Introducción al Servicio de Directorio

Rafael Calzada Pradas

Índice

1 Servicio de Directorio

En este capítulo se presentan los conceptos básicos del Servicio de Directorio, cuales son sus características principales, cuales son sus principales aplicaciones y cuales son las diferencias con otras aplicaciones similares.

Tras esta introducción, se presentará el protocolo LDAP, se expondrán los diferentes modelos desde los que se puede estudiar el Directorio basado en LDAP y se explicará el formato LDIF, utilizado para intercambiar datos del Servicio de Directorio.

1.1 ¿Qué es el Servicio de Directorio?

El Servicio de Directorio o simplemente el Directorio es un término ambiguo, que se utiliza para referirse tanto a la información contenida, el conjunto hardware/software que gestiona dicha información, las aplicaciones cliente/servidor que utilizan esta información, etc. La conclusión que se extrae de esta situación, es que el Servicio de Directorio es un conjunto complejo de componentes que trabajan de forma cooperativa para prestar un servicio.

Todo el mundo ha utilizado alguna vez algún tipo de Directorio, desde la *Guía de teléfonos* hasta cualquier revista que contenga la programación televisiva. Utilizando estos ejemplos de la vida diaria vamos a presentar el Directorio, y sus principales características.

Los directorios permiten localizar información, para ello definen que información se almacenará y en que modo se organizará. Sin embargo, los directorios electrónicos difieren de estos directorios *clásicos* (que utilizan el papel como medio de transmisión).

1.1.1 El Directorio es Dinámico.

Uno de los principales problemas de los directorios clásicos se encuentra en que son estáticos, esto es, la información que contienen no es actualizada frecuentemente, por ejemplo, la guía telefónica suele editarse anualmente, esto implica que para conocer el número de teléfono de una persona que ha contratado el servicio telefónico después de la edición en curso, deberemos llamar a la compañía telefónica que un operador consulte la base de datos y nos proporcione el número buscado.

Otros directorios se actualizan con más frecuencia, por ejemplo la programación televisiva suele presentarse semanalmente, pero el desarrollo de los acontecimientos puede hacer que sufra

modificaciones, que generalmente advertimos cuando descubrimos que al programar el vídeo, en lugar de grabar la película prevista hemos grabado una edición especial de algún *reality-show*.

Estos problemas son la consecuencia del esquema y los costes de actualización de estos directorios, cuando una persona cambia de teléfono, la compañía telefónica incluye sus datos para la próxima edición, ya que editar y repartir las guías telefónicas cada vez que se produce un cambio es inviable. Al igual, las editoras de las revistas de programación televisiva deberían preguntar constantemente a las cadenas de televisión si se ha producido algún cambio en la programación y en caso afirmativo, deberían editar y distribuir de nuevo la revista y nosotros deberíamos ir al quiosco de la esquina por si ha salido una nueva edición…

Los directorios electrónicos pueden ser consultados/actualizados en tiempo real y su fiabilidad es por lo tanto mucho mayor.

1.1.2 El Directorio es Flexible.

La flexibilidad del Directorio electrónico se puede contemplar desde dos aspectos:

1.1.2.1 Contenido

Los datos almacenados en el Directorio son cualquier tipo de información que pueda se almacenada en un fichero. Esto permite ampliar la información almacenada sin muchas repercusiones, por ejemplo, podemos incluir la foto de la persona junto con su número de teléfono. En un directorio clásico es muy costoso hacer este tipo de cambios, además el volumen de la guía de teléfonos haría imposible hacer una búsqueda en un tiempo razonable.

1.1.2.2 Organización

Generalmente los directorios clásicos están organizados para realizar búsquedas de un determinado modo, por ejemplo, la guía de teléfonos nos permite buscar el número de teléfono de una persona, pero encontrar a quien corresponde el número de teléfono que tenemos apuntado en el *Post-it* del monitor puede ser, cuando menos, una tarea difícil.

Las compañías telefónicas intentaron afrontar este problema editando otras guías, de tirada mucho menor, que permitían realizar este tipo de búsquedas, pero su alto coste les ha obligado a dejar de editarlas.

La ventaja de los directorios electrónicos es que la organización de la información permite localizarla de diferentes maneras, incluso puede realizar búsquedas aproximadas, algo que es imposible con los directorios clásicos.

1.1.3 El Directorio puede ser Seguro.

Una desventaja de los directorios clásicos es que no puede controlarse el acceso, cualquier persona que tenga acceso físico a la guía tiene acceso a toda la información contenida en ella.

Los datos de un abonado pueden estar accesibles por cualquier persona (empresas de telemarketing, etc) o inaccesibles a cualquier persona (incluyendo personas que puede que nos interesen que conozcan nuestro número de teléfono). Claramente ninguna de las soluciones es la óptima. El problema se encuentra en la forma en que se distribuyen estas guías, cualquier persona tiene acceso a ellas y por lo tanto a toda la información contenida en ellas...

Con directorios electrónicos, puede ser controlado el acceso a los datos en función de diferentes criterios, por ejemplo que los datos domiciliarios solo sean accesibles por los vecinos del mismo bloque, etc. Aunque este control no es la solución (ya que cualquier persona autorizada puede imprimir los datos y entregar la copia impresa a quien ella desee), siempre permite un nivel de seguridad superior a los directorios clásicos.

1.1.4 El Directorio es configurable.

Otra desventaja de los directorios tradicionales se encuentra en que su contenido es genérico, por ejemplo, la revista con la programación televisiva, contiene la programación de las cadenas de ámbito nacional o autonómico. Algunas contienen las programaciones de las cadenas via-satélite, pero, o lo hacen con referencias muy escuetas o el tamaño de la publicación la hace inmanejable.

Los directorios electrónicos, por el contrario, permiten la personalización de los datos que se muestran a los distintos usuarios, por ejemplo, un alumno puede ver las calificaciones que ha obtenido en los exámenes, pero no puede ver las de sus compañeros de clase. Sin embargo, puede ser interesante que un profesor pueda consultar las notas de los alumnos que han cursado la asignatura que ha impartido.

A modo de resumen, en los directorios electrónicos se puede establecer la información que recibe una persona en función de sus necesidades y qué personas pueden acceder a dicha información.

1.1.5 Descripción del Directorio

Una vez que hemos presentado el concepto intuitivo de directorio, vamos a presentar las características técnicas que tienen los directorios.

Un directorio puede verse como una base de datos especializada, las diferencias entre una base de datos de propósito general y un directorio son las siguientes:

- ✓ Relación entre lecturas y escrituras.
- ✓ Extensibilidad.
- ✓ Distribución de los datos.
- ✓ Replicación de los datos.
- ✓ Rendimiento
- ✓ Estándares.

1.1.5.1 Relación entre lecturas y escrituras

En un directorio se espera un número muy alto de lecturas frente a escrituras, esto se debe a que generalmente la información contenida en el directorio cambia raramente, por ejemplo cuántas veces cambiamos el número de teléfono y cuántas veces alguien busca nuestro teléfono en el directorio para llamarnos...

Este es un aspecto importante, ya que mientras que en una base de datos de propóstio general, las optimizaciones se realizan tanto en las lecturas como en las escrituras, al crear un directorio, los esfuerzos de optimización se concentran en las búsquedas y lecturas, mientras que no importa que por ello se penalicen las actualizaciones.

1.1.5.2 Extensibilidad

El término *directory schema* se refiere a los tipos de información que se almacenan en el directorio, qué reglas debe cumplir dicha información y cómo se realizan las operaciones de búsqueda sobre estos datos.

La ventaja que presentan los directorios frente a las bases de datos tradicionales estriba en que dicho esquema de puede modificar para cubrir las necesidades que vayan surgiendo en la organización. Esta característica no suele encontrarse en las bases de datos de propósito general.

1.1.5.3 Distribución de los datos

Algunas bases de datos de propósito general permiten la distribución de los datos, pero generalmente esta distribución de datos permite únicamente almacenar una tabla en un servidor y otra en otro servidor distinto (fragmentación vertical), y la distribución de la información implica protocolos más complejos para la realización de actualizaciones, por lo que generalmente no suelen utilizarse.

Los directorios permiten que los datos referentes a toda una unidad organizativa sean almacenados en un servidor controlado por esta unidad (fragmentación horizontal). Este tipo de fragmentación simplifica las actualizaciones, ya que todos los datos referentes a una persona se encuentran en el mismo servidor y permite a su vez optimizar las búsquedas, ya que las consultas se pueden ejecutar en paralelo.

1.1.5.4 Replicación de la información

Las bases de datos de propósito general que admiten replicación de datos, están preparadas para replicar los datos en un número reducido de servidores, esto se debe a que las copias deben ser consistentes y por lo tanto, las actualizaciones deben realizarse de forma sincronizada entre las diferentes sedes.

En el caso de los directorios, es aceptable una inconsistencia temporal, por lo que el protocolo de replicación/actualización es menos restrictivo.

Inherente a la replicación de la información, se encuentra el aumento en la fiabilidad del sistema, ya que en caso de catástrofe, se puede utilizar el servidor replicado. Además también se puede obtener una mejora en el rendimiento al situar las replicas en redes *cercanas* a los usuarios, optimizando el camino de acceso al directorio y repartiendo la carga entre las distintas replicas.

La fiabilidad del directorio comienza a ser crítica en el momento en el que varias aplicaciones lo utilizan para tareas como autenticación , control de accesos y gestión de configuración.

1.1.5.5 Rendimiento

Las necesidades de rendimiento de un directorio frente a una base de datos de propósito general son considerablemente diferentes. Se espera que un servidor de base de datos permita hasta cientos de transacciones por segundo, mientras el rendimiento agregado del directorio se espera que sea del orden de miles de consultas por segundo.

Estos requerimientos de rendimiento se deben a que el directorio forma parte del núcleo de muchas aplicaciones, y por lo tanto debe estar preparado para responder a las múltiples consultas que estas aplicaciones pueden solicitarle.

1.1.5.6 Estándares

El hecho de que las bases de datos de propósito general utilicen ligeras variantes del estándar SQL no suele ser un problema, ya que rara vez tienen que interactuar dos bases de datos de diferentes fabricantes, sin embargo, dado que el directorio es una base de datos accesible desde múltiples aplicaciones, el estricto cumplimiento del estándar es un requisito indispensable.

Este aspecto es importante, ya que permite separar el desarrollo del cliente del desarrollo del servidor, permitiendo que cada desarrollo este optimizado en el sentido que sea conveniente.

Como valor añadido, el hecho de estar sujeto a un estándar permite que el administrador no esté restringido a un único fabricante, pudiendo cambiar de proveedor en el momento que lo considere conveniente, sin tener que cambiar el software que utilizan los clientes.

1.2 ¿Para qué puede utilizarse el Directorio?

Hasta ahora se han expuesto algunas de las aplicaciones de los directorios tradicionales y en qué medida pueden beneficiarse con la implantación de directorios electrónicos. Ahora ha llegado el momento de ver cuáles pueden ser las aplicaciones que pueden desarrollarse utilizando las características especiales de los directorios electrónicos.

1.2.1 Encontrar información.

Una de las principales utilidades de los directorios ha sido la de buscar información, de hecho, el prototipo de directorio siempre ha sido la guía de teléfonos, en la cual los abonados se encuentran ordenados alfabéticamente. La ventaja de los directorios electrónicos está en que permiten una

escalabilidad no disponible en los directorios tradicionales, basta imaginarse el espacio necesario para almacenar la guía de abonados a la compañía telefónica para comprender dicha escalabilidad.

Además, el hecho de ser directorios electrónicos permite acceder a la información contenida en ellos de maneras distintas a las tradicionales. Por ejemplo, se pueden realizar búsquedas por apellido, por dirección, teléfono, etc.

En el momento de acceder al directorio es conveniente diferenciar entre la operación de búsqueda y la operación de hojear. Ambas operaciones son complementarias, ya que por ejemplo, podemos conocer perfectamente el nombre del restaurante en el que deseamos reservar una mesa y otras simplemente sabemos que deseamos salir a cenar y preferimos hojear la *Guía del Ocio* para decidir que tipo de resturante vamos a elegir.

A veces las dos operaciones trabajan de forma complementaria, ya que cuando decidimos comprar ropa, solemos tener claro que tipo de prenda nos vamos a comprar, pero luego nos gusta que el dependiente nos muestre las distintas prendas de las que dispone.

1.2.2 Gestionar información.

A veces no basta con tener la información almacenada en un directorio electrónico, es muy importante que dicho directorio sea accesible desde todas las aplicaciones que son susceptibles de utilizarlo.

Cuando solo una aplicación accede a los datos, quizás el esfuerzo necesario para implantar un servicio de directorio electrónico estándar pueda ser innecesario, pero la experiencia demuestra que tarde o temprano varias aplicaciones utilizan esos datos y en el caso de no haber implantado un directorio centralizado nos encontramos con varios directorios que deben estar sincronizados y que acceden de maneras diferentes a los datos contenidos en directorios creados a medida. Esto implica un mayor esfuerzo para realizar el mantenimiento y un freno al desarrollo de nuevas aplicaciones basadas en el directorio.

Un caso muy común es el de los usuarios itinerantes (*roaming users*) o los ordenadores compartidos. Para solucionar problemas de este tipo, en 1997 se desarrollo la RFC 2247, que solo ha alcanzado el estado de *Proposed standard*, que intentaba desarrollar un protocolo que permitiese a las aplicaciones almacenar sus parámetros de configuración en un servidor, de modo que el

usuario pudiese ejecutar dicha aplicación desde múltiples ordenadores sin necesidad de configurar la aplicación.

