TEMA 5

Contenido

Servicios de red implicados en el despliegue de una aplicación web

Caso práctico

*En BK Programación, como cada lunes, tiene lugar una reunión entre **Ada**, la directora, **María**, la responsable del Área de Sistemas y **Juan**, el responsable del Área de Desarrollo, en la que se evalúan el estado actual de los proyectos en desarrollo o a desarrollar. En este caso, la reunión fue la siguiente:*

—He estado viendo las tareas relativas al proyecto en la aplicación de proyectos Redmine y veo que llevamos adelanto sobre lo previsto.

*—Sí, **Ada** —dijo **María—**, ya tenemos el proyecto encauzado, solamente quedan por resolver dos tareas: configurar la visibilidad a través de Internet para la aplicación web y la autentificación de usuarios. La verdad es que ha sido todo un acierto el empleo de Redmine para llevar a buen fin el proyecto. Se deja manejar muy bien y permite coordinarse. Ahora la verdad es que tenemos un control exhaustivo sobre el proyecto.*

*—Sí —reafirmó **Juan—**. La verdad es que el poder monitorizar las tareas y ver su ciclo de vida es todo un acierto. A todo esto, entonces, ¿cuándo podemos empezar con las pruebas en el servidor definitivo? -preguntó **Juan**-.*

*—Pues, pronto —dijo **María—**. Tan pronto como tengamos realizada la primera de las dos tareas, esto es, tan pronto como tengamos configurado la redirección DNS de la aplicación web al servidor destinado al proyecto.*

*—Bien, —dijo **Juan—**. Entonces iré ya creando las tareas respectivas en el Redmine, asignándolas a los responsables correspondientes, para las pruebas.*

*—Sí, estaría bien, ya que lo que nos estaba reteniendo en la tarea relativa al DNS era la elección del dominio DNS por parte del cliente, puesto que no lo tenía claro. Una vez resuelto, solamente debemos configurar el servidor DNS para apuntar el nombre de dominio DNS a la IP del servidor destinado al proyecto y verificar que la configuración se replica en el servidor esclavo. Por lo tanto a configurar el servidor DNS **BIND** y listo.*

*—Por cierto, ¿cuál es el sistema de autenticación elegido por el cliente? —preguntó **Juan—**.*

*—Ha elegido autentificación por LDAP -dijo **María**-. Al final se ha decidido por el montaje de un servidor LDAP frente a la otra opción considerada: una base de datos SQL. Así que lo configuraremos con **OpenLDAP**.*

*—Muy bien —dijo **Ada—**, veo que el proyecto va viento en popa, esperemos que continúe.*

*—Por mi parte -dijo **María**-, lo único que podría ralentizar el proyecto sería el envío de usuarios para darlos de alta en OpenLDAP, por lo demás...*

—Bien, si hasta ahora el cliente ha sido efectivo en plazos no hay porque suponer que no siga continuando siéndolo. En fin, manos a la obra. Volvemos a quedar la próxima semana a la misma hora, la sala ya está reservada, y comentamos de nuevo.

*—Vale, adiós —dijo **María—**.*

*—Hasta luego —dijo **Juan—**.*

1.- Servidores de nombres de dominio.

Caso práctico
*Para poder llevar a buen fin el proyecto, **María** se puso manos a la obra y determinó el siguiente escenario de trabajo para la realización de las dos últimas tareas del proyecto:*
- ✓ *Sistema Operativo Servidor: Debian GNU/Linux 6.0.*
- ✓ *Servidor Web: Apache (apache2).*
- ✓ *Servidor DNS Primario (Maestro): BIND (BIND9).*
- ✓ *Servidor DNS Secundario (Esclavo): BIND (BIND9).*
- ✓ *Servidor LDAP: OpenLDAP.*
- ✓ *Configuración de Red:*
  - ➔ *Servidor Web: 192.168.200.250.*
  - ➔ *Cliente de pruebas (desde donde se lanza el navegador): 192.168.200.100.*
  - ➔ *Servidor DNS Maestro: 192.168.200.250.*
  - ➔ *Servidor DNS Esclavo: 192.168.200.249.*
  - ➔ *Servidor OpenLDAP: 192.168.200.248.*

***María**, debe garantizar el correcto funcionamiento de la resolución DNS, por lo tanto, como con otros proyectos, prevé la redundancia del servicio mediante dos servidores DNS, uno actuando de primario y otro de secundario, y debido a sus características se ha decantado por el servidor DNS BIND.*
*Para la autentificación de usuarios, el cliente tras explicarle las alternativas se ha decantado por LDAP, por lo cual **María** configurará el servicio mediante OpenLDAP debido a sus características.*

¿Alguna vez te has parado a pensar qué es lo que pasa desde que escribes una dirección URL (*Dirección de Internet de un recurso, válida para su posible utilización a través de Internet, la cual permite que el navegador la encuentre y la muestre de forma adecuada*) en el navegador hasta que puedes ver la página web cargada? ¿Sería posible acordarse de las páginas si tuviésemos que navegar a través de IP y no pudiéramos navegar a través de nombres? ¿Qué es lo que pasa si cambiásemos la redirección DNS a otro servidor? ¿Es automático el cambio? ¿Cuánto tiempo tarda? ¿Qué tiempo se necesita para activar los nuevos cambios?…

Internet funciona mediante el protocolo TCP/IP (*el Transfer Control Protocol garantiza que los datos serán entregados en su destino sin errores y una vez recogidos ponerlos en el mismo orden en que se transmitieron*), efectuando conexiones mediante IP. ¿Qué quiere esto decir? Pues, que cada host (*dispositivo conectado a una red, que pueda proveer y utilizar servicios de ella*) en Internet se identifica mediante una IP, así es lo mismo visitar la página **http://www.rediris.es** que **http://130.206.13.20**

Entonces, ¿sería posible visitar cada página web conociendo su IP? Efectivamente, sólo que los seres humanos estamos más acostumbrados, a diferencia de las máquinas, a recordar nombres y no números. ¿Qué te es más fácil recordar el DNI de una persona o su nombre y apellidos?. Por lo cual, debe existir algo que nos traduzca los nombres a IPs o viceversa. Sí, por supuesto, este algo no es otro que el **servidor DNS** o un archivo de texto, típicamente denominando `hosts`, como el archivo `/etc/hosts` en sistemas GNU/Linux.

¿Pueden convivir en una misma máquina un servidor DNS y el archivo `/etc/hosts`? Pues, sí. Pero hay que tener en cuenta la preferencia. Así, en caso de coexistir, primero se intentará la resolución IP/Nombre mediante el archivo `/etc/hosts` y, en caso de no encontrar correspondencia, actuará el servidor DNS.

El fichero `/etc/hosts` permite alias de nombres de dominios, esto es, una misma IP puede apuntar a nombres distintos. Cada línea del fichero comenzará con una IP y en la misma línea, separados por espacios o tabuladores, puedes escribir los nombres de dominios correspondientes. El primer nombre, el más cercano a la IP, es considerado el principal, los demás son alias de éste.

1.1.- Sistema de nombres de dominio.

¿Cuántos servidores DNS existen? ¿Cuántas redirecciones DNS son posibles? ¿Existe un servidor DNS donde se guarden todos los dominios DNS posibles en Internet? ¿Qué son los servidores DNS Raíz?

¿Es necesario configurar un servidor DNS o se puede hacer la redirección mediante archivos de textos? Para la redirección deberá existir un **servidor DNS** que las resuelva o bien, en su defecto o a mayores, deberán existir las entradas correspondientes en el fichero del sistema local `/etc/hosts`. En caso de coexistir, primero se prueba la resolución en el fichero y luego en el servidor.

Entonces, ¿para qué montar un servidor si simplemente escribiendo en un fichero la relación IP/Nombre el sistema ya funcionaría? Pues, realmente depende, ya que si estás pensando en pocos equipos a resolver el nombre de dominio la simplicidad del fichero `/etc/hosts` te permitiría no tener que montar un servidor, pero si el número de equipos que deben resolver el nombre en IP es elevado, el sistema del fichero es complicado de mantener y deberías pensar en montar un servidor DNS.

