substring>
<simple> ::= <attr> <filtertype> <value>
<filtertype> ::= <equal> | <approx> | <ge> | <le>
<equal> ::= '='
<approx> ::= '~='
<ge> ::= '>='
<le> ::= '<='
<present> ::= <attr> '='
<substring> ::= <attr> '=' <initial> <any> <final>
<initial> ::= NULL | <value>
<any> ::= '*' <starval>
<starval> ::= NULL | <value> '*' <starval>
<final> ::= NULL | <value>

```

Caracteres especiales

Cualquiera de estos caracteres especiales puede aparecer en los filtros de búsqueda reemplazando a los siguientes caracteres ASCII:

Carácter ASCII	Secuencia de escape
*	\2a
(\28
)	\29
\	\5c
NUL	\00
/	\2f

EJERCICIO 4

SETEC ha abierto una nueva sucursal en Londres, añádala al directorio.

EJERCICIO 5

A partir del directorio creado para la empresa SETEC cree un nuevo usuario ficticio en la sede de Lisboa.

EJERCICIO 6

Al usuario que acaba de crear modifíquelo el apellido (campo sn).

EJERCICIO 7

Borre el campo apellido del usuario que acaba de crear

EJERCICIO 8

Borre del directorio al usuario que acaba de crear

5.- Control de accesos

El control de accesos nos permite configurar quien puede acceder a la información y quien puede borrarla o modificarla.

El control de accesos se configura en el archivo slapd.conf, la sintaxis básica es:

```
access to <what>
      [ by <who> <access> [ <control> ] ]+
```

<what>

The entity the access control directive applies to. This statement can have the following forms (this is an incomplete list. See the man page for a complete listing)

*

A wildcard that stands for all the entries.

dn[.<dnstyle>]=<pattern>

An entry based on a naming context such as dn="dc=somedomain,dc=com" which represents the base tree for somedomain.com, or dn="ou=People,dc=somedomain,dc=com" which represents the subtree based at the People organizational Unit.

<dnstyle> is an optional qualifier which defaults to regex . It can also be defined with base or exact, one, subtree, and children. I'll elaborate on <dnstyle> later.

attrs=<attrlist>

A comma separated list of attributes this access control directive applies to. An example would be, attrs=userPassowrd which controls access to an LDAP user's password.

There are also two special entries, entry and children which pertain to this entry and the entries which are part of the subtree based at this entry, respectively.

The last two elements are additive and can be used to control access to both the entry based at the particular naming context and its attributes.

<who>

Indicates to whom the access rules apply. Multiple <who> statements can be applied to an access control clause, thus indicating different access privileges to the same resource. This statement can have the following forms (this is an incomplete list. See the man page for the complete listing)

*

A wildcard entry that defaults to everybody.

anonymous

Access is granted to unauthenticated users, that is those who have not performed a bind with LDAP server. It is important to realize that, with the exception of peername below, everyone is considered anonymous until they've authenticated with the LDAP server.

users

Access is granted to authenticated users.

self

Access is granted to an entry by the entry itself. For example, the user identified by the DN of uid=user,ou=People,dc=somedomain,dc=com will be granted access to the entry rooted at uid=user,ou=People,dc=somedomain,dc=com in the DIT.

dn[.<dnstyle>]=<pattern>

Access is granted to the matching distinguished name indicated by <pattern>. The optional <dnstyle> defaults to regex but can also take the same forms as the <dnstyle> from the <dn> form of the <what> field. I'll elaborate more on this later.

group[.<style>]=<pattern>

Access is granted to entities whose DN is listed in the member attribute of the entry whose DN is matched by <pattern>. In other words, the <pattern> value represents an entry defined by the groupofNames objectClass, and the accessing entry has a distinguished name which matches one of the member attributes of that group. That still might not be very clear, so I'll elaborate on this later.

<style> is an optional value which defaults to regex. It may also take the form of base or exact (which is the same thing as base).

peername[.<style>]=<pattern>

Access is granted to those hosts whose IP address matches the value indicated by <pattern>. Essentially this is an access control list based on IP.

<style> is an optional value which defaults to regex. It may also take the form of base or exact (which is the same thing as base).

There are other possible <who> directives but these are the ones that I use most frequently.

<access>

This statement determines the access level or privileges those specified in the <who> statement will have. It can have the following forms (this is an incomplete list. See the man page for a complete listing)

none

No access is granted to this entry.

auth

Access is granted to this entry's attribute(s) to perform authentication/authorization. This is used for the bind operation. Until a user authenticates she is considered an anonymous user. I'll elaborate on this later.

compare

Access is granted to compare an attribute(s) with this entry's attribute(s). This does not mean you can search for entries using this attribute(s).

search

Access is granted to search on this entry's attribute(s). This does not imply that you can read the other attributes of the entries returned by this search, just that you can search for them.

read

Access is granted to read this entry and it's prescribed list of attributes (defaults to all but can be delimited by the attrs=<attrlist> directive).

write

Access is granted to modify this entry and it's prescribed list of attributes (defaults to all but can be delimited by the `attrs=<attrlist>` directive).

Each access level is incremental in interpretation and implies all previous access levels. For example, search implies compare and auth access, whereas write implies complete access.

<control>

The optional `<control>` qualifier alters the flow of access rule application. It can have the following (documented) forms

stop

This is the default and it causes access checking to stop on a match. Let me repeat that, **all further access checking stops on a match**. This means that, unless you indicate otherwise, the order of your access control directives matters. I'll elaborate on this further.

continue

Continue evaluating the current access control clause after a match. That is, continue on to the next `<who>` directive.

break

On a match, break out of the current access control clause and evaluate the next clause that follows.

Although I already stated it, it's important to understand that the default is to **stop** access control evaluation on the first match. By default, the order of your ACLs matters. Understanding this can save you a lot of pain and suffering later.

Pongamos algunos ejemplos:

```
# ACL Uno
#
access
to attr=userPassword
  by self write
  by anonymous auth
  by * none
```

ACL Uno permite a los usuarios autenticados cambiar el password (by self write), a los no autenticados a autenticarse (by anonymous auth) y por defecto no permite mostrar el password a ningún usuario (by * none).

```
#ACL dos
#
access to dn="*.*,dc=ejemplo,dc=net" attr=userPassword
  by dn="cn=root,dc=ejemplo,dc=net" write
  by self write
  by * auth
```

Esta lista de control de acceso se puede interpretar como: *Para los atributos userPassword de todas las entradas bajo "dc=ejemplo,dc=net", se dará permiso de escritura al usuario administrador, al usuario propietario, y al resto se les permitirá la operación de autenticación.*

```
#ACL Tres
access to dn=".*,dc=ejemplo,dc=net" attr=mail
  by dn="cn=root,dc=ejemplo,dc=net" write
  by self write
  by * read
```

Con esta ACL lo que se hace es proteger el atributo de contraseña para que no pueda ser inspeccionado por cualquiera. Este es un ejemplo similar al anterior, pero se permite la lectura del atributo *mail* (es decir, la dirección de correo electrónico) a cualquiera, mientras que se permite su modificación a la cuenta administrativa y al dueño del atributo.

```
#ACL cuatro
#
access to dn=".*,ou=People,dc=ejemplo,dc=net"
  by * read
```

Si bajo la entrada con DN "ou=People,dc=ejemplo,dc=net" se almacenan las cuentas de usuario del sistema, entonces deberíamos permitir sólo la lectura de estos datos a todo el mundo (sin permitir la modificación de por ejemplo, el nombre de usuario, ni siquiera al propio usuario).