Actualmente algunas aplicaciones utilizan el servicio de directorio para acceder a información necesaria para su funcionamiento, por ejemplo el programa *Netscape Communicator*, permite almacenar la configuración del programa, los enlaces almacenados, e incluso las claves privadas correspondientes al los certificados electrónicos emitidos a favor del usuario. Además, el interfaz de correo electrónico permite utilizar el servicio de directorio para realizar consultas y acceder a las direcciones de correo de los destinatarios de los mensajes.

Otro ejemplo, en el que se suele comenzar con un directorio propietario es el de los servidores Web que requieren autenticación, si nuestro sistema solo consta de un único servidor Web, la solución más sencilla pasa por crear la base de datos de usuarios en el servidor y realizar allí las actualizaciones pertinentes. Pero cuando deseamos que varios servidores accedan a dicha base de datos de usuarios, el trabajo empieza a complicarse, teniendo que implementar los mecanismos de sincronización entre los distintos servidores, y lo que puede resultar más desalentador, todo el esfuerzo realizado solo puede ser utilizado en servidores Web, si en el futuro deseamos dar algún servicio a nuestros usuarios deberemos adaptar nuestra aplicación al esquema anterior.

1.2.3 Aplicaciones de Seguridad.

El servicio de directorio es el soporte ideal para la distribución de los certificados electrónicos personales, en concreto, el directorio resuelve dos problemas principales:

- La gestión de la infraestructura de clave pública:
  - ✓ Creación: Ya que permite incorporar al certificado los datos contenidos en el servidor LDAP.
  - ✓ Distribución: Ya que permite tener accesibles mediante un protocolo estándar los certificados electrónicos.
  - ✓ Destrucción: Ya que permite implementar la revocación de un certificado con la simple operación de borrado del certificado del servidor LDAP.

- El problema de la ubicación de los certificados. El directorio es el lugar natural donde los usuarios pueden acceder a los certificados de los restantes usuarios, de una manera cómoda y fácil de integrar con las restantes aplicaciones.

1.3 ¿Qué no es el Directorio?

Antes de avanzar en la descripción de las virtudes de los directorios, es conveniente aclarar las diferencias entre el directorio y otros programas y/o servicios.

1.3.1 Directorio vs Bases de datos.

Generalmente se describe un directorio como una base de datos, pero es una base de datos especializada cuyas características la apartan de una base de datos relacional de propósito general. Una de estas características especiales es que son accedidas (búsqueda o lectura) mucho más que actualizadas (escritura). Muchos usuarios pueden estar consultando el número de teléfono de una persona, o buscando una estación de trabajo con un programa concreto, pero generalmente tanto el número de teléfono de la persona, como los programas instalados en una estación no cambian con excesiva frecuencia. Por ello:

- Los directorios están optimizados para accesos en lectura, frente a las bases de datos convencionales, que se encuentran optimizadas para lectura y escritura.

- Los directorios están optimizados para almacenar información relativamente estática, por lo que no son recomendables para almacenar datos que cambian con frecuencia como por ejemplo la carga de una estación de trabajo.

- Los directorios no soportan transacciones. Las transacciones son operaciones de base de datos que permiten controlar la ejecución de una operación compleja, de modo que dicha operación se completa totalmente o no se ejecuta en absoluto. Las bases de datos convencionales implementan esta funcionalidad, a costa de hacer su implementación más compleja. Pero el tipo de información que se almacena generalmente en el directorio no requiere una consistencia estricta y se considera aceptable que el número de teléfono de una persona no este actualizado de forma temporal.

- La mayoría de las bases de datos convencionales utilizan el lenguaje de consulta SQL, que permite el desarrollo de funciones de consulta y actualización muy complejas, a costa del tamaño y complejidad de la aplicación. Por otra parte, los directorios LDAP utilizan un

protocolo simplificado y optimizado que puede ser utilizado para la construcción de aplicaciones simples y pequeñas.

- Dado que generalmente los directorios son utilizados para consulta, en un entorno no transaccional, tanto el cliente como el servidor pueden ser optimizados y simplificados en esa dirección.

1.3.2 Directorio vs Sistemas de ficheros.

Los directorios están optimizados para almacenar pequeños fragmentos de información que puede estructurarse como entradas con diferentes atributos, en cambio, los sistemas de ficheros contienen archivos, a veces de tamaños superiores al gigabyte. Además, los sistemas de ficheros permiten acceder a un fichero y posicionarse dentro de él, sin embargo, los directorios a lo sumo permiten acceder a un atributo, pero no hay forma de posicionarse dentro de dicho atributo, que por lo tanto debe ser leído por completo.

1.3.3 Directorio vs Web.

Desde que apareció en escena el servicio de Web, se han desarrollado multitud de aplicaciones sobre este protocolo. Pero el servicio de Web está centrado en proporcionar un interfaz de usuario agradable, en ningún momento posee las capacidades de búsqueda que posee el servicio de directorio.

Si deseamos hacer accesible a los usuarios los contenidos de una base de datos, quizá el servicio Web sea la mejor elección, pero si deseamos que estos datos estén accesibles para una gran variedad de aplicaciones, el servicio de directorio es el adecuado.

1.3.4 Directorio vs DNS.

El servicio DNS se encarga de la traducción de nombres de dominio a direcciones IP y viceversa. Tiene además una ligera similitud con el servicio de directorio, ya que ambos proporcionan un interfaz de acceso a una base de datos jerárquica. Pero difieren en otros aspectos:

- El servicio DNS esta optimizado para realizar su cometido, es decir, la traslación de nombres de ordenadores a direcciones IP, mientras que los servidores de directorio están optimizados de forma más general.

- La información almacenada en el servicio DNS tiene una estructura fija, mientras que el servicio de directorio suele permitir la extensión de dicha estructura.

- El servicio de directorio permite actualizaciones, mientras que el servicio DNS no las permite.

- El servicio DNS opera con protocolos no orientados a conexión (UDP), mientras que los servicios de directorio suelen utilizar protocolos orientados a conexión.

Aún cuando hay varios trabajos del IETF para acomodar el servicio de directorio de modo que trabaje igual que el servicio DNS, entre los administradores del servicio DNS prevalece la máxima "*If it works do not fix it*"[1]

1.3.5 Integración del Directorio con otros servicios.

De lo expuesto anteriormente puede deducirse que el servicio de directorio es importante en sí mismo, pero lo es más debido a que puede ser el elemento aglutinador y la herramienta que faltaba para desarrollar aplicaciones que permitan desplegar nuevos servicios basados en la cooperación entre las distintas aplicaciones y el servicio de directorio.

- Primeramente el servicio de directorio puede actuar como servidor de autenticación, proporcionando el servicio de contraseña única. Además puede contener información necesaria para que los distintos servidores puedan decidir si un usuario puede acceder a determinada información.

- Seguidamente, podemos utilizar el servicio de directorio como repositorio en el cual almacenar la información que varios servidores deben compartir (por ejemplo, la configuración, información sobre el control de accesos, etc).

- Además, el directorio proporciona un protocolo estándar para gestionar toda la información contenida en él, evitando la necesidad de desarrollar dicho protocolo.

Otra utilidad que puede resultar interesante es la de emplear el servicio de directorio para indexar la documentación almacenada en el servidor Web. Actualmente hay muchas herramientas que pueden

[1] *Si funciona, no lo arregles*

generar índices basados en el formato de las marcas HTML contenidas en las distintas páginas, pero como norma general, los resultados no son lo precisos que a todos nos gustaría.

Con el advenimiento de XML, los documentos contarán con *metainformación*, es decir, información sobre la información que contienen, lo cual hará más fácil y eficaz la labor de indexación de los contenidos del servidor Web. Aquí es donde el servicio de directorio puede jugar un papel importante, ya que proporciona un acceso uniforme a la información contenida en él.

Esta última puede ser una de las mayores utilidades de los directorios, ya que permiten separar la operación de localización de la información del servidor que la contiene.

1.4 El Servicio de Directorio como infraestructura.

Un servicio de directorio accesible por multitud de aplicaciones, se convierte en una parte vital del sistema, al proporcionar un acceso uniforme a las personas, recursos y otros objetos del sistema, es decir, el directorio se ve como un todo uniforme, en lugar de un conjunto de partes independientes.

La utilización del servicio de directorio en la aplicaciones puede facilitar su desarrollo y ampliar su funcionalidad. Pensemos por ejemplo en una aplicación de videoconferencia punto a punto (entre dos personas), cuando un usuario desea entablar la conferencia con otro usuario necesita saber en que ordenador esta su otro interlocutor.

En el peor de los casos, se puede implementar un servidor de localización que permita a la aplicación determinar en que ordenador se encuentra cada usuario. Una vez determinado el puesto, debemos determinar si dicho puesto tiene las capacidades multimedia mínimas para poder establecer la videoconferencia, etc. Visto de este modo, la aplicación sería considerablemente compleja y constaría de varios módulos, cada uno especializado en determinadas tareas, aún así, lo peor de todo es que la aplicación que estamos desarrollando utiliza un protocolo propietario, y será difícil interactuar con otras aplicaciones similares.

Sin embargo, si utilizamos el servicio de directorio y almacenamos la información de configuración de los ordenadores y la ubicación de los usuarios, podemos determinar el puesto de trabajo en el que se encuentra una persona y las capacidades multimedia de dicho puesto, además con muy poco más de esfuerzo, podemos determinar el número de teléfono más cercano, para utilizarlo en caso de que el puesto de trabajo no posea capacidades multimedia.

Pero los beneficios de la utilización de un servicio de directorio no solo son los anteriores, ya que la información que utiliza una aplicación que no emplea un servicio de directorio estándar solo está accesible por esta aplicación.

En este entorno, las aplicaciones utilizan y gestionan su información, provocando que la misma información se encuentre en varias aplicaciones, lo que provoca que aparezcan inconsistencias.

Lo que se necesita en esta situación es un servicio de directorio común, que proporcione las funcionalidades que reclaman las aplicaciones, que sea multiplataforma, que sea accesible a través de un protocolo estándar y con una API estándar.

Cuando se dispone de un infraestructura de directorio de este tipo, los programadores aprovechan su tiempo desarrollando aplicaciones, en lugar de servicios de directorio específicos, del mismo tipo que se utilizan las RPC's, las primitivas de conexión TCP/IP, etc.

Cuando las aplicaciones utilizan un servicio de directorio común, diseñado de forma adecuada, es más fácil controlar los riesgos de fallo y concentrar los esfuerzos en mejorar la administración y tolerancia a fallos de este servicio.

1.5 LDAP: Conceptos y Arquitectura.

En 1988, la CCITT creó el estándar X.500, sobre servicios de directorio. En 1990 este estándar fue adoptado por la ISO, como *ISO 9594, Data Communications Network Directory, Recommendations X.500-X.521*.

X.500 organiza las entradas en el directorio de manera jerárquica, capaz de almacenar gran cantidad de datos, con grandes capacidades de búsqueda y fácilmente escalable. X.500 especifica que la comunicación entre el cliente y el servidor de directorio debe emplear el Directory Access Protocol (DAP). Pero DAP es un protocolo a nivel de aplicación, por lo que, tanto el cliente como el servidor debían implementar completamente la torre de protocolos OSI.

LDAP (Lightweight Directory Access Protocol) surge como una alternativa a DAP. Las claves del éxito de LDAP en comparación con DAP de X.500 son:

- LDAP utiliza TCP/IP en lugar de los protocolos OSI. TCP/IP requiere menos recursos y está más disponible, especialmente en ordenadores de sobremesa.

- El modelo funcional de LDAP es más simple y ha eliminado opciones raramente utilizadas en X.500. LDAP es más fácil de comprender e implementar.

- LDAP representa la información mediante cadenas de caracteres en lugar de complicadas estructuras ASN.1.

El esquema de interacción entre el cliente y el servidor LDAP sigue el siguiente esquema:

1. El cliente establece una sesión con el servidor LDAP. El cliente indica el servidor y el puerto en el que el servidor LDAP está escuchando. El cliente puede proporcionar información de autenticación o establecer una sesión anónima con los accesos por defecto.

2. El cliente efectúa las operaciones sobre los datos. LDAP proporciona capacidades de búsqueda, lectura y actualización.

3. Una vez finalizadas las operaciones, el cliente cierra la sesión.

1.5.1 Arquitectura Cliente-Servidor del servicio de Directorio.

Los servicios de directorio suelen implementarse siguiendo el modelo cliente-servidor, de modo que una aplicación que desea acceder al directorio no accede directamente a la base de datos, sino que llama a una función de la API (Application Programming Interface), que envia un mensaje a un proceso en el servidor. Dicho proceso accede al directorio y devuelve el resultado de la operación.

Algunas veces, el servidor puede convertirse en el cliente de otro servidor para conseguir la información necesaria para conseguir procesar la petición que se le ha realizado.