La complejidad radica en que en el fichero `/etc/hosts` los cambios son estáticos, así, para actualizar o activar un nuevo cambio debe editarse en todos los ficheros `/etc/hosts` implicados. Esto es, supón que posees 20 equipos que quieren resolver una página web, por ejemplo `www.debian.org` el procedimiento sería aproximadamente el siguiente:

1. Se escribe la página web en cada equipo en la barra de direcciones del navegador.
2. Se traduce el nombre DNS a una IP. ¿Cómo se produce esto? Pues, ahí está el quid de la cuestión: o bien mediante servidores DNS, o bien mediante ficheros estáticos `/etc/hosts`, con lo cual se debe modificar este fichero en cada cliente. Y esto, como bien puedes pensar, se hace arduo de manejar.

Pero, ¿y si la resolución tiene lugar mediante servidores DNS?, ¿y por qué servidores DNS y no servidor DNS? Bien, existe, a modo de resumen, un procedimiento de resolución DNS, más o menos, similar al siguiente (encontrarás el procedimiento exacto un poco más adelante):

- ✓ Primero, se debe averiguar que servidor DNS resuelve el domino raíz '`org`' a una IP.
- ✓ Segundo, una vez obtenida esa IP que gobierna el dominio raíz '`org`', se le pregunta por la IP del servidor DNS que gobierna el subdominio '`debian`' bajo '`org`'.
- ✓ Tercero, una vez obtenida la IP del servidor DNS que gobierna el dominio '`debian.org`' se le pregunta por la IP del equipo '`www.debian.org`'

Pero, entonces: ¿cuántos servidores DNS existen a la hora de preguntar? ¿existe un número limitado de redirecciones de consultas? ¿y, si se vuelve a hacer la misma consulta, hay que repetir el proceso?. Bien, pues no existe un número limitado de redirección de consultas, lo que sucede es que las consultas se van escalando hasta encontrar un servidor DNS que las resuelva, y escalando y escalando puede ser que las consultas se resuelvan en los últimos servidores DNS a los cuales se puede preguntar: los servidores raíz.

Pero, puede ser que no sea necesario escalar las consultas, puesto que todos los servidores DNS son servidores caché, lo que significa que recuerdan las consultas efectuadas. Por lo tanto, si se hace una consulta que ya está guardada en la caché, la respuesta es casi instantánea y ya ha sido resuelta. Es más, los equipos clientes, desde donde se hace la consulta a través del navegador como se indicaba en el ejemplo, también poseen una memoria caché DNS, de tal forma que anteriormente a preguntar al servidor DNS, se mira en la caché del propio sistema operativo, y si se obtiene la respuesta el proceso se ha acabado.

El sistema DNS en realidad es una base de datos distribuida, que permite la administración local de segmentos que juntos componen toda la base de datos local. Los datos de cada segmento están disponibles para toda la red a través de un esquema cliente-servidor jerárquico.

Según lo expuesto, y si en tu configuración de red del sistema operativo solamente posees un servidor DNS, entonces: ¿cuál sería el proceso para encontrar la IP de la dirección web: `http://www.debian.org/distrib/netinst`?

El proceso sería el siguiente:

1. Se consulta la memoria caché del sistema operativo: si ya existe la resolución a IP el proceso ha terminado, sino el proceso continúa en el paso 2.
2. Se consulta la memoria caché del servidor DNS que tengas configurado en la configuración de red del sistema operativo: si ya existe la resolución a IP el proceso ha terminado, sino el proceso continúa en el paso 3.
3. Se averigua qué servidor DNS resuelve el domino raíz '`org`' a una IP.
4. Una vez obtenida la IP que gobierna el dominio raíz '`org`', se le pregunta por la IP del servidor DNS que gobierna el subdominio '`debian`' bajo '`org`'.
5. Por último, una vez obtenida la IP del servidor DNS que gobierna el dominio '`debian.org`', se le pregunta por la IP del equipo '`www.debian.org`', y el proceso ha terminado.

Te proponemos que hagas un viaje por la siguiente página web donde se documenta los servidores raíz DNS.

http://www.root-servers.org/

1.1.1.- ¿Cómo es un nombre de dominio?

¿Qué es lo que sueles escribir en la barra de direcciones URL del navegador? Normalmente algo similar a: **www.debian.org** . Entonces, vienen siendo unos caracteres separados por puntos. ¿Qué es lo que significan esos puntos? ¿Qué dividen? Además, en el ejemplo expuesto, al escribir **www.debian.org** el navegador autocomplementa esta petición a **http://www.debian.org** , ¿por qué?

Todas estas preguntas tienen respuesta, así que vamos a por ellas:

- ✓ Primero: Los puntos separan dominios y subdominios, empezando de derecha a izquierda tendrás dominios de primer nivel y dominios de segundo, tercer, …, n-ésimo nivel, denominados subdominios. Así:
  - ➔ `org` es el dominio de primer nivel que identifica a organizaciones.
  - ➔ `debian` es un subdominio, en este caso dominio de segundo nivel bajo `org`, que identifica al nombre de la organización o al nombre de la empresa, sucursal, etc.
  - ➔ `www` es un subdominio, en este caso dominio de tercer nivel bajo debian, que identifica al equipo donde está colgada la página web, esto es, identifica el servidor web que aloja la página web. Es el dominio `www` que el servidor DNS redirecciona a la IP del servidor web.
- ✓ Segundo: `http://` es el protocolo de hypertexto que permite la correcta visualización de la página web en el navegador. Es lo que el navegador autocomplementa en caso de no estipular uno propio en la barra de direcciones URL con el nombre de dominio.

Los dominios de primer nivel identifican el tipo de página web que solicitas o bien la localización de la misma, por ejemplo:

- ✓ `net` identifica redes.
- ✓ `com` identifica comercio.
- ✓ `es` identifica localización España.
- ✓ `tk` identifica localización Tokelau.

Esto suele ser lo común, más no es obligatorio, es decir, si una empresa posee un dominio `com` puede dedicarse al sector de redes de comunicaciones y no poseer el dominio `net`, así como puede ser una empresa localizada en España y no poseer el dominio `es`.

A nivel gramatical los dominios deben cumplir una serie de requisitos. Por ejemplo:
- ✓ Sólo pueden estar **compuestos** de **letras** (alfabeto inglés), **números** y **guiones** ("-").
- ✓ **No** pueden **empezar** o **terminar** por **guiones**.
- ✓ Tienen que tener **menos de 63 caracteres** sin incluir la extensión, y más de uno o dos dependiendo del dominio de primer nivel.

Ahora bien, hoy día ya es posible registrar dominios con caracteres de otras lenguas no inglesas, como la ñ o la ç. Estos dominios se denominan **multilingües**.

La sintaxis de los nombres de dominio se discute en varios RFC (*Request for Comments. Serie de documentos en los que se detalla prácticamente todo lo relacionado con la tecnología de la que se sirve Internet: protocolos, recomendaciones, comunicaciones…*): RFC 1035, RFC 1123 y RFC 2181.

1.1.2.- Jerarquía de nombres de dominio.

El espacio de nombres de dominio (*el universo de todos los nombres de dominio*) está organizado de forma jerárquica. El nivel más alto en la jerarquía es el dominio raíz, que se representa como un punto (".") y el siguiente nivel en la jerarquía se llama dominio de nivel superior (**TLD**). Sólo hay un dominio raíz, pero hay muchosTLDs y cada TLD se llama dominio secundario del dominio raíz. En este contexto, el dominio raíz es el dominio principal, ya que está un nivel por encima de un TLD y cada TLD, a su vez, pueden tener muchos dominios hijos. Los hijos de los dominios de nivel superior se llaman de segundo nivel, los del segundo nivel se llaman de tercer nivel, los del tercer nivel de cuarto, y así sucesivamente.