```
#
# ACL Cinco
#
access to *
  by self write
  by users read
  by * none
```

ACL Dos permite a los usuarios autenticados realizar cualquier tipo de cambio en sus entradas (by self write) y leer otras entradas de otros usuarios (by users read). Los usuarios no autenticados no tienen acceso al directorio (by * none).

Un ACL más compleja podría ser:

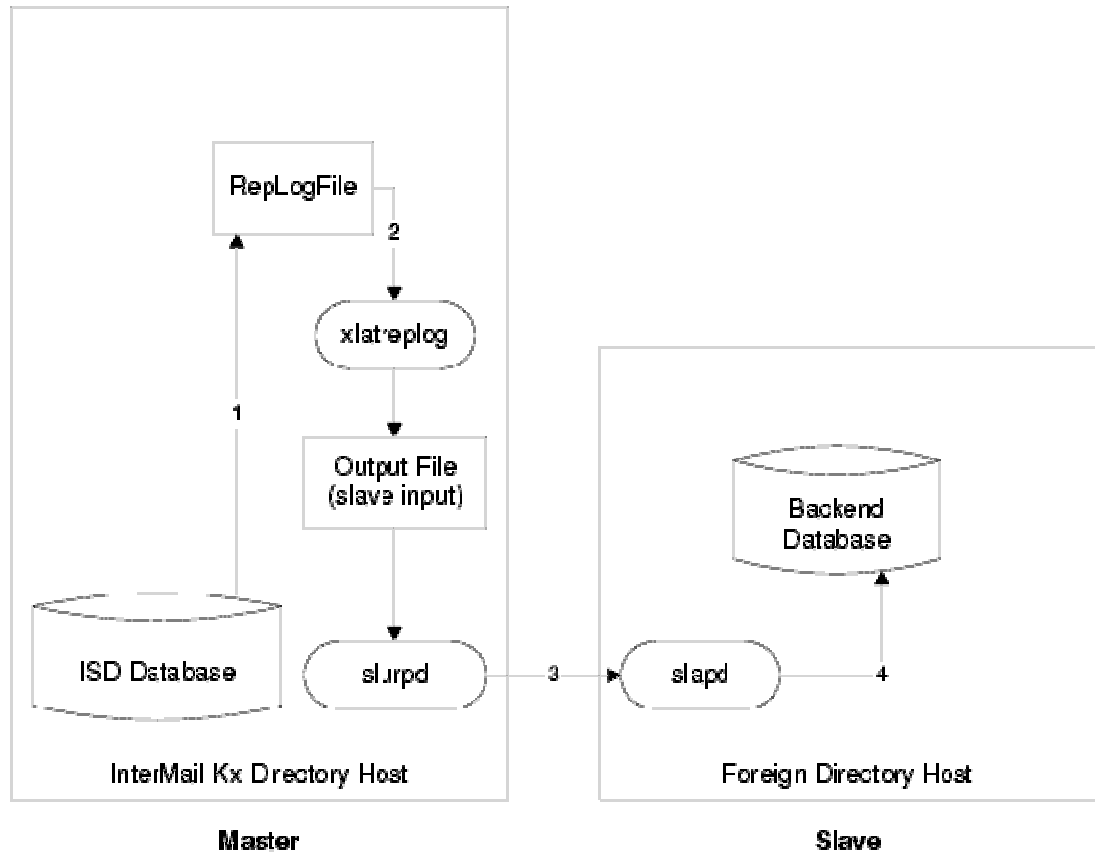
```
# ACL Seis
#
access to attr=userPassword
  by self          write
  by anonymous     auth
  by *             none
access to *
  by self          write
  by users         read
  by peername=127.0.0.1 read
  by peername=10.0.11.* read
  by *             none
```

6. Replicación de directorios

Para replicar un directorio con OpenLDAP debemos elegir un servidor master y uno o más servidores slave.

En el servidor master se configura el 'demonio' **slurpd** que se encarga de las replications.

El esquema de replicación es el siguiente:



En el servidor maestro tenemos que añadir las siguientes líneas al archivo `slapd.conf`

```
repllogfile /opt/openldap/var/openldap-slurp/replica/slurpd.repllog
replica host= slave1.empresa.com
bindmethod= simple
binddn= "cn=root,dc=setec,dc=com"
credentials= secret
```

Si queremos añadir un segundo servidor slave añadiremos además las líneas:

```
replica host= slave2.empresa.com
bindmethod= simple
```

```
binddn= "cn=root,dc=setec,dc=com"  
credentials= secret
```

En cada servidor esclavo solo tendremos que añadir las siguientes líneas en el archivo slapd.conf:

```
updateref ldap://master.empresa.com:389  
updatedn "cn=root,dc=setec,dc=com"
```

A partir de este momento cada cambio que se realice en el servidor master será replicado a todos los servidores slave.

7. Backup y disaster recovery de BBDD en LDAP y OPENLDAP

Desgraciadamente OpenLDAP no se integra muy bien con las aplicaciones comerciales de backup. Por suerte el sistema de réplica nos permite tener varios servidores que son accedidos desde las aplicaciones, de tal forma que si falla uno siempre podemos tener otro servidor con todos los datos.

No se puede realizar backup “en caliente” de un servidor OpenLDAP, el método de backup más común es programar un **script** que a determinada hora:

- Para el servidor de LDAP (slapd y slurp)
- Hace una copia de la base de datos y la configuración
- Comprime la copia y la deja en un punto accesible por nuestro programa de backup
- Arranca el servidor y comprueba que todo funciona correctamente.

Para recuperar un servidor basta con:

- Instalar la misma versión de OpenLDAP con la que estamos trabajando.
- Copiar la base de datos y el archivo de configuración
- Arrancar el servidor

Ejemplo de un script de backup diario para Linux:

```
#!/bin/sh

FECHA_ACTUAL=`date +%d-%m-%Y`

# Para el servidor OpenLDAP
/etc/init.d/ldap stop

# Crea archivos comprimidos con las bases de datos y la configuración
tar -czvf /opt/backup/var_lib_ldap-$FECHA_ACTUAL.tgz /var/lib/ldap
tar -czvf /opt/backup/etc_openldap-$FECHA_ACTUAL.tgz /etc/openldap

# Borra las copias almacenadas más de 15 días
find /opt/backup -type f -name "*.tgz" -ctime +15 -exec rm -f {} \;

# Inicia el servidor OpenLDAP
/etc/init.d/ldap start
```

8. Monitorización

Para monitorizar el funcionamiento correcto de un servidor OpenLDAP hay que controlar los siguientes parámetros:

- del sistema operativo:
 - nivel de ocupación del procesador
 - nivel de ocupación de los discos duros
 - nivel de paginación de la memoria RAM
 - número de procesos del servidor
 - los procesos slapd y slurpd están corriendo
- del servidor OpenLDAP
 - un proceso remoto que realice consultas al servidor (Nagios, Tivoli, OpenView...) que nos avise si slapd deja de responder. Debemos fijar un tiempo respuesta en la consulta, por ejemplo 5 segundos.
 - Un proceso local que compruebe el tamaño de slapd en memoria y que nos avise si excede el tamaño que le hemos fijado.
 - Un proceso local que comprueba cada 5 minutos si slapd está corriendo, avisará al administrador de sistemas si falla y reiniciará slapd si no está corriendo.

9. Integración con lenguajes de programación

9.1 PHP

El soporte LDAP no está activado por defecto en LDAP. Necesitas compilarlo con la opción **--with-ldap[=DIR]**. Donde DIR es el directorio de instalación de OpenLDAP

Un ejemplo de búsqueda

```
<?php
// basic sequence with LDAP is connect, bind, search, interpret search
// result, close connection

echo "<h3>LDAP query test</h3>";
echo "Connecting ...";
$ds=ldap_connect("localhost"); // must be a valid LDAP server!
echo "connect result is " . $ds . "<br />";

if ($ds) {
    echo "Binding ...";
    $r=ldap_bind($ds); // this is an "anonymous" bind, typically
    // read-only access
    echo "Bind result is " . $r . "<br />";
}
```

```

echo "Searching for (sn=S*) ...";
// Search surname entry
$sr=ldap_search($ds, "o=My Company, c=US", "sn=S*");
echo "Search result is " . $sr . "<br />";

echo "Number of entires returned is " . ldap_count_entries($ds, $sr) . "<br />";

echo "Getting entries ...<p>";
$info = ldap_get_entries($ds, $sr);
echo "Data for " . $info["count"] . " items returned:<p>";

for ($i=0; $i<$info["count"]; $i++) {
    echo "dn is: " . $info[$i]["dn"] . "<br />";
    echo "first cn entry is: " . $info[$i]["cn"][0] . "<br />";
    echo "first email entry is: " . $info[$i]["mail"][0] . "<br /><hr />";
}

echo "Closing connection";
ldap_close($ds);

} else {
    echo "<h4>Unable to connect to LDAP server</h4>";
}
?>

```

Más información en <http://es.php.net/ldap>

9.2 Perl

Para conectar a servidores LDAP desde scripts programados en PERL necesitaremos unos módulos que se pueden descargar desde

<http://ldap.perl.org/>

Un ejemplo de búsqueda

Un ejemplo de búsqueda

```

#!/usr/bin/perl

use Net::LDAP;

$ldap = Net::LDAP->new("localhost");

$ldap->bind("cn=es,dc=setec,dc=com", password=>"secret");

$msg = $ldap->search(filter=>"(objectClass=*)", base=>"dc=setec,dc=com");

@entries = $msg->entries;

foreach $entry (@entries) {
    $entry->dump;
}

```

9.3 Python

Primero necesitamos instalar un módulo para Python llamado **python-ldap** que está disponible en

<http://python-ldap.sourceforge.net/>

Un ejemplo de búsqueda

```
import ldap
try:
    l = ldap.open("127.0.0.1")