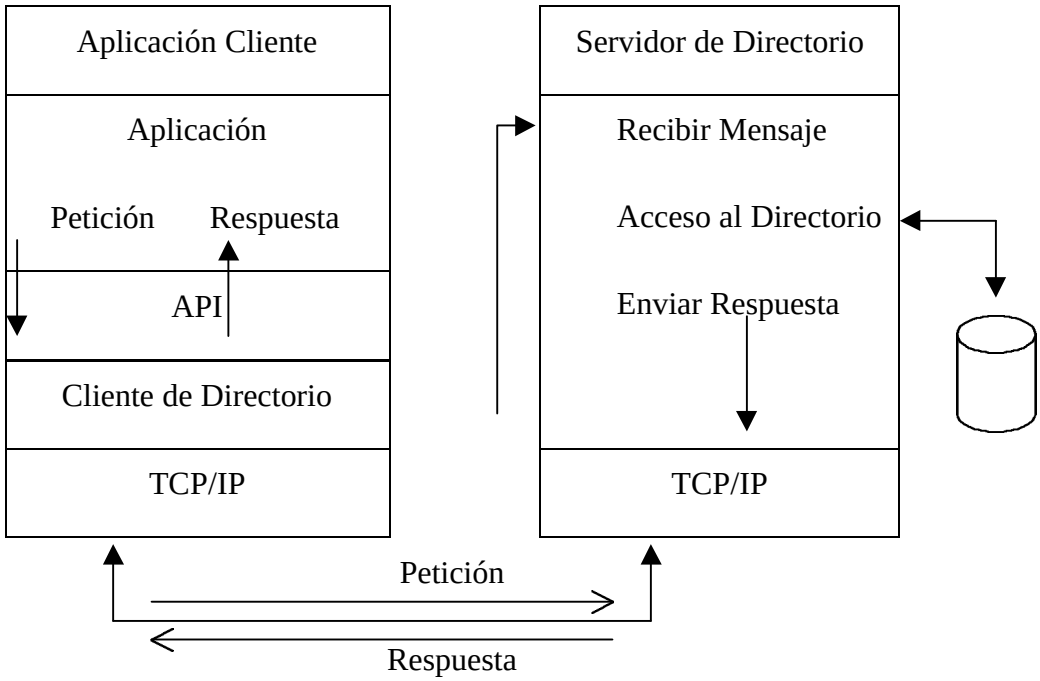

Ilustración 1 Arquitectura cliente-servidor del directorio

Siguiendo esta arquitectura el cliente no depende de la arquitectura del servidor y el servidor puede implementar el directorio de la forma más conveniente.

1.5.2 Directorios distribuidos.

El servicio de directorio puede estar centralizado o distribuido. En el caso de ser centralizado, un único servidor da todo el servicio de directorio, respondiendo a todas la consultas de los clientes. Si el directorio esta distribuido, varios servidores proporcionan el servicio de directorio.

Cuando el servicio de directorio esta distribuido, los datos pueden estar fraccionados y/o replicados. Cuando la información esta fraccionada, cada servidor de directorio almacena un subconjunto único y no solapado de la información, es decir, una entrada es almacenada en un solo servidor. Cuando la información esta replicada, una entrada puede estar almacenada en varios servidores. Generalmente cuando el servicio de directorio es distribuido, parte de la información está fraccionada y parte está replicada.

1.5.3 Seguridad del directorio.

La seguridad de la información almacenada en el directorio es uno de los aspectos fundamentales. Algunos directorios deben permitir el acceso público, pero cualquier usuario no debe poder realizar cualquier operación.

Cualquier usuario puede buscar la dirección de correo de un empleado, pero solo el empleado o el administrador deber tener permiso para modificarla. El departamento de Organización y Recursos Humanos debe tener permiso para buscar el número de teléfono privado de un empleado, pero ninguno de sus compañeros debe tener acceso a esta información.

La política de seguridad define *quién* tiene *qué tipo* de acceso sobre *qué* información.

El directorio debe permitir las capacidades básicas para implementar la política de seguridad. El directorio puede no incorporar estas capacidades, pero debe estar integrado con un servicio de red fiable que proporcione estos servicios básicos de seguridad.

Inicialmente se necesita un método para autenticar al usuario, una vez que se ha verificado la identidad del cliente, se puede determinar si esta autorizado para realizar la operación solicitada.

Generalmente las autorizaciones están asadas en ACL (Access Control List). Estas listas se pueden unir a los objetos y/o los atributos contenidos en el directorio. Para facilitar la administración de estas listas, los usuarios con los mismos permisos, son agrupados en grupos de seguridad.

Debido a que LDAP nació como alternativa *ligera* a DAP para el acceso a servidores X.500, sigue el modelo X.500.

El directorio almacena y organiza la información en estructuras de datos denominadas *entradas*.

Cada entrada del directorio describe un objeto (una persona, una impresora, etc). Cada entrada tiene un nombre llamado *Distinguished Name*(DN), que la identifica unívocamente. Un DN consiste en una secuencia de partes más pequeñas llamadas *Relative Distinguished Name* (RDN), de forma similar a como el nombre de un fichero consiste en un camino de directorios en muchos sistemas operativos (UNIX, por ejemplo).

Las entradas pueden ser organizadas en forma de árbol basándose en los DN, a este árbol de entradas de directorio se le conoce como Directory Information Tree (DIT).

Una clase de objeto es una descripción general de un tipo de objeto. El Schema define que clases de objetos se pueden almacenar en el directorio, que atributos deben contener, que atributos son opcionales y el formato de los atributos.

LDAP define primitivas de acceso y modificación de las entradas del directorio:

- Búsqueda siguiendo un criterio especificado por el usuario.
- Añadir una entrada.
- Borrar una entrada.
- Modificar una entrada.
- Modificar el DN de una entrada.
- Comparar una entrada.

1.5.4 Modelos de LDAP.

Además de definir el protocolo de acceso al directorio, el estándar LDAP define cuatro modelos que permiten entender mejor el servicio de directorio.

- **Modelo de información**, describe la estructura de la información almacenada en el directorio LDAP.
- **Modelo de nombrado**, describe como se organiza e identifica la información en el directorio LDAP.
- **Modelo funcional**, describe que operaciones pueden ser realizadas sobre la información almacenada en el directorio LDAP.
- **Modelo de seguridad**, describe como puede protegerse la información contenida en el directorio LDAP frente a accesos no autorizados.

1.5.4.1 Modelo de información.

La unidad básica de información almacenada en el directorio es la entrada (entry). Generalmente una entrada representa un objeto del mundo real (una persona, un servidor, etc), pero el modelo no exige este aspecto.

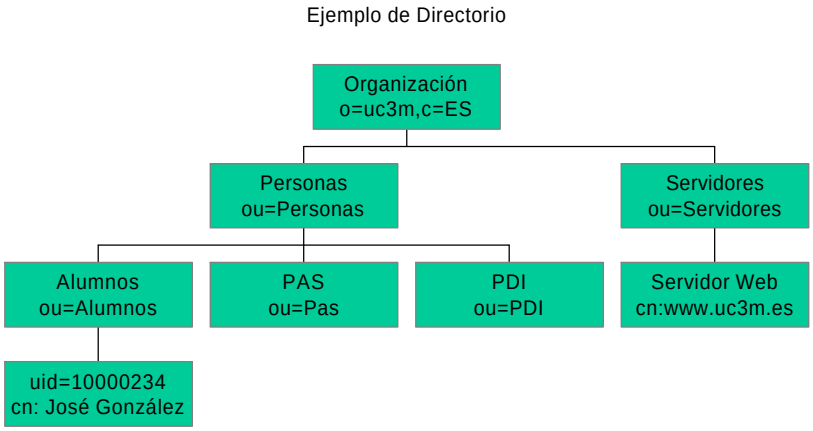

Ilustración 2 Árbol de directorio

En la figura se presenta un árbol de directorio en el cual cada una de las cajas representa una entrada en el directorio. Toda la información que se almacena en el directorio se almacena de forma jerárquica, formando el árbol de directorio, esta organización se trata con mayor profundidad en el siguiente punto.

Una entrada se compone de un conjunto de atributos, cada uno de ellos tiene un tipo y uno o varios valores. El tipo define la clase de información que va a almacenar y los valores son la información en sí.

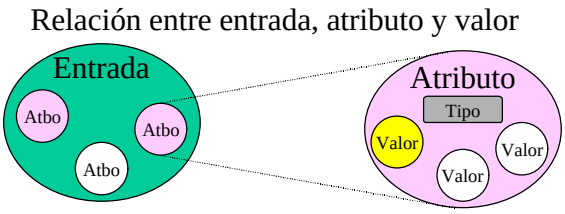

Ilustración 3 Relación entre entrada, atributo y valor

En la siguiente tabla se muestra un ejemplo de una entrada de directorio que contiene un atributo multivaluado.

Atributo	*Valor*
Cn:	José González González José
Sn:	José
Telephonenumber:	+34 91 123 456 78
Mail:	Jgonza@uc3m.es
Uid:	10000234

Ilustración 4 Ejemplo de entrada de directorio

Los tipos de los atributos tienen asociada una determinada sintaxis, que describe los tipos de datos que se van a almacenar como valores de este atributo, además define como se van a realizar las comparaciones en las búsquedas.

Algunos ejemplos de sintaxis son:

Sintaxis	*Descripción*
Bin	Información binaria
Ces	Case Exact String, deben coincidir mayúsculas y minúsculas en las búsquedas.
Cis	Case Ignore String, las mayúsculas y minúsculas no tienen relevancia en las búsquedas
Tel	Número de teléfono. Son tratados como texto, pero se ignoran espacios y guiones
Dn	Distinguished Name.

Ilustración 5 Ejemplos de sintaxis de atributos

Otras restricciones pueden asociarse a los tipos de atributos, para limitar el número de valores o el tamaño total de un atributo, por ejemplo se puede limitar el tamaño del atributo *photo*, para evitar la utilización de espacio de almacenamiento más allá de los límites razonables. Por otra parte el atributo *uid* no debería de ser multivaluado.

Los esquemas (schemas) definen el tipo de objetos que se van a almacenar en el directorio, también contiene los atributos que tienen estos objetos y si son opcionales u obligatorios. El esquema también define las subclases de objetos y en que puntos del DIT pueden aparecer.

Dado que cada servidor puede definirse su propio esquema, para permitir la interoperabilidad entre distintos servidores de directorio, se espera que un esquema común sea estandarizado (RFC 2252 y RFC 2256).

1.5.4.2 Modelo de nombrado.

El modelo de nombrado de LDAP define como se organizan y se referéncian los datos, es decir, define los tipos de estructuras que se pueden definir utilizando las entradas. Una vez organizadas las entradas formando una determinada estructura, el modelo de nombrado nos indica como referenciar estas entradas.

Las entradas son organizadas dentro del DIT en base a su Distinguished Name (DN). El DN es un nombre único que identifica de forma unívoca a una entrada. Los DNs son secuencias de Relative Distinguished Names (RDNs) y cada RDN se corresponde con una rama del DIT partiendo de la raíz hacia la entrada dentro del directorio.

De esto se deduce que no puede haber ninguna entrada *suelta*, solo la entrada raíz puede no tener entrada *padre*. En el caso de añadir una entrada en un punto inexistente en el directorio, el servidor devolverá un mensaje de error y no realizará la operación.

Esta flexibilidad permite que el directorio almacene la información de la forma más conveniente, se puede crear un grupo que contenga todas las personas de la organización y otro que contenga todos los grupos o se puede elegir una estructura que refleje la estructura jerárquica de la organización.

A continuación se muestra una ilustración con un ejemplo de DIT, en el que puede observar el esquema de nombrado.

En la cúspide del árbol se encuentra la organización, cuyo *nombre distinguido* es **o=uc3m,c=ES**, bajo esta entrada se encuentran dos entradas de *unidad organizativa (ou)*, sus *nombres distinguidos relativos* son **ou=Personas** y **ou=Servidores**, bajo la entrada **ou=Personas** se encuentran tres entradas que también pertenecen a unidades organizativas **ou=Alumnos**, **ou=PAS** y **ou=PDI**. Bajo la entrada correspondiente a la unidad organizativa de Alumnos se encuentra la entrada **uid=10000234**.

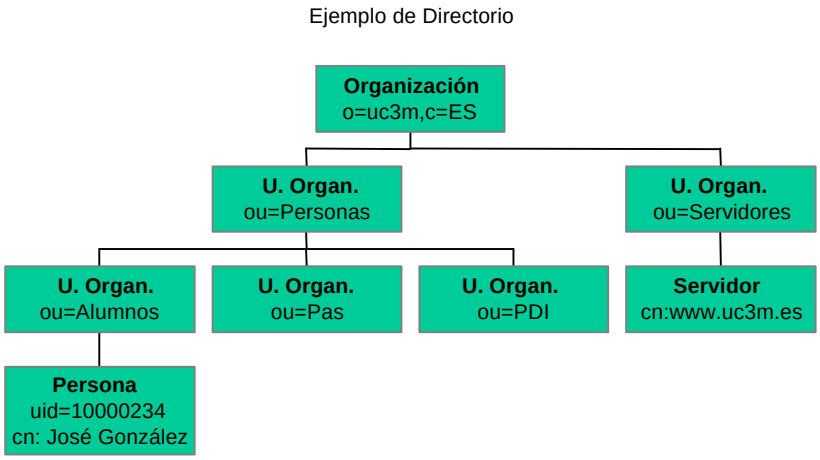

Ilustración 6 Esquema de nombrado del directorio

El Distinguished Name para el alumno José González es

```
uid=10000234, ou=Alumnos, ou=Personas, o=uc3m, c=es
```

Pero un esquema similar a este no permitiría determinar de forma cómoda los alumnos que estan matriculados de una determinada asignatura. Para ello LDAP permite el uso de alias, que serían los *enlaces simbólicos* de UNIX o los *accesos directos* de Windows.

Pero los alias no son la panacea, ya que pueden suponer una penalización importante en el rendimiento. Esto se debe a que un alias es un enlace a cualquier entrada en cualquier servidor LDAP, esto puede provocar que el servidor LDAP tenga que conectarse a otro servidor mientras esta resolviendo una consulta.

A veces los objetivos que se pretenden conseguir con los alias se pueden conseguir con las directivas *referral*.