Por lo tanto el DNS, organiza los nombres de máquina (**hostname**) en una jerarquía de dominios separados por el carácter punto '.'. Un **dominio** es una colección de nodos relacionados de alguna forma (*porque están en la misma red, tal como los nodos de una empresa*). Por ejemplo:

```
rrhh.departamento.empresa.org
marketing.departamento.empresa.org
contabilidad.consultas.empresa.org
```

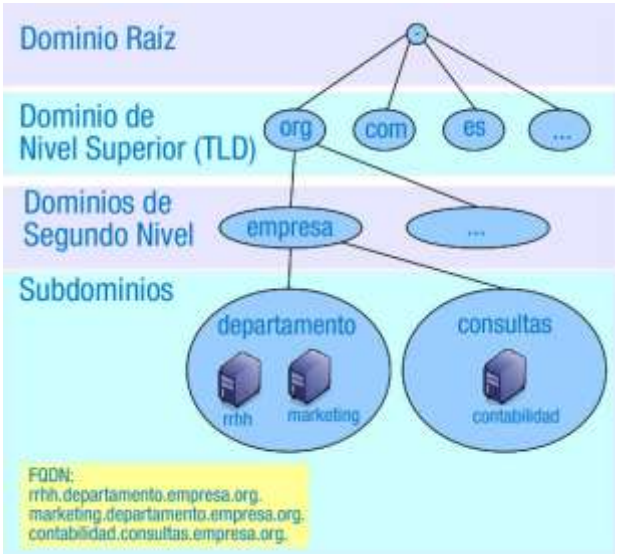
Donde:
- ✓ La empresa agrupa sus nodos en el dominio de primer nivel "`org`". Éste es un **TLD**.
- ✓ La empresa tiene un subdominio, dominio de segundo nivel "`empresa`" bajo "`org`". Así "`empresa`" es un dominio de segundo nivel, hijo del TLD "`+`".
- ✓ A su vez puedes encontrar nuevos subdominios dentro, en este caso: "`departamento`" y "`consultas`". Es decir, dominios de tercer nivel, hijos a su vez del dominio de segundo nivel "`empresa`".
- ✓ Finalmente, un nodo que tendrá un nombre completo conocido como totalmente cualificado o **FQDN**, que es la concatenación de: TLD, dominio de segundo nivel, dominio de tercer nivel, etc., tal como:

```
rrhh.departamento.empresa.org, marketing.departamento.empresa.org, contabilidad.consultas.empresa.org.
```

También es posible tener un dominio de cuarto nivel, dominio de quinto nivel, y así sucesivamente.

En la siguiente figura puedes ver una parte del espacio de nombres. La raíz del árbol, que se identifica con un punto sencillo, es lo que se denomina dominio raíz y es el origen de todos los dominios. Para indicar que un nombre es FQDN, a veces se termina su escritura en un punto, aunque por lo general se omite. Este punto significa que el último componente del nombre es el dominio raíz. Así, por ejemplo en el nombre de dominio:

```
rrhh.departamento.empresa.org.
```



El símbolo del dominio raíz es el punto situado más a la derecha del nombre del dominio.

Sólo hay una raíz de dominio, pero hay más de 250 dominios de nivel superior, clasificados en los siguientes tres tipos:

- ✓ `TLD` de código de país (`ccTLD`): dominios asociados con **países** y territorios. Hay más de 240 ccTLD. Están formados por **2 letras**, por ejemplo: `es`, `uk`, `en`, y `jp`.
- ✓ Dominios de nivel superior genéricos (`gTLD`): están formados por **3 o más letras**. A su vez se subdividen en:
  - ➔ Dominios de internet patrocinados (`sTLD`): representan una comunidad de intereses, es decir, detrás existe una organización u organismo público que propone el dominio y establece las reglas para optar a dicho dominio. Por ejemplo: `edu`, `gov`, `int`, `mil`, `aero`, `museum`.
  - ➔ Dominios de internet no patrocinados (`uTLD`). Sin una organización detrás que establezca las reglas para optar a dicho dominio. La lista de gTLD incluye: `com`, `net`, `org`, `biz`, `info`.

En el siguiente enlace puedes encontrar una lista actualizada de los dominios de primer nivel existentes.

http://data.iana.org/TLD/tlds-alpha-by-domain.txt

1.2.- Ventajas del DNS.

"¿Qué sabe el pez del agua donde nada toda su vida?"

Albert Einstein

¿Qué pasaría si dispones de 20 equipos y en todos actualizas una entrada DNS en el fichero `/etc/hosts`, salvo en 3 de ellos? Sí, esos tres quedarían no actualizados. ¿Y si en la próxima actualización el cambio no se replica en otros 3, que pueden ser los mismos o no? ¿Y en la próxima …?

Bien, parece que el sistema de modificar el archivo `/etc/hosts` no parece muy buena idea, puesto que al ser cambios estáticos, más de un cambio puede quedar en el tintero, obteniendo al final un sistema no homogéneo. Así, parece claro que la solución, para obtener un sistema no heterogéneo es el DNS.

El DNS permite que cualquier cambio efectuado solamente en un servidor se replique en todos los servidores DNS que la configuración permita, de tal forma que el cambio sólo se efectúa en un servidor, obteniendo así facilidad y simplicidad en el cambio. Por lo tanto, cualquier cambio es dinámico: configuras solamente un servidor y éste se encarga de replicar el cambio.

Por otro lado, que es lo que pasa si un servidor DNS está caído y por lo tanto la conectividad con el mismo no es posible: ¿quedaría todo el sistema inhabilitado? ¿te podrías conectar aún a páginas web? Bien, pues como cada servidor DNS se ocupa de su zona, eso no imposibilita el acceso a otras zonas y por lo tanto a la visibilidad y conectividad de otros dominios que no dependan de ese servidor DNS. Es más, es posible que no solamente exista un servidor DNS configurado para controlar esa zona, y por lo tanto tampoco esa zona estuviese no visible.

Una zona DNS es aquella parte del DNS para la cual se ha delegado la administración, es decir, cuando configuras un dominio en un servidor DNS, éste debe pertenecer a una zona. Así, en los archivos de configuración de zona se indicará qué IP va con el servicio web www, el servicio de correo mail, etc. Los tipos de zonas posibles son dos:

1. **Zona de Búsqueda Directa:** las resoluciones de esta zona devuelven la dirección IP correspondiente al recurso solicitado. Realiza las resoluciones que esperan como respuesta la dirección IP de un determinado recurso.
2. **Zona de Búsqueda Inversa:** las resoluciones de esta zona buscan un nombre de equipo en función de su dirección IP; una búsqueda inversa tiene forma de pregunta, del estilo "¿Cuál es el nombre DNS del equipo que utiliza la dirección IP 192.168.200.100?".

Los servidores DNS no solamente sirven para la resolución de nombres en Internet, también se pueden utilizar en redes locales. Así, las entradas existentes en los DNS de la red local podrían ser visibles en Internet, o no, solamente sirviendo resolución a los equipos de la red local. De esta forma, cuando un usuario de la red local intenta acceder a un recurso local, podrá utilizar **nombres** en lugar de direcciones IP. Si el usuario desea acceder fuera de la red local a algún recurso en Internet, el DNS local nunca podrá llevar a cabo dicha resolución y se la traslada al siguiente servidor DNS (que sí estará en Internet) en su jerarquía de servidores DNS, hasta que la petición sea satisfecha.