    # you should set this to ldap.VERSION2 if you're using a v2 directory
    l.protocol_version = ldap.VERSION3

    # Pass in a valid username and password to get
    # privileged directory access.
    # If you leave them as empty strings or pass an invalid value
    # you will still bind to the server but with limited privileges.

    username = "cn=Manager, o=anydomain.com"
    password = "secret"

    # Any errors will throw an ldap.LDAPError exception
    # or related exception so you can ignore the result
    l.simple_bind(username, password)
except ldap.LDAPError, e:
    print e

# handle error however you like
```

10. Pruebas de calidad y carga

Antes de poner cualquier servicio en producción realizaremos una serie de pruebas de calidad sobre nuestro servidor, donde mediremos el tiempo de respuesta para diferentes niveles de carga.

La herramienta *SilkPerformer* de Segue realiza pruebas completas de carga que nos ayudarán a simular los niveles de carga adecuados así como a realizar un dimensionamiento correcto de nuestros servidores y a conocer sus límites.

Debido al elevado precio de esta herramienta hay empresas que realizan los servicios de test de software (en España por ejemplo inQALabs).

Referencias:

SilkPerformer

<http://www.seguc.com/products/load-stress-performance-testing/silkperformer.asp>

in.Qua Labs

<http://www.inqalabs.es>

11. Optimizando el rendimiento de OpenLDAP

Conseguiremos que nuestro software esté mucho más optimizado si desactivamos antes de compilar las opciones que no vamos a usar y activamos otras específicas, por ejemplo.

```
./configure \  
--disable-debug \  
--disable-ipv6 \  
--enable-crypt \  
--without-tls \  
--with-threads
```

Esto le prepara a OpenLDAP para configurarse de una forma especial:

- Sin soporte para debugging.
- Sin soporte para IPv6.
- Activa el soporte para passwords cifrados (con crypt).
- Desactiva la encriptación TLS/SSL.
- Activa el soporte para threads.

Y a continuación:

```
[root@dep openldap-2.0.25]# make depend
[root@dep openldap-2.0.25]# make
[root@dep openldap-2.0.25]# cd tests/
[root@dep tests]# make test
[root@dep tests]# cd
[root@dep /root]# find /* > OpenLDAP1
[root@dep /root]# cd /var/tmp/openldap-2.0.25/
[root@dep openldap-2.0.25]# make install
[root@dep openldap-2.0.25]# install -d -m 700 /var/lib/ldap
[root@dep openldap-2.0.25]# rm -rf /var/run/openldap-ldbm
[root@dep openldap-2.0.25]# rm -f /etc/openldap/*.default
[root@dep openldap-2.0.25]# rm -f /etc/openldap/schema/*.default
[root@dep openldap-2.0.25]# strip /usr/lib/liblber.a
[root@dep openldap-2.0.25]# strip /usr/lib/liblber.so.2.0.5
[root@dep openldap-2.0.25]# strip /usr/lib/libldap.a
[root@dep openldap-2.0.25]# strip /usr/lib/libldap.so.2.0.5
[root@dep openldap-2.0.25]# strip /usr/lib/libldap_r.a
[root@dep openldap-2.0.25]# strip /usr/lib/libldap_r.so.2.0.5
[root@dep openldap-2.0.25]# /sbin/ldconfig
[root@dep openldap-2.0.25]# cd
[root@dep /root]# find /* > OpenLDAP2
[root@dep /root]# diff OpenLDAP1 OpenLDAP2 > OpenLDAP-Installed
```

El comando **strip** descartará todos los símbolos de los archivos seleccionados. Esto significa que el tamaño en memoria de las librerías será inferior y por tanto mejorará su rendimiento cada vez que se ejecuten debido que disponen de menos líneas para ser leídas por el sistema cada vez que las usa.

Más opciones de optimización:

- **Usa sólo los atributos index por los que vas a realizar las búsquedas**
Cada índice requiere tiempo de mantenimiento y usa memoria adicional, cuantos menos índices uses menos tiempo de mantenimiento será necesario y dispondrás de más memoria libre.
- En un servidor OpenLDAP con mucha carga LDAP, necesitarás añadir la siguiente línea en el archivo slapd.conf

loglevel 0

Esta línea indica que no se use ciclos extra de CPU en generar archivos de registro a través del demonio syslog.

Esta ganancia de prestaciones es muy apreciable pero significa perder control sobre las acciones que realiza la aplicación. Está tu mano está el usarla o no.

- Puedes mejorar el rendimiento del proceso de registro de eventos en algunos sistemas que usan syslog (por ejemplo Linux) configurando dicho demonio para no sincronizar el archivo de registros en cada escritura (consulta la ayuda man syslog.conf)

En Linux puedes realizar esta operación anteponiendo un guión “-“ en el archivo de registro en syslog.conf, por ejemplo