Un servidor puede no almacenar todo el DIT, por ejemplo puede almacenar solo las entradas correspondientes a una de las ramas, por ejemplo solo la información relativa a los alumnos de la

universidad. El nodo más alto del DIT es el sufijo de todas las entradas del servidor (recordar que el esquema de nombrado es jeráquico). Dicho sufijo sería: **ou=Alumnos,ou=Personas,o=uc3m,c=es**, pero dado que el servidor no almacena todo el servidor, es necesario enlazarlo de algún modo para formar un directorio distribuido, esto se consigue con las entradas referral, que funcionan como punteros que indican donde esta la información buscada.

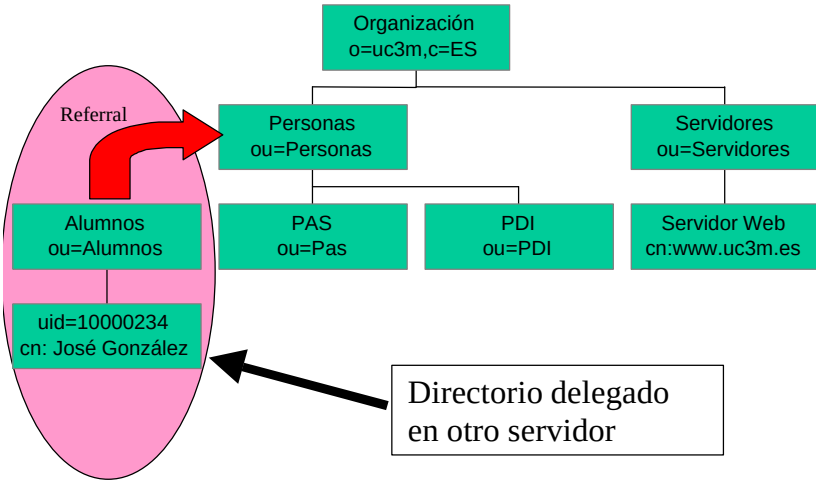

Ilustración 7 Directorio delegado. Utilización de referrals

En principio el API de LDAP permite especificar si se desea que se devuelvan las entradas de tipo *referral* o se desea que sean *seguidas automáticamente* por el servidor.

Este tipo de entradas permite particionar y distribuir el servicio de directorio entre varios servidores. Partes del DIT incluso pueden ser replicadas, permitiendo aumentar el rendimiento y la tolerancia a fallos.

1.5.4.3 Modelo funcional.

Una vez comprendido el modelo de información y el modelo de nombrado, es necesario un modelo que permita controlar el acceso a los datos contenidos en el directorio.

El modelo funcional, define un conjunto de operaciones divididas en tres grupos. Las operaciones de consulta permiten realizar búsquedas en el directorio y recuperar datos. Las operaciones de actualización permiten añadir, borrar, renombrar y modificar entradas del directorio. Las operaciones de autenticación y control permiten la identificación de los clientes y del directorio, así como controlar ciertos aspectos de una sesión.

Operaciones de consulta

Las dos operaciones de consulta permiten buscar y obtener información almacenada en el directorio.

Operación search

Esta operación permite buscar en el directorio las entradas que cumplen las especificaciones indicadas, estas especificaciones permiten indicar el punto de inicio de la búsqueda, la profundidad, que valores deben tener determinados atributos y que atributos serán devueltos si la entrada cumple las especificaciones.

Para realizar la búsqueda, se deben especificar los siguientes parámetros:

- **Base**, DN que indica el punto de partida para la búsqueda.
- **Scope**, ámbito de la búsqueda, puede ser:
  - ✓ *Base*, solo se busca en la entrada base.
  - ✓ *One*, se busca en el nivel inmediantemente inferior a la entrada base.
  - ✓ *Subtree*, se busca en todo el subárbol bajo la entrada base.
- **Filtro de búsqueda**, indica el criterio de búsqueda.
- **Atributos a devolver**, se puede indicar que atributos se devuelven y si se devuelve el valor del atributo o el tipo de dato contenido
- **Alias derreferencing**, indica si el servidor debe seguir las entradas *referral* o por el contrario debe enviarse la petición al servidor referenciado.
- **Límite**, indica el número máximo de entradas que serán devueltas o el tiempo empleado para realizar dicha búsqueda. Los servidores pueden imponer límites más estrictos que los indicados por los clientes.

Operación compare

Esta operación es similar a la operación de búsqueda utilizando un filtro de equiparación, pero la diferencia se encuentra en que dada una entrada que cumple con las especificaciones pero que no tiene el atributo que se desea devolver, el directorio devuelve un valor especial, para indicar que dicha entrada cumple con los requisitos, pero no dispone del atributo.

Operaciones de actualización

Hay cuatro operaciones que permiten añadir, borrar, renombrar (modificar el DN) y modificar.

Operación add

Esta operación permite añadir nuevas entradas al directorio, recibe como parámetros el DN de la entrada a crear un los atributos y los valores asociados. Para poder realizar esta operación se debe cumplir que:

- El nodo *padre* de la entrada exista en el directorio.
- No debe haber otra entrada con el mismo DN.
- La entrada debe cumplir con los requisitos especificados en el esquema.
- El control de accesos permita esta operación.

Operación delete

Esta operación permite eliminar entradas del directorio, recibe como parámetro el DN de la entrada a borrar. Para poder realizar esta operación, se deben cumplir las siguientes condiciones:

- La entrada a borrar debe existir en el directorio.
- Dicha entrada no debe tener ningún *hijo*.
- El control de accesos debe permitir esta operación.

Operación rename

Esta operación permite modificar el DN de una entrada, para poder renombrar una entrada se deben cumplir las siguientes condificiones:

- La entrada a renombrar debe existir.
- No debe existir una entrada con el nuevo DN.
- El control de accesos debe permitir esta operación.

LDAPv2 no tiene la operación de modificar el DN de una entrada, en su lugar tiene la operación modificar RDN, que permite modificar el RDN de una entrada, pero no permite mover la entrada de una rama a otra del DIT.

Operación modify

Permite la modificación de los atributos de una entrada. Para poder ejecutarse esta operación deben cumplirse las siguientes condiciones:

- La entrada a modificar debe existir.
- Las modificaciones de los atributos deben realizarse.
- La entrada resultante debe ser conforme al esquema.
- El control de accesos debe permitir la actualización.

Este punto indica que las operaciones en LDAP son atómicas, si alguna de las modificaciones falla, toda la operación de actualización falla.

Operaciones de autenticación y control

LDAP incorpora dos operaciones de autenticación (*bind* y *unbind*) y una de control (*abandon*).

Operación bind

Esta operación permite autenticar al cliente frente al directorio. Hay varios tipos de autenticación, desde una sesión anónima, una sesión autenticada, en la que el usuario se ha identificando proporcionando la contraseña, hasta una sesión cifrada, utilizando los mecanismos SASL (SASL se añadió a LDAPv3 para superar la debilidad del esquema de autenticación propuesta en LDAPv2).

Las sesiones anónimas, en las que no se ha especificado el usuario ni la contraseña, que solo tienen sentido para operaciones de búsqueda, ya que no se ha realizado ningún tipo de comprobación de la identidad del cliente.

La autenticación básica, en las que al establecer la conexión se envían al servidor el nombre distinguido del usuario y su contraseña en claro, el servidor considera que el cliente se ha autenticado si la contraseña coincide con la almacenada en el campo ***userPassword***. Esta información es enviada en claro desde el cliente al servidor[2], lo cual implica un riesgo de seguridad muy alto.

[2]Algunas implementaciones codifican en base64 esta información, según las especificaciones MIME.

LDAPv3 contempla el establecimiento de sesiones cifradas, en este tipo de sesiones, el cliente envía el nombre distinguido del usuario, el método de autenticación que va a emplear y las credenciales necesarias para autenticarse.

El mecanismo SASL, introducido en LDAPv3, establece que los métodos de autenticación son los siguientes, Kerberos versión 4, S/Key, GSSAPI, CRAM-MD5 y External. La principal ventaja de este diseño estriba en que permite la ampliación a nuevos métodos de autenticación utilizando el método external, de hecho SSL (y su sucesor TSL) utilizan este método.

Operación unbind

Esta operación cierra la conexión con el servidor LDAP.

Operación abandon

Esta operación permite indicar al servidor LDAP que el cliente abandona la operación en curso.

1.5.4.4 Modelo de seguridad

En LDAPv2 solo se permiten sesiones anónimas y autenticación mediante texto en claro, debido a esto algunos fabricantes incorporaron mecanismos de seguridad adicionales, como Kerberos.

La operación bind de LDAPv3 tiene soporte para Simple Authenticacion Security Layer (SASL), además se han definido operaciones extendidas, una de ellas relacionada con la seguridad es la *Extension for Transport Layer Security (TSL) for LDAPv3*.

Aún cuando el modelo de control de accesos no se ha especificado en LDAP, todos los servidores implementan algún tipo de control, más o menos flexible.

1.6 El formato de intercambio de datos LDIF

El formato LDIF es el estándar para representar entradas del directorio en formato texto. Una entrada del directorio consiste en dos partes. El DN o *nombre distinguido*, que debe figurar en la primera línea de la entrada y que se compone de la cadena **dn:** seguida del DN de la entrada. La segunda parte son los atributos de la entrada. Cada atributo se compone de un nombre de atributo, seguido del carácter dos puntos, **:**, y el valor del atributo. Si hay atributos multivaluados, deben ponerse seguidos.

No hay ningún orden preestablecido para la colocación de los atributos, pero es conveniente listar primero el atributo **objectclass**, para mejorar la legibilidad de la entrada.

```
dn: uid=jose,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
objectclass:person
objectclass:organizationalPerson
objectclass:inetOrgPerson
objectclass:account
objectclass:posixAccount
objectclass:top
uid:jose
sn:LOPEZ ABELLAN
cn:JOSE JUAN LOPEZ ABELLAN
description:LOPEZ ABELLAN, JOSE JUAN
loginshell:/bin/sh
uidnumber:202
gidnumber:100
gecos:Jose J. Lopez
mail:jose@di.uc3m.es
room:1.1.D.04
postaladdress:Despacho: 1.1.D.04$Edificio AGUSTIN DE BETANCOURT$CAMPUS DE
 LEGANES$BUTARQUE, 15.$28911 LEGANES$MADRID
telephonenumber:916249961
telephonenumber:916249980
homedirectory:/home/DI/jose
```

Ilustración 8 Entrada del directorio en formato LDIF

Las líneas excesivamente largas pueden partirse con un retorno de carro, y añadiendo un espacio al principio de la siguiente línea.

```
Description:Este es un ejemplo
 de atributo excesivamente largo
 partido en varias líneas.
```

Ilustración 9 Ejemplo de atributo largo partido en varias líneas

Si un atributo contiene valores no ASCII, como por ejemplo una imagen JPEG, se codifica en formato Base64

```
Dn: uid=rafa,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
Objectclass: person
Objectclass: organizationalPerson
Objectclass: inetOrgPerson
Objectclass: account
Objectclass: posixAccount
Objectclass: top
Uid: rafa
Sn: CALZADA PRADAS
Cn: RAFAEL CALZADA PRADAS
Description: CALZADA PRADAS, RAFAEL
Loginshell: /bin/sh
Uidnumber: 1001
Gidnumber: 100
Gecos: RAFAEL CALZADA PRADAS, 700000170, 52139294L
Telephonenumber: 916249481
Room: 1.0.G.02
Postaladdress: Despacho: 1.0.G.02$Edificio AGUSTIN DE BETANCOURT$CAMPUS DE LEG
 ANES$BUTARQUE, 15.$28911 LEGANES$MADRID
Mail: rafa@di.uc3m.es
Homedirectory: /home/DI/rafa
Jpegphoto:: R0lGODlheAHdAfUAAKysqKmlmQwPDggKCVVNSk1KRycoJhwhJcm8p7a1qktIQUdBP
 83MxMm8wObu6ubm1djR28jO10dISDxBSFNhcVNbYxcXFRESEae+xpumv6GugJuHgScuOScqLYFyc
 XxvZmxhWWFfWJF+fo17crrFz7m5uYOBeH58b6yYmqeSj+T7/NX3+lVUUkdMUiAhHhcaG3FzcnFwZ
 XBlZGJjZbqlpJyblJ2LjFFWW0E/PI2Pk3eCk83m546Lfzg5OjQyLmBVUiwAAAAAeAHdAQUG/sDYK
 UYsykKhgpKFBMVgHyjsRK1ar1iribo9mb5gsJdaFHrD6PRXZGKz0yd3e/4+i8kxGYvV6nUOBy8vF
 4SFFxYWL4iILy6NLpAWhxaQjpSVlJcuBwY+Bi6LmZqfoJWmLgapqZGSkqCIlauQPrQ9CwsKBD8gH
 71VUTFRV3JrX2cjJshfyHJsI24iz3NdVWDIym/ZdGrc3d7VamNb41lDME8nMOcxMxUsEhIFu0VDV
 Orq6flTUyYw/eP+/G3xx6XgGHH91k05M4YHwhPKipngMRENMWUQq8Dw4GFMRw8yZoRg0cPHn0ACX
--More--
```

Ilustración 10 Atributo binario codificado en base64

El formato LDIF también puede ser utilizado para realizar actualizaciones y/ borrar entradas del directorio. El formato en este caso contiene en la primera línea el DN de la entrada sobre la que se aplica el cambio. La segunda línea indica el cambio a realizar y las siguientes líneas contienen los pares atributo-valor que componen el cambio.