Por ejemplo, con un servidor DNS en nuestra red local, que resuelve la IP `192.168.200.100` a `cliente.local` y viceversa, puedes ejecutar el comando `ping` indistintamente contra dicha IP o contra el nombre del equipo en el dominio:

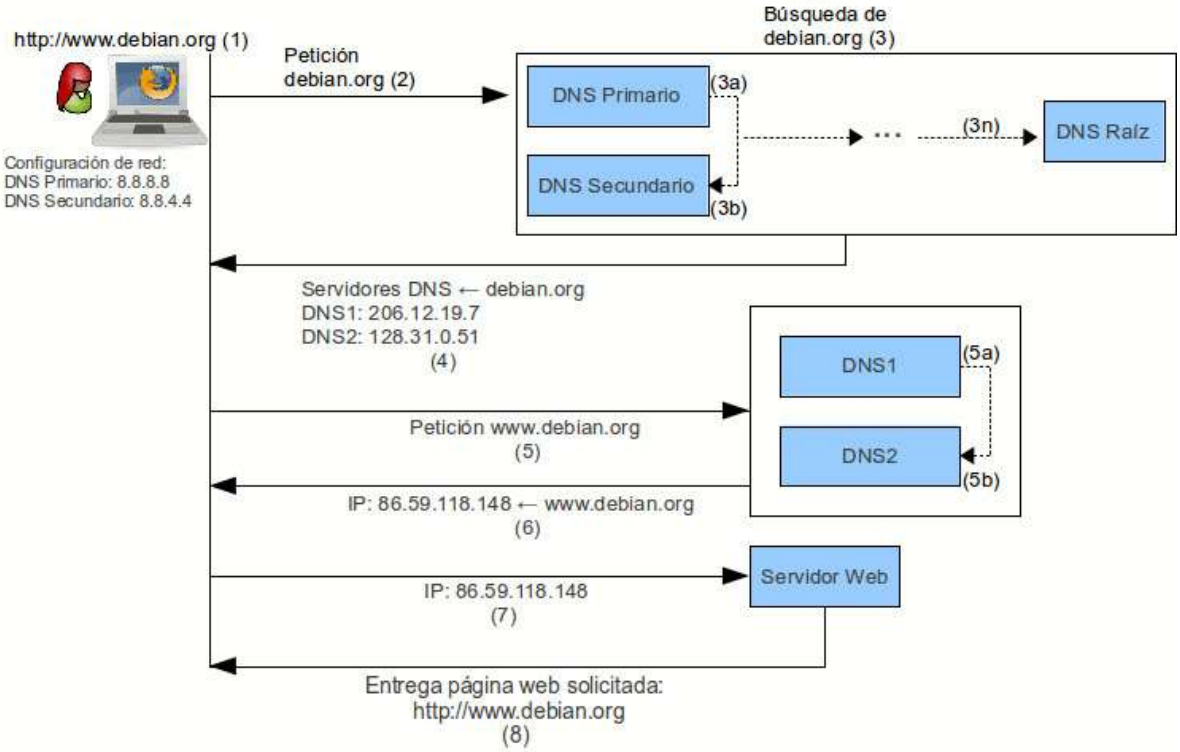
```
ping 192.168.200.100
ping cliente.local
```

En ambos casos, deberías obtener la misma respuesta. Esto suele ser muy útil cuando los hosts reciben su IP por DHCP (*Protocolo de red que permite a los clientes de una red IP obtener sus parámetros de configuración de red automáticamente*) ya que puede ocurrir que desconozcamos la IP que tiene cierto equipo pero sí conocer su nombre en el dominio, que será invariable.

Podemos resumir entonces las ventajas de la configuración y empleo de un servidor DNS en las siguientes:

1. Desaparece la carga excesiva en la red y en los hosts: ahora la información esta distribuida por toda la red, al tratarse de una base de datos distribuida.
2. No hay duplicidad de nombres: el problema se elimina debido a la existencia de dominios controlados por un único administrador. Puede haber nombres iguales pero en dominios diferentes.
3. Consistencia de la información: ahora la información que está distribuida es actualizada automáticamente sin intervención de ningún administrador.

1.3.- Funcionamiento del DNS.



La anterior imagen presenta gráficamente el funcionamiento del DNS, tomando como ejemplo la página web www.debian.org y considerando que la información de la petición del dominio a buscar no se encuentra en tu ordenador o en un servidor DNS local existente en tu red o en tu ordenador.

1. A través de tu navegador quieres consultar la página web oficial de Debian escribiendo en la barra de direcciones laURL http://www.debian.org.
2. El navegador busca la información de las DNS del dominio **debian.org**.
3. Internet está ordenada en forma de árbol invertido, si no encuentra la información en tu ordenador, irá a buscarla a los servidores DNS que posees en la configuración de red de tu ordenador, típicamente los proporcionados por tu Proveedor de Servicios a Internet (ISP): DNS Primario (3a) o DNS Secundario (3b). De no estar, seguirá buscándola a niveles superiores y, en último lugar, lo encontrará en el Servidor de Nombres Raíz: DNS Raíz (3n).
4. La información buscada: las IP correspondientes al servidor DNS que gobierna el dominio debian.org, llega a tu ordenador: DNS1→ 206.12.19.7 y DNS2→ 128.31.0.51. Suelen ser dos porque las especificaciones de diseño de DNS recomiendan que, como mínimo, deben existir dos servidores DNS para alojar cada zona, a la que pertenece cada dominio.
   Tu ordenador ahora intentará conectar con el servidor DNS1 (5a) o ante cualquier problema de conexión con éste lo intentará con el servidor DNS2 (5b). Éstos son los servidores de nombres donde se encuentra información acerca de dónde se puede buscar la página web (servidor de la web), una dirección de correo electrónico (servidor de correo), etc.
5. Tu ordenador recibirá la información acerca de la localización de la página web, o sea, la dirección IP del servidor web donde está alojada la página.
6. Tu ordenador se dirigirá luego al servidor web y buscará la página web en él.
7. Por último, el servidor web devuelve la información pedida y tú recibes la página web, visualizándola en el navegador.

Pero, y si vuelves a consultar la página web oficial de Debian escribiendo en la barra de direcciones la URL http://www.debian.org, ¿se repetirá de nuevo todo el proceso? Para contestar esta pregunta hay que establecer dos situaciones:

1. El host desde el que vuelves a realizar la consulta es el mismo: Si no lo es, antes de repetir todo el proceso se intentaría con lo expuesto en el siguiente punto, pero si es el mismo, al haber hecho

la consulta desde este host, la resolución dominio-IP se guarda durante algún tiempo en la memoria caché del mismo, por lo cual no será necesario repetir todo el proceso de nuevo. Si el tiempo en el que la memoria caché guarda la resolución ha expirado se volverá a repetir el proceso de nuevo.

2. Existe un servidor DNS caché en tu red o en tu host: por lo tanto, si un segundo cliente, que tiene configurado este servidor DNS, vuelve a realizar la misma petición, como este servidor tiene la respuesta almacenada en su memoria caché, responderá inmediatamente sin tener que cursar la petición a ningún servidorDNS de Internet. Si el tiempo en el que la memoria caché guarda la resolución ha expirado se volverá a repetir el proceso de nuevo.

1.4.- DNS Dinámico.

¿Es posible si dispones de una conexión a Internet con IP dinámica ofrecer servicios en Internet?

Parece claro que si dispones de una IP estática de conexión a Internet, previo pago de un plus por disponer siempre de una misma IP para tu conexión a Internet, simplemente deberías enrutar las peticiones de los servicios que ofreces a los hosts que esperan la conexión a esos servicios. Si además, posees nombres de dominios puedes redireccionar esos nombres a las IP de tus hosts a través del servidor DNS.

Pero, volviendo a la pregunta, qué es lo que pasa si quieres hacer lo mismo y no dispones de IP estática, esto es, cada vez que te conectas a Internet tu IP, aunque a veces sea la misma, no siempre es la misma. Pues, sí, sí es posible, ¿cómo?. A priori, si lo piensas un poco, lo único que necesitarías sería:

1. Recoger la IP de tu conexión cada vez que te conectas en Internet.
2. Una vez recogida tu IP difundirla en Internet. Para difundirla, o bien lo haces de forma estática, y cada vez que la recoges te preocupas de hacer los cambios necesarios para difundirla, o bien de forma dinámica configuras un programa para que automáticamente recoja la IP y la difunda.

Está claro, que la mejor opción es difundirla de forma dinámica, para ello puedes aprovecharte de servicios ofrecidos, incluso de forma gratuita, por **https://www.dyndns.com/account/** , **http://www.no-ip.com/services/managed_dns/free_dynamic_dns.html** y **http://freedns.afraid.org/** . De hecho, hoy en día, los routers que los ISP suelen montar ya poseen la opción de configuración por DNS dinámica.

Entonces, el **DNS dinámico** es un sistema que permite la actualización en tiempo real de la información sobre nombres de dominio situados en un servidor de nombres, siendo usado, mayoritariamente, para asignar un nombre de dominio de Internet a un ordenador con dirección IP variable (dinámica).