```
# LDAP logs
local4.*                -/var/log/ldap
```

- Puedes optimizar la memoria cache usando las directivas cachesize y dbcachesize del archivo de configuración slapd.conf.

Por ejemplo:

```
cachesize 1000000
dbcachesize 10000000
```

Permitirá almacenar en la memoria cache de 1.000.000 de registros y reservará hasta 100MB para cada archivo de índices en uso.

- Slapd sincroniza por defecto la base de datos de backend en cada modificación. Esta forma de actuar es segura pero lenta. En situaciones donde no es necesaria esta protección (como por ejemplo un servidor slave) puedes deshabilitar la sincronización de escrituras incluyendo la siguiente directiva en slapd.conf

```
dbcacheNoWsync
```

Berkeley DB vs LDBM

Para cualquier instalación es recomendable utilizar Sleepycat (BerkeleyDB).

Al resto de bases de datos cada vez se le da menos soporte, y las

ventajas de transaccionalidad, cache y recuperación en caso que se corrompa hacen que Sleepycat sea la elección más adecuada.

Referencias:

<http://www.openldap.org/faq/data/cache/756.html>

12. Seguridad avanzada

Para verificar la seguridad de nuestro servidor OpenLDAP usaremos herramientas de análisis de vulnerabilidades de forma periódica.

Las dos herramientas más conocidas son:

- nessus: Herramienta OpenSource de análisis de vulnerabilidades.
- e-Eye Retina: herramienta comercial para realizar auditorías de seguridad

Estas herramientas nos ayudarán a corregir fallos de configuración que puedan afectar a nuestros servidores OpenLDAP o problemas derivados del propio software.

Además todo administrador del sistema deberá estar al día sobre los problemas de seguridad que aparecen en OpenLDAP. Este seguimiento puede hacerse a través de listas de correo de seguridad o en la propia lista de correo de OpenLDAP.

13. Interfaces gráficos

PHPLDAPADMIN

<http://phpldapadmin.sourceforge.net/>

Interfaces gráficos

<http://www.ldapguru.net/modules/mydownloads/viewcat.php?cid=8&min=20&orderby=titleA&show=10>

14. Sobre el autor y agradecimientos

Jose Manuel Suárez es consultor de sistemas informáticos y comunicaciones desde hace casi 10 años. Ha trabajado como responsable de los departamentos de sistemas informáticos y consultoría de Montejava, Motorpress, Teknoland, Ya.com y Betybyte. Actualmente trabaja en Goa Sistemas como director técnico.

Su primera experiencia con LDAP y OpenLDAP se remonta al año 1998 cuando empezó a implementar un backend para los servidores Web de Grupo Z basado en LDAP. Posteriormente implantó el sistema de backend del sistema de correo electrónico de los usuarios y clientes de Ya.com y Mixmail. También ha realizado diversas implantaciones de LDAP para CocaCola, Grupo Z y Terra.

En el año 2002 colaboró con Lighthbrige Inc. en Silicon Valley (USA) en el desarrollo de una infraestructura de back-end basada en LDAP (Oracle y OpenLDAP) para almacenar los datos necesarios para realizar una pasarela del MSN Messenger de Microsoft con las principales redes de telefonía inalámbrica de USA.

Como profesor ha impartido varios cursos de LDAP y OpenLDAP para grandes empresas.

El autor reconoce que este manual no habría sido posible gracias a las siguientes páginas de donde se han recogido algunos textos e incluso imágenes.

An introduction to OpenLDAP

http://ldapman.org/articles/intro_to_ldap.html

Introduction to LDAP

<http://twistedmatrix.com/users/tv/ldap-intro/ldap-intro.html>

Directory Project: OpenLDAP Upgrade

<http://www.stanford.edu/services/directory/openldap/history/choice.html>

LDAP

<http://www.gfc.edu.co/estudiantes/anuario/2001/sistemas/andrea/ldap.html>

LDAP Filters

<http://www.winnetmag.com/Article/ArticleID/38949/38949.html>

LDAP Implementation HOWTO

<http://www.faqs.org/docs/Linux-HOWTO/LDAP-Implementation-HOWTO.htm>