Para añadir una entrada

```
Dn: nombre distinguido
Changetype: add
Tipo_atributo: valor
```

```
dn: uid=jose,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
changetype: add
objectclass:person
objectclass:organizationalPerson
objectclass:inetOrgPerson
objectclass:account
objectclass:posixAccount
objectclass:top
uid:jose
sn:LOPEZ ABELLAN
cn:JOSE JUAN LOPEZ ABELLAN
description:LOPEZ ABELLAN, JOSE JUAN
loginshell:/bin/sh
uidnumber:202
gidnumber:100
gecos:Jose J. Lopez
mail:jose@di.uc3m.es
room:1.1.D.04
postaladdress:Despacho: 1.1.D.04$Edificio AGUSTIN DE BETANCOURT$CAMPUS DE
 LEGANES$BUTARQUE, 15.$28911 LEGANES$MADRID
telephonenumber:916249961
telephonenumber:916249980
homedirectory:/home/DI/jose
```

Ilustración 11 Formato LDIF para añadir una entrada

Para borrar una entrada basta indicar el cambio delete

Dn: *nombre distinguido*
Changetype: delete

```
Dn: uid=jose,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
Changetype: delete
```

Ilustración 12 Formato LDIF para borrar una entrada

Para modificar una entrada

Dn: *nombre distinguido*
Changetype: modify
TipoCambio: atributo
atributo: *valor*

Por ejemplo para añadir un atributo:

```
Dn: uid=jose,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
Changetype: modify
Add: telephoneNumber
TelephoneNumber: 916249500
```

Ilustración 13 Formato LDIF para añadir un atributo

Para eliminar todas los valores de un determinado atributo

```
dn: uid=jose,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
changetype: modify
delete: telephoneNumber
```

Ilustración 14 Formato para eliminar un atributo completamente

Para borrar un valor de un atributo:

```
dn: uid=jose,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
changetype: modify
delete: telephoneNumber
telephoneNumber: 916249500
```

Ilustración 15 Formato para eliminar un valor de un atributo

Si deseamos actualizar un atributo

```
dn: uid=jose,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
changetype: modify
replace: telephoneNumber
telephoneNumber: 916249500
```

Ilustración 16 Formato para modificar un valor de un atributo

Varias operaciones se pueden combinar en un único fichero si las separamos por un guión

```
dn: uid=jose,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
changetype: modify
delete: telephoneNumber
telephoneNumber: 916249500
-
add: mail
mail: pepe@pepe.com
-
delete: description
```

Ilustración 17 Formato para combinar varias operaciones

Cuando se realiza una operación que combina varias, el servidor las trata como una única operación, por lo que si una de ellas falla, el servidor devolverá el error correspondiente y dejará inalterada la entrada.

Otro tipo de cambio permitido es la modificación del *nombre distinguido relativo* (modrdn), que permite cambiar parte del nombre distinguido (LDAPv3 permite cambiar el DN completo de una entrada).

```
Dn: uid=jose,ou=SERVICIO DE INFORMATICA,ou=PAS y PDI,ou=Personas,o=uc3m,c=es
```

```
Changetype: modrdn
newrdn: uid=rafa
deleteoldrdn: 0
```

Ilustración 18 Formato para modificar el RDN de una entrada

El parámetro **deleteoldrdn** permite especificar si debe borrarse la entrada original.

Los programas encargados de realizar estas operaciones en el directorio son:

Programa	*Función*
Ldapadd	Permite añadir entradas en el servidor de directorio
Ldapdelete	Permite borrar entradas del directorio
Ldapmodify	Permite modificar entradas del directorio
Ldapmodrdn	Permite modificar el RDN de una entrada.
Ldapsearch	Permite realizar búsquedas en el directorio.

Ilustración 19 Herramientas para la gestión del directorio

1.7 Filtros de búsqueda en LDAP

El formato básico de los filtros de búsqueda de LDAP es el siguiente:

Atributo operador valor

El atributo se refiere al atributo sobre el que vamos a realizar la operación de comparación. El operador puede ser uno de los siguientes:

Operador	*Descripción*	*Ejemplo*
=	Devuelve las entradas cuyo atributo tiene el valor especificado.	`Mail=rafa@di.uc3m.es` Busca todas las entradas con la dirección de correo especificada

>=	Devuelve las entradas cuyo atributo sea mayor o igual que el valor especificado	Sn>=calzada Busca todas las entradas cuyo apellido vaya desde calzada hasta el final
<=	Devuelve las entradas cuyo atributo sea menor o igual que el valor especificado	Sn<=calzada Busca todas las entradas desde el principio hasta calzada
=*	Devuelve las entradas que tienen valor asignado en el atributo especificado	Mail=* Busca las entradas que tengan asignado valor en el atributo mail
~=	Devuelve las entradas cuyo atributo tenga un valor similar al especificado	Sn ~= calzado Busca todas las entradas cuyo valor sea parecido a calzado[3]

Ilustración 20 Operadores de búsqueda

Además de los operadores presentados, el carácter ***** tiene el significado de comodín y puede ser empleado con el operador **=**. Por ejemplo, cn=Da*G*z, encontraría las entradas de David Gutiérrez y de Daniel González.

Pero además, los operadores de busquedas pueden combinarse utilizando los operadores booleanos, dando lugar a expresiones de búsqueda más complejas. La sintaxis para combinar filtros de búsquedas es la siguiente:

```
( operador_n-ario (filtro1) (filtro2) (filtro3) … )

(operador_unario (filtro))
```

Operador n-ario	*Significado*
&	AND lógico de los filtros.

[3] LDAP utiliza los algoritmos *metaphone* o *soundex* para realizar estas aproximaciones, para más información, consultar la sección de referencias.

|	OR lógico de los filtros.

Ilustración 21 Operadores n-arios para filtros de búsqueda

Operador unario	*Significado*
!	NOT lógico del filtro.

Ilustración 22 Operador unario para filtros de búsqueda

Veamos a continuación algunos ejemplos de filtros de búsqueda y su significado.

Filtro	*Significado*
`(& (uid=*0202) (cn=Rafael*))`	Busca entradas cuyo campo **uid** terminé en **0202** y cuyo campo **cn** empiece por **Rafael**
`(| (cn=José) (cn=Juan))`	Busca entradas cuyo campo **cn** sea **José** o **Juan**
`(| (& (cn=Juan) (ou=Ventas)) (cn=José))`	Busca entradas cuyo campo **cn** sea **Juan** y cuyo campo **ou** sea **Ventas**, y entradas cuyo campo **cn** sea **José**
`(! (cn=Rafael))`	Busca todas las entradas cuyo campo **cn** no sea **Rafael**

Ilustración 23 Ejemplos de filtros de búsqueda

Para realizar búsquedas sobre atributos cuyos valores contienen alguno de los caracteres reservados para la construcción de los filtros, deben utilizarse *secuencias de escape*.

Carácter	*Valor (Decimal)*	*Valor (Hexadecimal)*	*Secuencia de escape*
*	42	0x2A	\2A
(	40	0x28	\28
)	41	0x29	\29
\	92	0x5C	\5C
NUL	0	0x00	\00

Ilustración 24 Secuencias de escape que se deben utilizar en filtros de búsqueda

2 Descripción del trabajo final.

2.1 Servicio de directorio

2.1.1 Modelo de información

Al diseñar el modelo de información se ha intentado ser lo más fiel posible a las clases y atributos predefinidos en la RFC 2256. Sin embargo, ha sido necesario definir clases que albergasen los atributos propios de la Universidad Carlos III.

La clase **uc3mPersona** recoge los atributos que son comunes a las distintas personas cuyos datos son almacenados en el servicio de directorio.

Uc3mpersona	*Significado*
NIF	Número de NIF de la persona

Ilustración 25 Atributos de la clase uc3mPersona

La clase **uc3mAlumno**, contiene los atributos que son comunes a los alumnos de la universidad, en principio solo el Número de Identificación del Alumno es relevante.

Uc3malumno	*Significado*
NIA	Número de identificación de Alumno

Ilustración 26 Atributos de la clase uc3mAlumno

La clase **uc3mTrabajador**, recoge los atributos que son comunes al Personal de Administración y Servicios y Personal Docente e Investigador de la universidad.

Uc3mtrabajador	*Significado*
NIU	Número de identificación Universitario

Ilustración 27 Atributos de la clase uc3mTrabajador

2.1.2 Modelo de nombrado

En el modelo de nombrado se ha organizado en un primer nivel en *Alumnos, Personal de Admnistración y Servicios* y *Personal Docente e Investigador*. Esta división ha permitido delegar las entradas correspondientes a alumnos en un segundo servidor LDAP.

Ilustración 28 Esquema del primer nivel del árbol de directorio

La información se almacenará en dos servidores de directorio relacionados entre sí, el servidor principal, contendrá las entradas correspondientes a la organización y al Personal de Administración y Servicios y al Personal Docente e Investigador. El servidor secundario tendrá delegadas las entradas correspondientes a los alumnos matriculados.

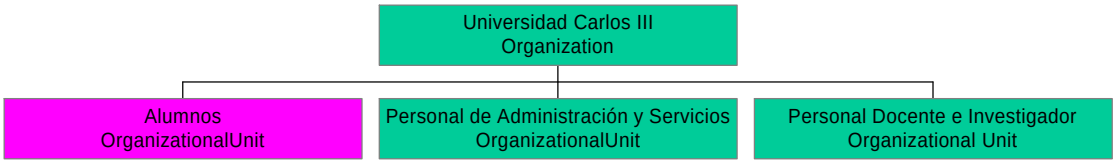

Ilustración 29 Esquema de distribución de los datos entre los servidores de directorio

Bajo la entrada de Personal de Administración y Servicios se han creado entradas que corresponden con los servicios de la universidad y bajo dichas entradas se han creado las entradas correspondientes a los trabajadores correspondientes a cada servicio.

Ilustración 30 Esquema de nombrado para Personal de Administración y Servicios

Bajo la entrada de Personal Docente e Investigador se han creado entradas que se corresponden con los departamentos de la universidad, bajo la entrada de cada departamento se han creado las entradas correspondientes a las áreas de cada departamento. Bajo cada área se han incluido las entradas correspondientes a los docentes de dicha área.

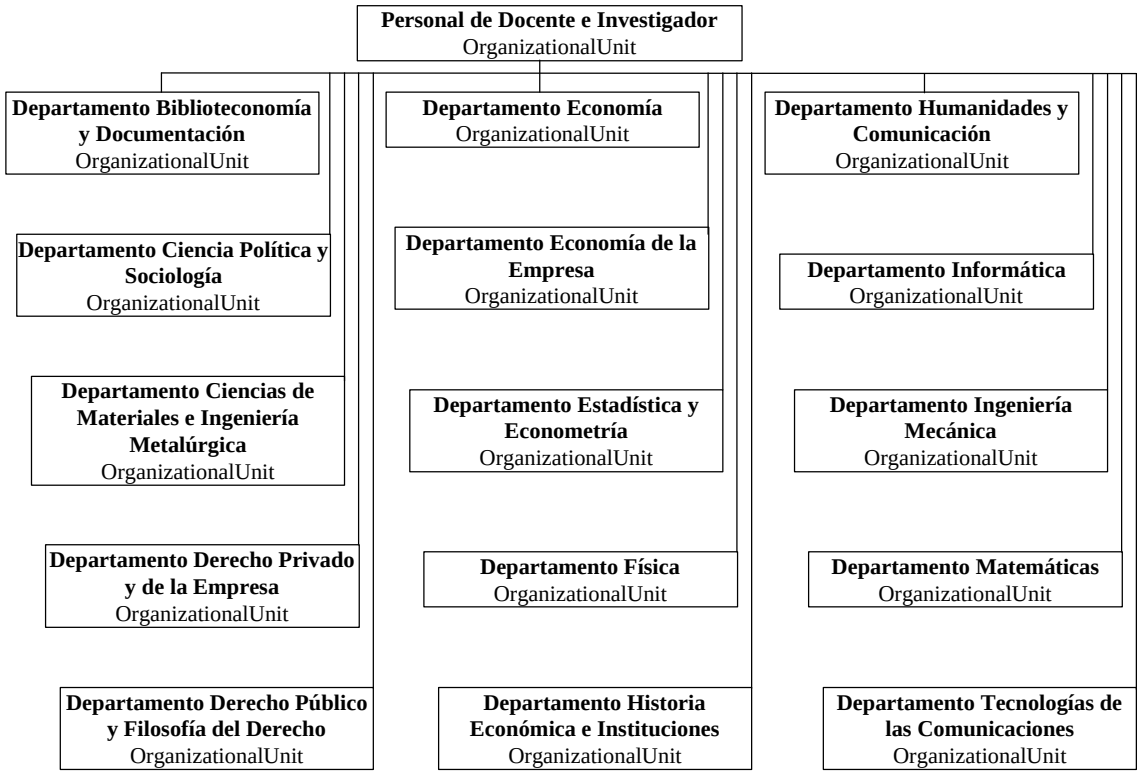

Ilustración 31 Esquema de nombrado para Personal Docente e Investigador

2.1.3 Modelo funcional

Las operaciones que se pueden realizar sobre los datos almacenados en el directorio vienen determinadas por el servidor instalado (OpenLDAP), que solo soporta LDAP versión 2.

OpenLDAP soporta las dos operaciones de consulta (*search* y *compare*).

Respecto a las operaciones de actualización, OpenLDAP permite las operaciones *add*, *delete* y *modify*. La operación *rename* la soporta parcialmente, ya que LDAPv2 solo contempla la posibilidad de renombrar el RDN de una entrada.

En cuanto a las operaciones de autenticación y control, OpenLDAP solo implementa las opciones recogidas en LDAPv2, esto es, permite el establecimiento de una sesión anónima y la autenticación básica, en la que la contraseña del usuario viaja en claro por la red. Para solucionar este grave problema se han utilizado herramientas para cifrar las conexiones extremo a extremo, por ejemplo ***stunnel***.

Las limitaciones en las operaciones que se pueden realizar sobre los datos, serán superadas en la versión 2.0 de OpenLDAP.