El DNS dinámico, así, puede ofrecer servicios en Internet en hosts que posean conexión con dirección IP dinámica, la típica configuración que los ISP ofrecen para conectarse a Internet.

De todos modos, aunque existe la posibilidad de ofrecer servicios en Internet desde tu propia casa, debes tener en cuenta que, usualmente, la infraestructura técnica y la electrónica de red que poseas no se pueda comparar con los servidores ofrecidos por empresas de Hosting, así: ¿posees balanceadores de carga? ¿redundancia en caso de fallos? ¿generadores eléctricos que garanticen conexión eléctrica permanente a pesar de caída eléctrica?¿Y, sobre todo, dispones del ancho de banda necesario para permitir múltiples conexiones concurrentes sin perjudicar el servicio ofrecido?

Si no tienes configurado un servidor DNS con las entradas de dominio necesarias, puedes generar estas entradas modificando el archivo **/etc/hosts**, añadiéndolas al final del mismo:

```
#IP nombre-dominio
192.168.200.250 empresa1.com www.empresa1.com
```

```
192.168.200.250 empresa2.com www.empresa2.com
```

Cada campo, de cada entrada, puede ir separado por espacios o por tabulados.
Estas entradas solamente serán efectivas en el equipo en el que se modifique el archivo `/etc/hosts`. Así, debes modificar el archivo `/etc/hosts` en cada equipo que quieres que se resuelvan esas entradas.

1.5.- Tipos de servidores DNS.

Como puedes comprobar en la siguiente imagen, existen varios tipos de servidores DNS que describiremos a continuación.

Dependiendo de la configuración y funcionamiento de los servidores, éstos pueden desempeñar distintos papeles:

- ✓ **Servidores primarios** (*primary name servers*). Estos servidores almacenan la información de su zona en una base de datos local. Son los responsables de mantener la información actualizada y cualquier cambio debe ser notificado a este servidor.
- ✓ **Servidores secundarios** (*secundary name servers*). También denominados *esclavos*, aunque a su vez pueden ser *maestros* de otros servidores secundarios. Son aquellos que obtienen los datos de su zona desde otro servidor que tenga autoridad para esa zona. El proceso de copia de la información se denomina *transferencia de zona*.
- ✓ **Servidores maestros** (*master name servers*). Los servidores maestros son los que transfieren las zonas a los servidores secundarios. Cuando un servidor secundario arranca busca un servidor maestro y realiza la transferencia de zona. Un servidor maestro para una zona puede ser a la vez un servidor primario o secundario de esa zona. Así, se evita que los servidores secundarios sobrecarguen al servidor primario con transferencias de zonas. Por ejemplo, en la imagen el servidor DNS3 pide la zona al servidor DNS2 y no al servidor DNS1, con lo cual se evita la sobrecarga del servidor DNS1. Los servidores maestros extraen la información desde el servidor primario de la zona.
- ✓ **Servidores sólo caché** (*caching-only servers*). Los servidores sólo caché no tienen autoridad sobre ningún dominio: se limitan a contactar con otros servidores para resolver las peticiones de los clientes DNS. Estos servidores mantienen una memoria caché con las últimas preguntas contestadas. Cada vez que un cliente DNS le formula una pregunta, primero consulta en su memoria caché. Si encuentra la dirección IP solicitada, se la devuelve al cliente; si no, consulta a otros servidores, apunta la respuesta en su memoria caché y le comunica la respuesta al cliente. Disponer de un servidor caché DNS en nuestra red local aumenta la velocidad de la conexión a Internet pues cuando navegamos por diferentes lugares, continuamente se están realizando peticiones DNS. Si nuestro caché DNS almacena la gran mayoría de peticiones que se realizan desde la red local, las respuestas de los clientes se satisfarán prácticamente de forma instantánea proporcionando al usuario una sensación de velocidad en la conexión. Muchos routers ADSL ofrecen ya este servicio de caché, tan solo hay que activarlo y configurar una o dos IPs de servidores DNS en Internet. En los equipos de nuestra red local podríamos poner como DNS primario la IP de nuestro router y como DNS secundario una IP de un DNS de Internet.

Los servidores secundarios son importantes por varios motivos. En primer lugar, por seguridad: debido a que la información se mantiene de forma redundante en varios servidores a la vez. Si un servidor tiene problemas, la información se podrá recuperar desde otro. Y en segundo lugar, por velocidad: porque evita la sobrecarga del servidor principal

distribuyendo el trabajo entre distintos servidores situados estratégicamente (por zonas geográficas, por ejemplo).
Todos los servidores DNS guardan en la caché las consultas que resolvieron.
Una transferencia de zona puede darse en cualquiera de los casos siguientes:

- ✓ Cuando vence el intervalo de actualización de una zona.
- ✓ Cuando un servidor maestro notifica los cambios de la zona a un servidor secundario.
- ✓ Cuando se inicia el servicio Servidor DNS en un servidor secundario de la zona.
- ✓ Cuando se utiliza el comando `rndc` en un servidor secundario de la zona para iniciar manualmente una transferencia desde su servidor maestro, por ejemplo:

```
rndc retransfer proyecto-empresa.local
```

donde:

`retransfer` → indica que la acción a realizar es una transferencia.
`proyecto-empresa.local` → es el nombre de la zona que quieres transferir.

1.6.- Servidores raíz.

La organización que gestiona globalmente los servidores raíz por concesión del gobierno estadounidense es la ICANN, la cual es una organización sin fines de lucro que opera a nivel internacional, responsable de asignar espacio de direcciones numéricas de protocolo de Internet (IP), identificadores de protocolo y de las funciones de gestión [o administración] del sistema de nombres de dominio de primer nivel genéricos (gTLD) y de códigos de países (ccTLD), así como de la administración del sistema de servidores raíz. Aunque en un principio estos servicios los desempeñaba IANA y otras entidades bajo contrato con el gobierno de EE.UU., actualmente son responsabilidad de ICANN.

ICANN es responsable de la coordinación de la administración de los elementos técnicos del DNS para garantizar una resolución unívoca de los nombres, de manera que los usuarios de Internet puedan encontrar todas las direcciones válidas. Para ello, se encarga de supervisar la distribución de los identificadores técnicos únicos usados en las operaciones de Internet, y delegar los nombres de dominios de primer nivel, como: **com**, **info**, etc.

Otros asuntos que preocupan a los usuarios de Internet, como reglamentación para transacciones financieras, control del contenido de Internet, correo electrónico de publicidad no solicitada (SPAM) y protección de datos, están fuera del alcance de la misión de coordinación técnica de ICANN.

En el siguiente enlace puedes encontrar más información sobre ICANN.
http://www.icann.org/

Las empresas, ciudades u organizaciones podrán registrar sus propios dominios genéricos, tras la decisión adoptada el 20 de Junio de 2011 por la ICANN en Singapur. Esta iniciativa permitirá que las direcciones de los dominios puedan terminar con el nombre de compañía, ciudad, etc., en vez de .com, .net o.org.

"ICANN ha abierto el sistema de direcciones de Internet a las ilimitadas posibilidades de la imaginación humana. Nadie puede saber dónde nos llevará esta histórica decisión", dijo Rod Beckstrom, presidente y jefe ejecutivo de la organización.

Los servidores raíz son entidades distintas. Hay 13 servidores raíz o, más precisamente, 13 direcciones IP en Internet en las que pueden encontrarse a los servidores raíz (los servidores que

tienen una de las 13 direcciones IP pueden encontrarse en docenas de ubicaciones físicas distintas). Todos estos servidores almacenan una copia del mismo archivo que actúa como índice principal de las agendas de direcciones de Internet. Enumeran una dirección para cada dominio de nivel principal (**.com**, **.es**, etc.) en la que puede encontrase la propia agenda de direcciones de ese registro.

En realidad, los servidores raíz no se consultan con mucha frecuencia (considerando el tamaño de Internet) porque una vez que los ordenadores de la red conocen la dirección de un dominio de nivel principal concreto pueden conservarla, y sólo comprueban de forma ocasional que esa dirección no haya cambiado. Sin embargo, los servidores raíz siguen siendo una parte vital para el buen funcionamiento de Internet.