2.1.4 Modelo de seguridad

Como ya se ha comentado en el punto anterior, debido a los problemas de seguridad que conlleva LDAPv2, se han utilizado herramientas de cifrado para impedir que las contraseñas viajen en claro a través de la red, también se han tenido en cuenta la posibilidad de utilizar tecnologías de red que no permiten la interceptación de las comunicaciones como Ethernet conmutada y ATM.

Respecto al control de accesos, se ha establecido que el administrador del servicio de directorio tiene permisos de lectura y modificación sobre todos los atributos. Este usuario puede crear entradas nuevas y puede realizar cualquier operación.

El propio usuario puede realizar cualquier operación sobre sus datos, tiene por lo tanto permisos de lectura y modificación.

El resto de usuarios, incluidos los usuarios anónimos, tienen permiso de lectura sobre todos los atributos del usuario, excepto sobre su contraseña.

El atributo ***userPassword***, tiene un tratamiento especial, ya que sobre él solo se permiten operaciones de comparación (necesarias para realizar la autenticación).

Para mejorar la seguridad de la administración de los datos contenidos en el servicio de directorio, se han creado *administradores de datos*, estos usuarios tienen permisos de modificación de determinados atributos sobre todas las entradas contenidas en el servicio de directorio. Así por ejemplo, el ***administrador de claves***, tiene permisos para actualizar el atributo ***userPassword*** de cualquier usuario del servicio de directorio. El ***administrador de grupos***, tiene permisos para crear y actualizar cualquier grupo de usuarios empleado para la autenticación en el servicio de Web.

Este esquema permite evitar la utilización de la contraseña del administrador del servicio de directorio en gran multitud de operaciones, por lo que si alguna de las constraseñas de administradores de datos se ve comprometida, solo una parte de los datos almacenados en el directorio se verá comprometida.

Por motivos estéticos, se ha establecido listas de control de acceso de modo que ni los *administradores de datos* ni los *grupos de usuarios* son accesibles en lectura, solo en comparación. Esta solución permite evitar que los datos correspondientes a estas entradas sean mostrados a usuarios sin privilegios de administración del servicio de directorio.

2.1.5 Relación entre los distintos servidores LDAP

Como ya se ha comentado anteriormente, el servidor principal contiene todas las entradas correspondientes a la Universidad Carlos III, el servidor secundario únicamente contiene las entradas correspondientes a los alumnos.

Este diseño permite distribuir la carga entre ambos servidores, ya que si bien, el número de alumnos varias veces superior, tanto las consultas como las operaciones de actualización van a ser menos frecuentes.

La documentación existente de OpenLDAP indica que soporta referencias LDAPv2 siguiendo la RFC 1777. La experiencia nos ha demostrado, que el soporte para referencias es limitado, ya que si bien las búsquedas en sub-árbol funcionan correctamente, las operaciones de autenticación y las búsquedas en base no funcionan.

Ante esta situación, se ha tomado la determinación de esperar a la próxima versión de OpenLDAP, que incluye el soporte para referencias siguiendo la RFC 2251. Hasta ese momento las búsquedas y operaciones de autenticación deberán realizarse en el servidor correspondiente.

2.1.6 Medidas de contingencia

A lo largo de la instalación y pruebas de los servidores de directorio se detectaron fallos en el servidor que provocaban caídas en el servidor de directorio, estas caídas se debían a que los datos almacenados en el servidor no eran conformes al esquema.

Para evitar estas situaciones, se habilitó la comprobación del esquema en las operaciones de actualización de datos. Esta medida surtió los efectos deseados y desde ese momento no se han detectado fallos de funcionamiento en el servidor de directorio, pero esto no ha sido considerado suficiente, por lo que se han desarrollado programas de

monitorización del servicio de directorio, que lanzan la ejecución del servicio en el momento que se detecta una caída de éste.

Para evitar los problemas derivados de una avería hardware en alguno de los dos servidores, se han configurado de modo que un servidor es replica del otro y viceversa, de modo que los datos siempre se encuentren en los dos servidores y que salvo una avería hardware simultánea en ambos servidores, el servicio estaría garantizado.

Para el caso de una avería hardware simultánea, el Servicio de Informática cuenta con un servicio de copias de seguridad de los servidores centrales, estas copias se realizan durante la madrugada y por lo tanto solo se perdería los datos correspondientes a las operaciones de actualización realizadas desde la última copia de seguridad.

2.1.7 Parámetros de instalación y configuración de los servidores OpenLDAP

Los servidores OpenLDAP se han instalado en el directorio `/usr/local/openldap-`*`versión`* donde versión se corresponde con la versión instalada (utilizando la opción `--prefix=/usr/local/openldap-`*`versión`*).

Un enlace simbólico relaciona el directorio `/usr/local/openldap` con la versión en explotación. Este método permite al administrador del servicio de directorio instalar y probar nuevas versiones de la distribución sin que esto afecte al servicio en explotación. Además facilita la *marcha atrás* de una actualización en el caso de detectar fallos de funcionamiento.

Si se desea hacer uso de las librerías y ficheros de cabecera para compilar algún programa, habrá que incluir los directorios `/usr/local/openldap/include` y `/usr/local/openldap/lib`. Al realizar la instalación se han compilado tanto las librerías estáticas como las dinámicas o compartidas, por lo que los nuevos programas pueden hacer uso de las librerías necesarias.

3 Conclusiones y trabajos futuros.

Múltiples y muy variadas son las aplicaciones del servicio de directorio, en un primer momento el servicio de directorio debería ir integrándose progresivamente con las aplicaciones existentes.

Los primeros pasos que deben darse pueden ir encaminados a uniformizar el control de accesos a áreas restringidas de los servidores Web.

También puede inciarse un proceso hacia la *contraseña única* para acceder a los servicios prestados en servidores centrales (integración de los mecanismos de acceso a servidores Linux, Solaris y Windows NT con el servicio de directorio LDAP).

Las nuevas aplicaciones que vayan desarrollándose pueden hacer uso del directorio en la medida en que lo necesiten.

4 Referencias.

4.1 LDAP.

- *Understanding LDAP*, Heinz Johner, Larry Brown, Franz-Stefan Hinner, Wolfgang Reis, Johan Westman, Junio 1998.

  Un excelente manual de iniciación al servicio de directorio. Ha servido de guía para la redacción del punto 1.2. Así como para el diseño del servicio de directorio. Este libro esta accesible públicamente en http://www.redbooks.ibm.com/abstracts/sg244986.html.

- *Understanding and Deploying LDAP Directory Services*, Timothy A. Howes, Mark Smith y Gordon Good, 1999, Ed Macmillan Network Architecture and Development Series.

  Excelente libro en el que se exponen los conceptos principales del servicio de directorio. El texto se complementa con varios casos prácticos, en los que se cubre el diseño y la implantación. Algunos capítulos del libro están accesibles públicamente en http://developer.netscape.com:80/docs/books/macmillan/ldap/ldapbk.html. Este libro se puede adquirir directamente en Amazon.

- *LDAP Schema Viewer*

  Servidor Web con las clases definidas en las distintas RFCs. Este servidor es públicamente accesible desde http://www.hklc.com/ldapschema/

- *Microsoft Active Directory Server*,

  La información referente a este producto junto con Windows 2000 puede encontrarse en http://www.microsoft.com/windows/server/Overview/exploring/directory.asp

- *Netscape Java Directory SDK*, Kit de desarrollo de LDAP para aplicaciones Java.

  Se ha utilizado para desarrollar la aplicación empleada para medir la capacidad de respuesta del servidor LDAP. Este kit esta accesible públicamente en http://developer.netscape.com.

- *Novell Directory Services. LDAP Services for NDS*,

  Información sobre el sistema operativo Novell Netware, accesible desde http://www.novell.com/products/nds/ldapov.html

- *OpenLDAP*.Software servidor LDAP.

  Este software esta accesible públicamente en http://www.openldap.org. La versión actual (1.2.7) implementa LDAPv2, la versión 2.0 que implementará el estándar LDAPv3 tiene prevista su publicación en el cuarto cuatrimestre de 1999.

- *perl-ldap*, módulo PERL

  Módulo PERL utilizado para acceder al directorio desde servidores UNIX, este modulo esta accesible en cualquier mirror de CPAN, por ejemplo el de RedIRIS. Puede instalarse desde el interfaz **cpan** de perl mediante *install Net::LDAP*.

- *The Four LDAP Models*, Tim Howes, Mark Smith y Gordon Good.

  Este artículo es un resumen capítulo 3 del libro *Understanding and Deploying LDAP Directory Services*, y está accesible públicamente en http://developer.netscape.com/viewsource/ldap_models/ldap_models.html.

- *RFC 1777 Lightweight Directory Access Protocol (v2)*

- *RFC 2251 Lightweight Directory Access Protocol (v3).*

- *RFC 2252 Lightweight Directory Access Protocol (v3): Attribute Sintax Definitions.*

- *RFC 2253 Lightweight Directory Access Protocol (v3): 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 2244 ACAP -- Application Configuration Access Protocol*.
- *Why do I need Directory when I could use a Relational Database?,*

  Esta presentación esta accesible en http://www.stanford.edu/~hodges/talks/EMA98-DirectoryServicesRollout/Steve_Kille/index.htm.

- *El directorio LDAP*, por Julio Sánchez

  Ponencia expuesta por uno de los desarrolladores de OpenLDAP en el II Congreso HISPALINUX desarrollado en la Universidad Carlos III en Noviembre de 1999. Las transparencias de esta ponencia están accesibles públicamente en http://congreso.hispalinux.es/docs/plenarias/ldap.ps.gz.

- Innosoft LDAP World

  Servidor Web que contiene la información sobre las especificaciones del protocolo LDAP, accesible desde http://www.critical-angle.com/ldapworld/index.html

- The SLAPD y SLURPD Administrator's Guide

  Guía del administrador del servidor LDAP de la distribución de la Universidad de Michigan. Prácticamente es aplicable a OpenLDAP, salvo contadas excepciones. Esta accesible públicamente en: http://www.umich.edu/~dirsvcs/ldap/doc/guides

- Algoritmo Soundex

  Algoritmo de búsqueda fonética desarrollado por Levi Cook para la elaboración del censo de 1880 en EEUU con el objetivo de realizar búsquedas aproximadas utilizando las características fonéticas de las palabras. Este algoritmo esta descrito

en el libro *The art of computer programming: Sorting and searching*, de Donald Knuth. Más información puede encontrarse en: http://www.rootscomputing.com/howto/soundex/soundex.htm
http://www2.lib.udel.edu/subj/hist/resguide/soundex.htm

- Algoritmo Metaphone

  Este algoritmo de búsqueda fonética fue descrito por Lawrence Philips en la publicación *Computer Language* en 1990 y pretende solventar las deficiencias de Soundex. Más información puede encontrarse en: http://www.lanw.com/java/phonetic/default.htm.

4.2 Certificación.

- *OpenSSL.*Software de cifrado.

  Software que permite tanto el establecimiento de sesiones utiliznado el protocolo SSL como la emisión de certificados X.509v3. Este software esta accesible públicamente en http://www.openssl.org.

- *Building a Corporate Public Key Infrastructure*, Christopher King.

  Artículo que cubre los principales aspectos a tener en cuenta al desarrollar una infraestructura de clave pública. Este artículo esta accesible públicamente en http://www.infoseceng.com/corppki.html.

- *Using SSLeay to generate site and client certificates for Microsoft Internet Information Server,* Philip Wherry, Enero 1.997.

  Artículo que describe paso a paso como emitir un certificado adaptado para ser utilizado por el programa Microsoft Internet Information Server (IIS). Este artículo esta accesible públicamente en http://www.wherry.com/private-ca.html.

- *Gpkcs11*, Librería de desarrollo PKCS#11

  Esta librería esta accesible públicamente en ftp://alpha.gnu.org/pub/gnu/gpkcs11/

- OpenSSL PKCS#12 Program FAQ

  Accesible públicamente en http://www.drh-consultancy.demon.co.uk/pkcs12faq.html

4.3 SSL

- *Introduction to SSL.*

  Introducción al protocolo SSL, desarrollado inicialmente por Netscape. Éste documento esta accesible públicamente en http://developer.netscape.com/docs/manuals/security/sslin/contents.html.

- *Stunnel.*

  Programa que permite cifrar conexiones a servicios TCP/IP, permite incrementar la seguridad de los servicios accesibles en un servidor (permitiendo que la conexión con cliente sea cifrada), como cifrar el acceso al servidor desde el cliente. Este programa esta accesible públicamente en http://mike.daewoo.com.pl/computer/stunnel.

- *Sslwrap.*

  Programa que permite cifrar conexiones a servicios TCP/IP, permite incrementar la seguridad de los servicios accesibles en un servidor (permitiendo que la conexión con cliente sea cifrada), como cifrar el acceso al servidor desde el cliente. Este programa esta accesible públicamente en http://www.rickk.com/sslwrap.

4.4 Web.

- *Apache.* Software de servidor Web.

  Uno de los servidores Web más utilizado en Internet. Permite además la inclusión de módulos adicionales al servidor, para expandir sus capacidades. Este software esta accesible públicamente en http://www.apache.org. Los módulos diseñados para este servidor Web están accesibles en http://modules.apache.org.

- *ModSSL*. Módulo de cifrado para el servidor Apache.

  Software que permite incorporar al servidor Apache las funcionalidades necesarias para el establecimiento de comunicaciones seguras basadas en el estándar SSL y la utilización de autenticación del servidor y del cliente basada en certificados X.509v3. Este software esta accesible públicamente en http://www.modssl.org.

- *Auth_LDAP*. Módulo de acceso al servidor LDAP para el servidor Apache.

  Este módulo permite incorporar la autenticación basada en el servicio LDAP en el servidor Apache. Este software esta accesible públicamente en http://www.rudedog.org/auth_ldap, pueden localizarse más módulos para LDAP en http://modules.apache.org/

4.5 Java

- *Thinking in Java*, Bruce Eckel.

  Libro de programación en Java, accesible públicamente en http://www.bruceeckel.com/TIJ2/index.html.

- *Java Database Programming with JDBC*, Patrick Patel y Karl Moss

  Puede obtenerse más información sobre este libro en Amazon.

- *JDBC Database Access with Java, A tutorial and Annotated Reference*, Graham Hamilton, Rick Catell y Maydene Fisher.

  Puede obtenerse más información sobre este libro en Amazon.

4.6 Integración del servicio de directorio

- *Configuring Netscape roaming with OpenLDAP*, de **Kartik Subbarao**

  Artículo publicado en la revista LinuxWorld, públicamente accesible desde http://linuxworld.com/linuxworld/lw-1999-09/lw-09-ldap-netscape.html

- *LDAP-HOWTO*

  Documento en el cual se recoge la instalación de OpenLDAP en un servidor RedHat Linux y se configura el servicio de directorio como repositorio de la configuración del navegador *Netscape Communicator*, este artículo esta accesible públicamente en http://www.grennan.com/ldap-HOWTO.html

- *Scalable WebMail HOWTO*

  Documento que recoge los pasos necesarios para implementar un servicio de correo electrónico accesible via Web. Utiliza PostFix como MTA, Cyrus como servidor IMAP, IMP como interfaz Web y el servicio de directorio basado en OpenLDAP permite distribuir los buzones entre varios servidores, además de proporcionar el servicio de autenticación. Este documento esta públicamente accesible desde http://horde.org/papers/Scalable_webmail_HOWTO.html

- *Sendmail 8.10*

  La nueva versión de sendamail incluye soporte para LDAP. Las principales características de esta versión pueden obtenerse en: http://www.sendmail.org/8.10.html.

- *nss_LDAP*

  Módulo de validación basado en LDAP para sistemas Solaris y Linux. Este módulo utiliza la tecnología *Nameservice Switch interface* desarrollada originalmente por Sun Microsystems, puede obtenerse en: http://www.padl.com/nss_ldap.html

- *pam_LDAP*

  Módulo de autenticación basado en LDAP para sistemas Solaris y Linux. Este módulo utiliza la tecnología *Pluggable Authentication Module API* definida en *OSF DCE RFC 86.0.* Este módulo permite cambiar la contraseña en el servicio de directorio y además puede crifrar las conexiones con el servidor LDAP utilizando SSL/TSL. Este módulo puede obtenerse en: http://www.padl.com/pam_ldap.html

5 Glosario de términos.

Autenticación, operación por la que se verifica la identidad de una persona o servidor.

Autenticación débil, operación de autenticación basada en el conocimiento de una clave.

Autenticación fuerte, operación de autenticación basada en la posesión de un objeto físico no duplicable y una clave de acceso a la información contenida en dicho objeto, por ejemplo una tarjeta inteligente (smart-card) y el PIN de acceso.

Distinguished Name, Véase Nombre Distinguido.

Directory Information Tree, árbol jerárquico de entradas del directorio.

Esquema, Definición de los tipos de objetos que se pueden almacenar en el directorio, asi como de los atributos de cada clase de objeto y si estos atributos son obligatorios u opcionales.

Nombre Distinguido, Atributo que identifica una entrada en el directorio, esta compuesta por una sucesión de Nombres Distinguidos Relativos.

Nombre Distinguido Relativo, Partes integrantes del Nombre Distinguido, referencian partes del árbol de información del directorio (DIT).

Open source, programa o librería que se distribuye con el código fuente, lo que permite ser modificado para ser adaptado al entorno de producción, en caso de ser necesario.

Relative Distinguished Name, Vease Nombre Distinguido Relativo.

Schema, Vease Esquema.

6 Índice de ilustraciones

7 Anexos

7.1 El servidor de directorio OpenLDAP

OpenLDAP es una iniciativa para desarrollar un servidor LDAP *open source* basado en la distribución original de la Universidad de Michigan.

La distribución contiene los siguientes programas:

Slapd: Es el servidor LDAP, permite la utilización de múltiples tipos de bases de datos:

- ✓ LDBM, base de datos de alto rendimiento.
- ✓ Shell, interfaz con cualquier comando o shell script del sistema operativo.
- ✓ Password, interfaz con un fichero de contraseñas.

Permite la gestión del control de accesos mediante listas de control de accesos.

Funciona en modo multi-thread si el sistema operativo lo permite.

Permite la replicación de las operaciones de actualización siguiendo un esquema maestro/esclavo, muy útil en entornos en los que la fiabilidad y rendimiento de un único servidor ***slapd*** no sean suficientes.

Ldapd: Es una pasarela de LDAP a X.500, este programa se utiliza para permitir a los clientes LDAP consultar el directorio X.500 ya existente y no tiene sentido utilizarlo si el servidor de directorio esta basado en LDAP.

Slurpd: Es un programa que permite al servidor ***slapd*** proporcionar el servicio de replicación de operaciones de actualización. Es el responsable de la distribución de los cambios que se realizan en el servidor *maestro* hacia los servidores *esclavos*. Libera al servidor ***sladp*** de los problemas relacionados con los servidores *esclavos* inaccesibles o caídos ya que ***slurpd*** gestiona automáticamente las actualizaciones fallidas. ***Sladp*** y ***slurpd*** se comunican a través de un fichero de texto que se utiliza como traza de cambios.

Librerías: Las librerías LDAP pueden ser generadas para su enlazado de forma estática y/o dinámica. Las ventajas e inconvenientes derivados de la utilización de cada

uno de los tipos de librerías son los mismos que para cualquier otra librería. Enlazar los programas con librerías estáticas genera programas ejecutables más grandes y tienden a ejecutarse de manera más lenta, además la instalación de una nueva versión de la librerías obliga a recompilar todos los programas que se enlazaron con ella. Las librerías dinámicas permiten generar ejecutables más pequeños, que generalmente se ejecutan más rápido, la actualización de la librería a una versión más moderna hace que automáticamente todos los programas que la utilizan se beneficien en la actualización, en contrapartida, si cambia el interfaz con la librería, se deben recompilar todos los programas que utilizan dicha librería.

7.1.1 Instalación de OpenLDAP

Para instalar el servidor de directorio OpenLDAP hace falta seguir los siguientes pasos:

1. Copiar el software en el servidor. Este software esta públicamente accesible en http://www.openldap.org. Habitualmente hay dos versiones, la distribución *release*, es la distribución original, la distribución *stable*, tiene corregidos los errores que se han podido detectar en la versión *release*.

2. Descomprimir y desempaquetar la distribución.

```
$ gzip –dc openldap-stable.tgz | tar xvf
ldap/
ldap/doc/
ldap/doc/man/
ldap/doc/man/Makefile.in
ldap/doc/man/man1/
…
```

3. Ejecutar el comando *configure* con las opciones necesarias. Para ello es aconsejable ejecutarlo primero con la opción ***–help***.

```
$ ./configure --help
Usage: configure [options] [host]
Options: [defaults in brackets after descriptions]
Configuration:
  --cache-file=FILE       cache test results in FILE
  --help                  print this message
  --no-create             do not create output files
  --quiet, --silent       do not print `checking...' messages
  --version               print the version of autoconf that created configure
Directory and file names:
  --prefix=PREFIX         install architecture-independent files in PREFIX
                          [/usr/local]
  --exec-prefix=EPREFIX   install architecture-dependent files in EPREFIX
```

```
                          [same as prefix]
  --bindir=DIR            user executables in DIR [EPREFIX/bin]
  --sbindir=DIR           system admin executables in DIR [EPREFIX/sbin]
  --libexecdir=DIR        program executables in DIR [EPREFIX/libexec]
  --datadir=DIR           read-only architecture-independent data in DIR
                          [PREFIX/share]
  --sysconfdir=DIR        read-only single-machine data in DIR [PREFIX/etc]
  --sharedstatedir=DIR    modifiable architecture-independent data in DIR
                          [PREFIX/com]
  --localstatedir=DIR     modifiable single-machine data in DIR [PREFIX/var]
  --libdir=DIR            object code libraries in DIR [EPREFIX/lib]
  --includedir=DIR        C header files in DIR [PREFIX/include]
  --oldincludedir=DIR     C header files for non-gcc in DIR [/usr/include]
  --infodir=DIR           info documentation in DIR [PREFIX/info]
  --mandir=DIR            man documentation in DIR [PREFIX/man]
  --srcdir=DIR            find the sources in DIR [configure dir or ..]
  --program-prefix=PREFIX prepend PREFIX to installed program names
  --program-suffix=SUFFIX append SUFFIX to installed program names
  --program-transform-name=PROGRAM
                          run sed PROGRAM on installed program names
Host type:
  --build=BUILD           configure for building on BUILD [BUILD=HOST]
  --host=HOST             configure for HOST [guessed]
  --target=TARGET         configure for TARGET [TARGET=HOST]
Features and packages:
  --disable-FEATURE       do not include FEATURE (same as --enable-FEATURE=no)
  --enable-FEATURE[=ARG]  include FEATURE [ARG=yes]
  --with-PACKAGE[=ARG]    use PACKAGE [ARG=yes]
  --without-PACKAGE       do not use PACKAGE (same as --with-PACKAGE=no)
  --x-includes=DIR        X include files are in DIR
  --x-libraries=DIR       X library files are in DIR
--enable and --with options recognized:
  --with-subdir=DIR change default subdirectory used for installs
  --enable-debug        enable debugging (yes)
  --enable-proctitle    enable proctitle support (yes)
  --enable-libui        enable library user interface (yes)
  --enable-cache        enable caching (yes)
  --enable-dns          enable dns support (no)
  --enable-referrals    enable referrals (yes)
  --enable-cldap        enable connectionless ldap (no)
  --enable-x-compile    enable cross compiling (no)
  --enable-dmalloc      enable debug malloc support (no)
  --with-kerberos       use Kerberos (auto)
  --with-threads        use threads (auto)
  --with-yielding-select        with implicitly yielding select (auto)
LDAPD Options:
  --enable-ldapd        enable building ldapd (no)
SLAPD Options:
  --enable-slapd        enable building slapd (yes)
    --enable-aclgroups  enable ACL group support (auto)
    --enable-cleartext  enable cleartext passwords (yes)
    --enable-crypt      enable crypt(3) passwords (auto)
    --enable-wrappers   enable tcp wrapper support (no)
    --enable-phonetic   enable phonetic/soundex (no)
    --enable-rlookups   enable reverse lookups (auto)
    --enable-ldbm       enable ldbm backend (yes)
      --with-ldbm-api   use LDBM API (auto)
      --with-ldbm-type  use LDBM type (auto)
    --enable-passwd     enable passwd backend (no)
--enable-shell      enable shell backend (no)
SLURPD Options:
  --enable-slurpd       enable building slurpd (auto)
Library Generation & Linking Options
  --enable-static[=PKGS]  build static libraries [default=yes]
  --enable-shared[=PKGS]  build shared libraries [default=no]
  --enable-fast-install[=PKGS]  optimize for fast installation [default=yes]
  --with-gnu-ld           assume the C compiler uses GNU ld [default=no]
  --disable-libtool-lock  avoid locking (might break parallel builds)
```

Se han marcado las opciones que se han considerado interesantes. La mayoría de ellas aparecen activadas por defecto, con la excepción de las generación de las librerías compartidas o dinámicas (shared).

4. Una vez configurado, se comprueban las dependencias y se procede a compilar el servidor.

```
$ make depend
$ make
```

5. Tras la compilación del servidor, procederemos a comprobar que los programas generados funcionan de modo correcto

```
$ cd tests
$ make
```

6. Si todo ha ido correctamente, se puede proceder a la instalación del servidor

```
$ su
Password:
# make install
```

Para más información pueden consultarse los ficheros *README* e *INSTALL* que acompañan a la distribución y la documentación sobre el proceso de instalación en el servidor Web del proyecto OpenLDAP (http://www.openldap.org).

7.1.2 Configuración de OpenLDAP

La configuración del servidor OpenLDAP se almacena en el fichero `<directorio_instalacion>/etc/openldap/slapd.conf`. Este fichero contiene información para el servidor **slapd**, pero también es utilizado por **slurpd**, programa encargado de las réplicas, y por los programas de indexación LDBM (**ldif2ldbm**, **ldif2index**, **ldif2id2entry** y **ldif2id2children**.

El fichero slapd.conf contiene una serie de directivas globales, que se aplican al servidor slapd y todas las bases de datos definidas, seguidas de definiciones de bases de datos específicas.

El formato general del fichero tiene el siguiente aspecto:

```
# comment - these options apply to every database

<global configuration options>

# first database definition & configuration options

database    <backend 1 type>

<configuration options specific to backend 1>

# subsequent database definitions & configuration options

...
```

Pueden incluirse tantas secciones específicas para bases de datos como sean necesarias. Las opciones incluidas en estas secciones tienen prioridad sobre las opciones generales.

Las líneas en blanco y las que comiencen por # son tratadas como comentarios.

Las lineas que comiencen con un espacio en blanco son tratadas como continuación de la línea anterior.

7.1.2.1 Opciones globales de configuración

Las opciones descritas en esta sección se aplican a todos los servicios de base de datos, salvo que sean redefinidas en una definición específica.