Las entidades encargadas de operar los servidores raíz son bastante autónomas pero, al mismo tiempo, colaboran entre sí y con ICANN para asegurar que el sistema permanece actualizado con los avances y cambios de Internet.

Los trece servidores raíz DNS se denominan por las primeras trece letras del alfabeto latino, de la A hasta la M (**A.ROOT-SERVERS.NET.**, **B.ROOT-SERVERS.NET.**, …,**M.ROOT-SERVERS.NET.**), y están en manos de 9 organismos y corporaciones diferentes e independientes, principalmente universidades, empresas privadas y organismos relacionados con el ejercito de EE.UU. Aproximadamente la mitad depende de organizaciones públicas estadounidenses.

En el siguiente enlace encontrarás la lista actualizada de los servidores raíz DNS.

Anexo I - Los servidores raíz DNS.

En el siguiente enlace accederás al contenido de la zona de los servidores raíz DNS. Esta información es publicada por los servidores raíz DNS.

Información publicada por los servidores raíz DNS.

1.7.- Tipos de registros DNS.

Una base de datos DNS se compone de uno o varios archivos de zonas utilizados por el servidor DNS. Cada zona mantiene un conjunto de registros de recursos estructurados.

Todos los registros de recursos (RR) tienen un formato definido que utiliza los mismos campos de nivel superior, según se describe en la tabla siguiente:

Formato de los registros de recursos DNS	
Campo	**Descripción**
Propietario	Indica el nombre de dominio DNS que posee un registro de recursos. Este nombre es el mismo que el del nodo del árbol de la consola donde se encuentra un registro de recursos.
Tiempo de vida (TTL)	Para la mayor parte de los registros de recursos, este campo es opcional. Indica el espacio de tiempo utilizado por otros servidores DNS para determinar cuánto tarda la información en caché en caducar un registro y descartarlo. Por ejemplo, la mayor parte de los registros de recursos que crea el servicio del servidor DNS heredan el TTL mínimo (predeterminado) de 1 hora desde el registro de recurso de inicio de autoridad (SOA) que evita que otros servidores DNS almacenen en caché durante demasiado tiempo. En un registro de recursos individual, puede especificar unTTL específico para el registro que suplante el TTL mínimo (predeterminado) heredado del registro de recursos de inicio de autoridad. También se puede utilizar el valor cero (0) para el TTL en los registros de recursos que contengan datos volátiles que no estén en la memoria caché para su uso posterior una vez se complete la consulta DNS en curso.
Clase	Contiene texto nemotécnico estándar que indica la clase del registro de recursos.

	Por ejemplo, el valor "IN" indica que el registro de recursos pertenece a la clase Internet. Este campo es *obligatorio*.
Tipo	Contiene texto nemotécnico estándar que indica el tipo de registro de recursos. Por ejemplo, el texto nemotécnico "A" indica que el registro de recursos almacena información de direcciones de host. Este campo es *obligatorio*.
Datos específicos del registro	Un campo de longitud variable y *obligatorio* con información que describe el recurso. El formato de esta información varía según el tipo y clase del registro de recursos.

En la siguiente tabla se muestran los registros DNS más utilizados:

***Nota**: en los siguientes ejemplos de registros de recurso, el campo TTL se omite en caso de ser opcional. El campo TTL se ha incluido en la sintaxis de cada registro para indicar dónde puede agregarse.*

Tipos de registros DNS	
Registro	**Descripción, sintaxis y ejemplo**
A	**Descripción**: Address (*Dirección*). Este registro se usa para traducir nombres de hosts a direcciones IP versión **4**. **Sintaxis**: ***propietario clase ttl* A *IP_version4***. **Ejemplo**: `host1.ejemplo.com IN A 127.0.0.1.`
AAAA	**Descripción**: Address (*Dirección*). Este registro se usa para traducir nombres de hosts a direcciones IP versión **6**. **Sintaxis**: ***propietario clase ttl* AAAA *IP_version6***. **Ejemplo**: `host1ipv6.ejemplo.com. IN AAAA 1234:0:1:2:3:4:567:89ab.`
CNAME	**Descripción**: **C**anonical **Name** (*Nombre Canónico*). Se usa para crear nombres de hosts adicionales, o alias. Hay que tener en cuenta que el nombre de host al que el alias referencia debe haber sido definido previamente como registro tipo "A". Comúnmente usado cuando un servidor con una sola dirección IP ejecuta varios servicios, como: ftp, web... y cada servicio tiene su propia entrada DNS. También es utilizado cuando el servidor web aloja distintos dominios en una misma IP (*virtualhosts*). **Sintaxis**: ***propietario ttl clase* CNAME *nombreCanónico***. **Ejemplo**: `nombrealias.ejemplo.com CNAME nombreverdadero.ejemplo.com.` Como se ha comentado anteriormente `nombreverdadero.ejemplo.com` previamente debe estar definido como registro tipo **A**.
NS	**Descripción**: **N**ame **S**erver (*Servidor de Nombres*). Indica qué servidores de nombres tienen total autoridad sobre un dominio concreto. Cada dominio se puede asociar a una cantidad cualquiera de servidores de nombres. **Sintaxis**: ***propietario ttl IN* NS *nombreServidorNombreDominio***. **Ejemplo**: `ejemplo.com. IN NS nombreservidor1.ejemplo.com.`
MX	**Descripción**: **M**ail e**X**change (*Registro de Intercambio de Correo*). Asocia un nombre de dominio a una lista de servidores de intercambio de correo para ese dominio. **Sintaxis**: ***propietario ttl clase* MX *preferencia hostIntercambiadorDeCorreo***. **Ejemplo**: `ejemplo.com. MX 10 servidorcorreo1.ejemplo.com.` El número, en este caso 10, indica la preferencia, y tiene sentido en caso de existir varios servidores de correo. A menor número mayor preferencia.
PTR	**Descripción**: **P**oin**T**e**R** (*Indicador*). Traduce direcciones IP en nombres de dominio. También conocido como 'registro inverso', ya que funciona a la inversa del registro "A". **Sintaxis**: ***propietario ttl clase* PTR *nombreDominioDestino***. **Ejemplo**: `1.0.0.10.in-addr.arpa. PTR host.ejemplo.com.`
SOA	**Descripción**: **S**tart **O**f **A**uthority (*Autoridad de la zona*). Proporciona información sobre el servidor DNS primario de la zona. **Sintaxis**: ***propietario clase* SOA *servidorNombres personaResponsable (numeroSerie intervaloActualización intervaloReintento caducidad tiempoDeVidaMínimo)***. **Ejemplo:**

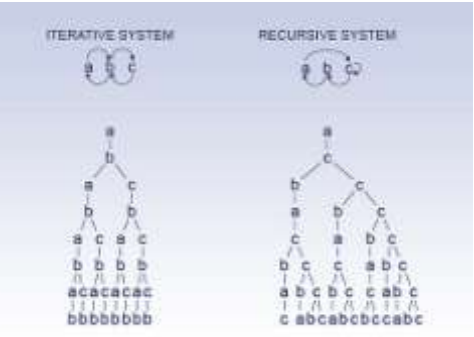
	`@ IN SOA nombreServidor.ejemplo.com. postmaster.ejemplo.com. (`
	`1 ; número de serie`
	`3600 ; actualizar [1h]`
	`600 ; reintentar [10m]`
	`86400 ; caducar [1d]`
	`3600 ) ; TTL mínimo [1h]`
	El propietario (*servidor DNS principal*) se especifica como "**@**" porque el nombre de dominio es el mismo que el origen de todos los datos de la zona (**ejemplo.com**). Se trata de una convención de nomenclatura estándar para registros de recursos y se utiliza más a menudo en los registros SOA. El número de serie es el número de versión de esta base de datos. Debes incrementar este número cada vez que modificas la base de datos.