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

Esta opción permite el acceso (especificado en `<accesslevel>`) al conjunto de entradas y/o atributos (especificados en `<what>`) por los solicitantes (especificados en `<who>`).

```
attribute <name> [<name2>] { bin | ces | cis | tel | dn }
```

Esta opción asocia una sintaxis con un nombre de atributo. Por defecto se asume que la sintaxis de un atributo es cis.

```
defaultaccess { none | compare | search | read | write }
```

Esta opción especifica los permisos que se asignan cuando no hay ninguna coincidencia en las listas de control de accesos. Un determinado nivel de acceso contiene los niveles inferiores. Por defecto es read.

```
include <filename>
```

Esta opción permite indicar a ***slapd*** que debe leer el fichero especificado como parte de la configuración. Esta opción suele emplearse para incluir los ficheros `slapd.at.conf` y `slapd.oc.conf`, que contienen las especificaciones de los tipos de datos de los atributos (utilizando la opción attribute) y las definiciones de las reglas del esquema (utilizando la opción objectclass)

```
Loglevel <integer>
```

Esta opción permite indicar el nivel de traza que se desea (utilizando el programa ***syslogd*** y la utilidad LOG_LOCAL4). Los niveles de traza son aditivos y los valores posibles son:

Nivel	*Significado*
1	Traza llamadas a función
2	Depuración la gestión de los paquetes
4	Depuración de muy bajo nivel
8	Gestión de conexiones
16	Muestra los paquetes recibidos y enviados
32	Procesamiento de los filtros de búsqueda
64	Procesamiento del fichero de configuración
128	Procesamiento de las listas de control de acceso
256	Estadísticas de conexiones/operaciones/resultados
512	Estadísticas de entradas enviadas
1024	Muestra la comunicación con los backends shell
2048	Muestra el análisis de la depuración de las entradas

Ilustración 32 Niveles de traza para el programa ***slapd***

```
objectclass <name> [ requires <attrs> ] [ allows <attrs> ]
```

Esta opción define las reglas del esquema para la clase definida. Esta opción se utiliza en conjunción con la opción schemacheck.

```
referral <url>
```

Esta opción permite especificar la referencia a devolver a un cliente cuando ***slapd*** no pueda localizar una base de datos local para completar la petición.

```
schemacheck { on | off }
```

Esta opción activa el control de las operaciones de actualización para que los resultados sean conformes al esquema. Por defecto esta desactivada (off).

```
sizelimit <integer>
```

Esta opción permite limitar el número máximo de entradas que el servidor ***slapd*** devolverá en una operación de búsqueda. Por defecto es 500.

```
timelimit <integer>
```

Esta opción especifica el tiempo máximo en segundos (en tiempo real) que el servidor ***slapd*** empleará para responder a una petición. Si no ha completado la petición, devolverá un mensaje indicando que el tiempo máximo ha sido excedido.

7.1.2.2 Opciones específicas de la base de datos LDBM

Estas opciones solo se aplican a las definiciones de bases de datos LDBM, esto es deben ir después de una linea `database ldbm` y antes de la definición de una nueva base de datos.

```
cachesize <integer>
```

Esta opción indica el número de entradas que el servidor ***slapd*** debe mantener en memoria, por defecto 1000.

```
dbcachesize <integer>
```

Esta opción especifica el número de bytes de memoria que se asignarán a la cache de los ficheros de índices abiertos. Aumentar el tamaño aumenta los requerimientos de memoria, pero aumenta drásticamente el rendimiento, especialmente durante las modificaciones y la creación de los índices.

```
directory <directory>
```

Esta opción especifica en que directorio se van a almacenar los ficheros de la base de datos.

```
index { <attrlist> | default } [ pres, eq, aprox, sub, none ]
```

Esta opción especifica los índices a mantener para un atributo determinado si solo se indica <attrlist>, todos los índices son mantenidos.

```
mode <integer>
```

Indica el modo de los ficheros de base de datos. Por defecto es 0600

7.1.2.3 Opciones para otras bases de datos

Los interfaces de base de datos adiciones (shell y password) no son utilizados generalmente. Para más información sobre el desarrollo de dichos interfades y la configuración de ***slapd*** para su utilización pueden encontrarse en *The SLAPD and SLURPD Administrator's Guide*.

7.1.2.4 Control de accesos

Los accesos a las entradas y atributos de ***slapd*** son controlados por las directivas access incluidas en el fichero de configuración. El formato de dichas directivas es el siguiente:

```
<access directive> ::= access to <what> [by <who> <access>]+
<what>             ::= * | [ dn=<regexp> ] [ filter=<ldapfilter> ] [ attrs=<attrlist> ]
<who>              ::= * | self | dn=<regexp> | addr=<regexp> | domain=<regexp> |
                       dnattr=<dn attribute>
<access>           ::= [self]none | [self]compare | [self]search | [self]read |
[self]write
```

Esta directiva permite el establecimiento de listas de control de acceso, conocidas como ACL, para establecer que operaciones puede realizar un cliente.

La parte <what> de la especificación de acceso determina a que entrada y/o atributo se aplicará dicha especificación. Las entradas se pueden especificar mediante una expresión regular que concuerde con el nombre distinguido de la entrada.

```
dn=.”*,o=Universidad Carlos III,c=es”
```

También se pueden seleccionar mediante un filtro de búsqueda sobre algún atributo

```
filter=”objectclass=uc3mTrabajador”
```

El selector “*” es un selector especial que permite seleccionar todas las entradas y es equivalente a “dn=.*”.

Los atributos de una entrada pueden ser seleccionados incluyendo la lista de atributos separados por comas.

```
attrs=mail,cn,sn
```

Para permitir el acceso a la entrada, se utiliza el nombre especial entry. Para poder modificar un atributo de una entrada no basta con tener permiso para acceder al atributo, es necesario tener permiso para acceder a la entrada.

La parte <who> identifica las entidades que tendrán permitido el acceso. Las entidades pueden especificarse mediante el identificador “*” concuerda con cualquier entrada, el identificador “self” concuerda con la entrada protegida en la parte <what>, o mediante una expresión regular que concuerde con el nombre distinguido.

```
dn=<regular expression>
```

Las entidades también pueden especificarse en función de la dirección IP del cliente o de su nombre de dominio.

```
addr=<regular expression>
domain=<regular expression>
```

La especificación dnattr permite proporcionar permisos de acceso a cualquier entidad cuyo nombre distinguido este en un atributo de la entrada (por ejemplo permitir darse de baja a una persona de una lista de distribución)

```
dnattr=dn-valued attribute name>
```

La parte <access> asigna los permisos de acceso, los niveles son incrementales, luego asignar el permiso write implica read, search y compare.

Cuando se evalúa una petición, primero se comprueban las directivas específicas de la base de datos en la que se encuentra la entrada, en el orden en el que aparecen en el fichero slapd.conf. En caso de ser no ser satisfactoria, se comprueban las directivas generales, en el orden en que aparecen en el fichero slapd.conf.

A continuación se muestran algunos ejemplos de directivas de control de acceso

```
access to * by * read
```

Esta directiva permite el acceso en lectura al cualquier entidad.

```
access by dn=”.*,o=Universidad Carlos III,c=ES”
    by * search
access to dn=”.*,c=ES”
    by * read
```

Esta directiva permite el acceso en lectura a las entradas bajo el árbol c=ES, excepto para aquellas bajo el subárbol “o=Universidad Carlos III,c=ES”, para los que solo se permite el acceso en búsqueda. Debe advertirse que si las directivas estuviesen en orden inverso, la directiva para las entradas de la Universidad Carlos III no sería comprobada nunca, ya que todas sus entradas se encuentran dentro del subárbol c=ES.

```
access to dn=”.*,o=Universidad Carlos III,c=ES” attr=homePhone
    By self                                          write
    By dn=”.*o=Universidad Carlos III,c=ES”          search
    By domain=.*\.uc3m\.es                           read
    By *                                             compare
access to dn=”.*, o=Universidad Carlos III,c=ES”
    By self                                          write
    By dn=”.*o=Universidad Carlos III,c=ES”          search
    By *                                             none
```

En este caso, la directiva se aplica a las entradas bajo “o=Universidad Carlos III,c=ES”. Todos los atributos excepto homePhone pueden ser escritos por la entidad propietaria, otras entidades de la Universidad pueden realizar búsquedas y el resto de las entidades no tienen permiso para acceder.

7.1.3 Programas de acceso al directorio

7.1.3.1 Ldapmodify y Ldapadd

```
ldapmodify [-a] [-b] [-c] [-r] [-n] [-v] [-k]
     [-d debuglevel] [-D binddn] [-W] [-w passwd] [-h ldaphost]
     [-p ldapport] [-f file]
ldapadd [-b] [-c] [-r] [-n] [-v] [-k] [-K] [-d debuglevel]
     [-D binddn] [-w passwd] [-h ldaphost] [-p ldapport]
     [-f file]
```

Estos programas permiten agregar y/o modificar entradas en el directorio. Las siguientes opciones están soportadas:

Opción	*Significado*
-a	Añadir entradas. Por defecto ldapmodify modifica entradas ya existentes.
-b	Los atributos que empiezan por / son tratados como datos binarios contenidos en el fichero especificado
-c	Modo continuo, si se produce un error al realizar el alta, se continua procesando las siguientes entradas.
-r	Por defecto, reemplaza los atributos de las entradas.
-n	Muestra lo que se va a hacer, pero sin hacerlo.
-v	Modo verbose.
-f fichero	Lee las entradas LDIF del fichero especificado en lugar de la entrada estándar.
-D bindDN	nombre distinguido con el que se realizará la operación bind.
-W	Pide la contraseña para realizar la operación bind.
-w contraseña	Contraseña con la que realizar la operación bind.

Opción	*Significado*
-h servidor	Nombre o dirección IP del servidor LDAP
-p puerto	Puerto en el que escucha el servidor LDAP

Ilustración 33 Opciones de los programas **ldapadd** y **ldapmodify**

7.1.3.2 Ldapdelete

```
ldapdelete [-n] [-v] [-k] [-K] [-c] [-d debuglevel]
     [-f file] [-D binddn] [-W] [-w passwd] [-h ldaphost]
     [-p ldapport] [dn]...
```

Este programa permite borrar entradas del directorio.

Opción	*Significado*
-n	Muestra lo que se va a hacer, pero sin hacerlo.
-v	Modo verbose.
-f fichero	Lee las entradas LDIF del fichero especificado en lugar de la entrada estándar.
-D bindDN	Nombre distinguido con el que se realizará la operación bind.
-W	Pide la contraseña para realizar la operación bind.
-w contraseña	Contraseña con la que realizar la operación bind.
-h servidor	Nombre o dirección IP del servidor LDAP
-p puerto	Puerto en el que escucha el servidor LDAP
Dn	Nombre distinguido de la entrada a borrar.

Ilustración 34 Opciones de los programas **ldapdelete**

7.1.3.3 Ldapsearch

```
ldapsearch [-n] [-u] [-v] [-k] [-K] [-t] [-A] [-B] [-L]
       [-R] [-d debuglevel] [-F sep] [-f file] [-D binddn] [-W]
       [-w bindpasswd] [-h ldaphost] [-p ldapport] [-b search-
       base] [-s base|one|sub] [-a never|always|search|find]
       [-l timelimit] [-z sizelimit] filter [attrs...]
```

Este programa permite realizar búsquedas en el directorio LDAP.

Opción	*Significado*
-n	Muestra lo que se va a hacer, pero sin hacerlo.
-u	Incluye el nombre distinguido en forma más amigable.
-v	Modo verbose.
-t	Guarda los valores de cada atributo en un fichero temporal. Útil para obtener atributos no-ASCII como *jpegPhoto* o *audio*.
-A	Solo recupera los atributos (no los valores asociados a los atributos). Útil para determinar si una entrada tiene definido un valor para un determinado atributo.
-B	Mostrar los atributos no-ASCII. Implica la opción ***-L***
-L	Mostrar los resultado en formato LDIF. Implica la opción ***-B*** y deshabilita la opción ***-F***
-R	No seguir automáticamente las referencias o *referrals*.
-F sep	Utiliza *sep* como separador entre los nombres de los atributos y los valores. El separador por defecto es ‘=’.
-S atributo	Ordena las entradas en función del *atributo*. Esta opción obliga a recuperar todas las entradas que cumplen la condición de búsqueda, ordenarlas y luego mostrarlas.
-f fichero	Lee las lineas del fichero especificado y realiza una búsqueda por cada línea leída.

Opción	*Significado*
`-D bindDN`	Nombre distinguido con el que se realizará la operación bind.
`-W`	Pide la contraseña para realizar la operación bind.
`-w contraseña`	Contraseña con la que realizar la operación bind.
`-h servidor`	Nombre o dirección IP del servidor LDAP
`-p puerto`	Puerto en el que escucha el servidor LDAP.
`-b base`	Especifica el punto de partida para la realización de la búsqueda.
`-s tipo`	Especifica el tipo de búsqueda que se va a realizar, puede ser *base*, *one*, o *sub*, para especificar un objeto, un nivel o búsqueda en el subárbol. Por defecto es *sub*.
`-l tiempo`	Especifica el tiempo máximo de espera para realizar la búsqueda. Si *tiempo* es 0, no hay limitación. El servidor LDAP puede imponer limitaciones de tiempo para la realización de una búsqueda, que solo el usuario *root* puede exceder.
`-z tamaño`	Especifica el número máximo de entradas que se desean recuperar. Si el *tamaño* es 0, no hay limitación. El servidor LDAP puede imponer limitaciones al número de entradas a recuperar, que solo el usuario *root* puede exceder.
`Filtro`	Filtro de búsqueda.
`Atributos`	Lista de atributos que deben ser recuperados. Si no se especifica, se recuperan todos los atributos.

Ilustración 35 Opciones de los programas **ldapsearch**