TXT	**Descripción**: **TeXT** (*Información textual*). Permite a los dominios identificarse de modos arbitrarios. **Sintaxis**: ***propietario ttl clase* TXT *cadenaDeTexto***. **Ejemplo**: `ejemplo.com. TXT "Ejemplo de información de nombre de dominio adicional."`
SPF	**Descripción**: **S**ender **P**olicy **F**ramework. Es un registro de tipo TXT que va creado en una zona directa del DNS, en la cual se pone las informaciones del propio servidor de correo con la sintaxis SPF. Se utiliza para evitar el envío de correos suplantando identidades. Por lo tanto, ayuda a combatir el SPAM, ya que, en este registro se especifica cual o cuales hosts están autorizados a enviar correo desde el dominio dado. El servidor que recibe, consulta el S para comparar la IP desde la cual le llega, con los datos de este registro. **Sintaxis**: ***propietario ttl clase* IN SPF *cadenaDeTexto***. **Ejemplo**: `ejemplo.com IN SPF "v=spf1 a:mail.ejemplo.com -all"`.

En el siguiente enlace encontrarás más información sobre el registro SPF.
http://www.openspf.org/

1.8.- Funcionamiento del cliente DNS.

"Los sabios son los que buscan la sabiduría; los necios piensan ya haberla encontrado."
Napoleón Bonaparte



Cuando utilizas en un programa un nombre DNS, éste debe ser resuelto a una IP. Entonces, un cliente DNS busca el nombre que se utiliza en el programa, consultando los servidores DNS para resolver el nombre. Cada mensaje de consulta que envía el cliente contiene tres grupos de información, que especifican una pregunta que tiene que responder el servidor:

- ✓ Un nombre de dominio DNS especificado, indicado como un nombre de dominio completo (FQDN).
- ✓ Un tipo de consulta especificado, que puede establecer un registro de recursos por tipo o un tipo especializado de operación de consulta.
- ✓ Una clase especificada para el nombre de dominio DNS.

Por ejemplo, el nombre especificado puede ser el nombre completo de un equipo, como `rrhh.departamento.empresa.org.`, y el tipo de consulta especificado para buscar un registro de recursos de dirección (A) por ese nombre. Considere una consulta DNS como una pregunta de un cliente a un servidor en dos partes, como: "¿Tiene algún registro de recursos de dirección (A) de un equipo llamado '`rrhh.departamento.empresa.org.`'?". Cuando el cliente recibe una respuesta del servidor, lee e interpreta el registro de recursos "A" respondido, y aprende la dirección IP del equipo al que preguntó por el nombre.

Las consultas DNS se resuelven de diferentes formas:

- ✓ A veces, un cliente responde a una consulta localmente mediante la información almacenada en la caché obtenida de una consulta anterior.
- ✓ El servidor DNS puede utilizar su propia caché de información de registros de recursos para responder a una consulta.
- ✓ Un servidor DNS también puede consultar, o ponerse en contacto con otros servidores DNS, en nombre del cliente solicitante para resolver el nombre por completo y, a continuación, enviar una respuesta al cliente. Este proceso se llama **recursividad**.
- ✓ Además, el mismo cliente puede intentar ponerse en contacto con servidores DNS adicionales para resolver un nombre. Cuando un cliente lo hace, utiliza consultas adicionales e independientes en función de respuestas de referencia de los servidores. Este proceso se llama **iteración**.

En general, el proceso de consulta DNS se realiza en dos partes:

- ✓ La consulta de un nombre comienza en un equipo cliente y se pasa al solucionador (resolver), el servicio Cliente DNS, para proceder a su resolución.
- ✓ Cuando la consulta no se puede resolver localmente, se puede consultar a los servidores DNS según sea necesario para resolver el nombre.

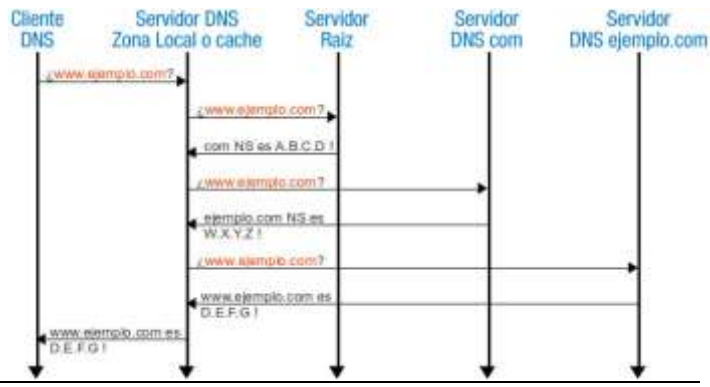
1.8.1.- Consultas recursivas.

"Daría todo lo que sé por la mitad de lo que ignoro.

"René Descartes

Tu ordenador (cliente DNS) formula una **consulta** a tu servidor DNS preferido (el que tienes configurado como primero en tu configuración de red, generalmente el proveedor de Internet). Cuando el servidor DNS recibe una consulta, primero comprueba si puede responder la consulta en las zonas configuradas localmente en el servidor, esto es, en las zonas que posee autoridad. Así, pueden ocurrir dos situaciones:

1. Si el nombre consultado existe, esto es, coincide con un registro de recursos correspondiente en la información de zona local, el servidor responde con autoridad y usa esta información para resolver el nombre consultado.
2. Si el nombre consultado no existe, esto es, no existe ninguna información de zona para el nombre consultado, a continuación el servidor comprueba si puede resolver el nombre mediante la información almacenada en la caché local de consultas anteriores. De nuevo, se dan dos situaciones:
   a. Si el servidor preferido puede responder al cliente solicitante con una respuesta coincidente de su caché, finaliza la consulta y responde con esta información.
   b. Si el servidor preferido no puede responder al cliente solicitante con una respuesta coincidente de su caché, el proceso de consulta puede continuar y se usa la recursividad para resolver completamente el nombre. Esto implica la asistencia de otros servidores DNS para ayudar a resolver el nombre. De forma predeterminada, el servicio cliente DNS solicita al servidor que utilice un proceso de recursividad para resolver completamente los nombres en nombre del cliente antes de devolver una respuesta. En la mayor parte de los casos, el servidor DNS se configura, de forma predeterminada, para admitir el proceso de recursividad como se muestra en el gráfico siguiente.



Para que el servidor DNS realice la recursividad correctamente, primero necesita información de contacto útil acerca de los otros servidores DNS del espacio de nombres de dominio DNS. Esta información se proporciona en forma de *sugerencias de raíz*, una lista de los registros de recursos preliminares que puede utilizar el servicio DNS para localizar otros servidores DNS que tienen autoridad para la raíz del árbol del espacio de nombres de dominio DNS. Los servidores raíz tienen autoridad para el dominio raíz y los dominios de nivel superior en el árbol del espacio de nombres de dominio DNS.

Un servidor DNS puede completar el uso de la recursividad utilizando las sugerencias de raíz para encontrar los servidores raíz. En teoría, este proceso permite a un servidor DNS localizar los servidores que tienen autoridad para cualquier otro nombre de dominio DNS que se utiliza en cualquier nivel del árbol del espacio de nombres.

Por ejemplo, piense en la posibilidad de usar el proceso de recursividad para localizar el nombre "`www.ejemplo.com`" cuando el cliente consulte un único servidor DNS. El proceso ocurre cuando un servidor y un cliente DNS se inician y no tienen información almacenada en la caché local disponible para ayudar a resolver la consulta de un nombre. El servidor supone que el nombre consultado por el cliente es para un nombre de dominio del que el servidor no tiene conocimiento local, según sus zonas configuradas.

Primero, el servidor preferido analiza el nombre completo y determina que necesita la ubicación del servidor con autoridad para el dominio de nivel superior "**com**". A continuación, utiliza una consulta iterativa al servidor DNS "**com**" para obtener una referencia al servidor "**ejemplo.com**". Finalmente, se entra en contacto con el servidor "ejemplo.com". Ya que este servidor contiene el nombre consultado como parte de sus zonas configuradas, responde con autoridad al servidor original que inició la recursividad. Cuando el servidor original recibe la respuesta que indica que se obtuvo una respuesta con autoridad a la consulta solicitada, reenvía esta respuesta al cliente solicitante y se completa el proceso de consulta recursiva.

Aunque el proceso de consulta recursiva puede usar muchos recursos cuando se realiza como se describe anteriormente, tiene algunas ventajas en el rendimiento para el servidor DNS. Por ejemplo, durante el proceso de recursividad, el servidor DNS que realiza la búsqueda recursiva obtiene información acerca del espacio de nombres de dominio DNS. Esta información se almacena en la caché del servidor y se puede utilizar de nuevo para ayudar a acelerar la obtención de respuestas a consultas subsiguientes que la utilizan o concuerdan con ella. Con el tiempo, esta información almacenada en caché puede crecer hasta ocupar una parte significativa de los recursos de memoria del servidor, aunque se limpia siempre que el servicio DNS se activa y desactiva.

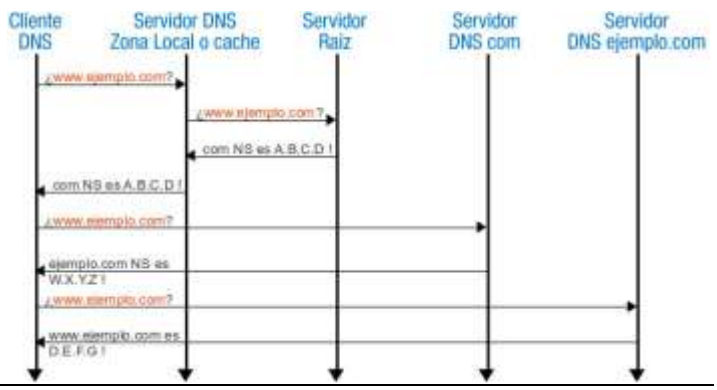
1.8.2.- Consultas iterativas.

"Yo no procuro conocer las preguntas; procuro conocer las respuestas.

"Confucio

La iteración es el tipo de resolución de nombres que se utiliza entre clientes y servidores DNS cuando se dan las condiciones siguientes:

- ✓ El cliente solicita el uso de la recursividad, pero ésta se encuentra deshabilitada en el servidor DNS.



- ✓ El cliente no solicita el uso de la recursividad cuando consulta el servidor DNS.

Una solicitud iterativa de un cliente informa al servidor DNS de que el cliente espera la mejor respuesta que el servidor DNSpueda proporcionar inmediatamente, sin entrar en contacto con otros servidores DNS.

Cuando se utiliza la iteración, un servidor DNS responde al cliente en función de su propio conocimiento específico acerca del espacio de nombres, sin tener en cuenta los datos de los nombres que se están consultando. Por ejemplo, si un servidor DNS de una intranet recibe una consulta de un cliente local para "`www.ejemplo.com`", es posible que devuelva una respuesta de su caché de nombres. Si el nombre consultado no está almacenado actualmente en la caché de nombres del servidor, puede que, para responder, el servidor proporcione una referencia, es decir, una lista de registros de recursos de dirección (**A**) y de servidor de nombres (**NS**) para otros servidores DNS que estén más cerca del nombre consultado por el cliente.

Cuando se proporciona una referencia, el cliente DNS asume la responsabilidad de continuar efectuando consultas iterativas a otros servidores DNS configurados para resolver el nombre. Por ejemplo, en el caso más complicado, el cliente DNS puede expandir su búsqueda a los servidores de dominio raíz en Internet en un esfuerzo por localizar los servidores DNS que tienen autoridad para el dominio "**com**". Una vez en contacto con los servidores raíz de Internet, puede recibir más respuestas iterativas de estos servidores DNS que señalan a los servidores DNS de Internet reales para el dominio "**ejemplo.com**". Cuando se proporcionan registros de estos servidores DNS al cliente, éste puede enviar otra consulta iterativa a los servidores DNS externos del dominio **ejemplo** en Internet, que pueden responder con una respuesta definitiva y con autoridad.

Cuando se utiliza la iteración, un servidor DNS puede ayudar en la resolución de la consulta de un nombre además de devolver su mejor respuesta propia al cliente. En la mayor parte de las consultas iterativas, un cliente utiliza su lista de servidores DNS configurada localmente para entrar en contacto con otros servidores de nombres a través del espacio de nombres DNS si su servidor DNS principal no puede resolver la consulta.

1.8.3.- Consultas inversas.

"El modo de dar una vez en el clavo es dar cien veces en la herradura.
"Miguel de Unamuno

En la mayoría de la consultas DNS los clientes normalmente realizan una búsqueda directa. Este tipo de consulta espera recibir una dirección IP como respuesta a la consulta. Pero, DNS también proporciona un proceso de búsqueda inversa, es decir, buscar un nombre de host a través de una dirección IP. Así, una búsqueda inversa busca la respuesta a una pregunta tipo como la siguiente: ¿Cuál es el nombre DNS del host que utiliza la dirección IP 192.168.200.100?.

DNS no se diseñó originalmente para aceptar este tipo de consulta. Un problema de compatibilidad con el proceso de consulta inversa es la diferencia en la forma en que el espacio de nombres DNS organiza e indexa los nombres, y cómo se asignan las direcciones IP. Si el único método para responder a la pregunta anterior fuera buscar en todos los dominios del espacio de nombres DNS, una consulta inversa llevaría demasiado tiempo y requeriría un procesamiento demasiado largo como para ser útil.

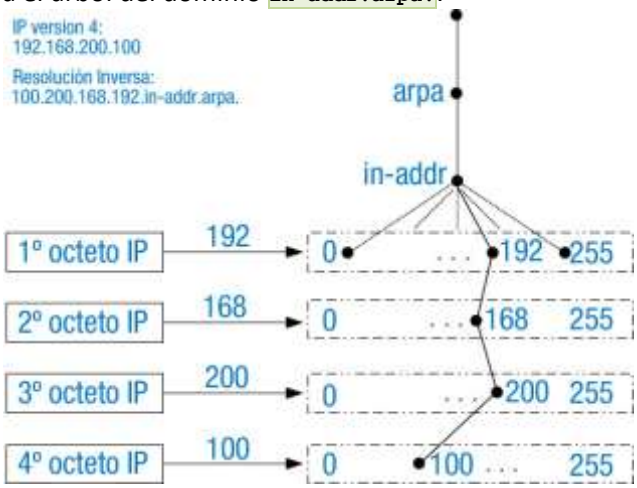
Entonces, para resolver este problema, en el estándar DNS se definió y se reservó un dominio especial para las IP versión 4, el dominio `in-addr.arpa`, en el espacio de nombres DNS de Internet con el fin de proporcionar una forma práctica y confiable para realizar las consultas inversas. Al crear el

espacio de nombres inverso, los subdominios del dominio `in-addr.arpa` se crean con el orden inverso de los números en la notación decimal con puntos de las direcciones IP. Por ejemplo, para la IP192.168.200.100 su resolución inversa sería:

```
100.200.168.192.in-addr.arpa.
```

Este orden inverso de los dominios para el valor de cada octeto es necesario porque, a diferencia de los nombres DNS, cuando se leen las direcciones IP de izquierda a derecha se interpretan al contrario. Cuando se lee una dirección IP de izquierda a derecha, se ve desde su información más general (una dirección IP de red) en la primera parte de la dirección a la información más específica (una dirección IP de host) que contienen los últimos octetos. Por esta razón, se debe invertir el orden de los octetos de las direcciones IP cuando se crea el árbol del dominio `in-addr.arpa.`.

Finalmente, el árbol del dominio `in-addr.arpa`, tal como se crea en DNS, requiere que se defina un tipo de registro de recursos adicional: el registro de recursos de puntero (**PTR**). Este registro de recursos se utiliza para crear una asignación en la zona de búsqueda inversa que, normalmente, corresponde a un registro de recurso de dirección (**A**) de host con nombre para el nombre del equipo DNS de un host en su zona de búsqueda directa.



El dominio `in-addr.arpa` se usa en todas las redes TCP/IP que se basan en el direccionamiento del Protocolo de Internet versión 4 (IPv4). Para el Protocolo de Internet versión 6 (IPv6) se usa un nombre de dominio especial diferente, el dominio `ip6.arpa`.

En el siguiente enlace puedes encontrar más información disponible acerca de IPv6 y DNS, con ejemplos acerca de cómo crear y usar nombres de dominio ip6.arpa, en el documento RFC 3596 Extensiones DNS compatibles con IP versión 6.

http://www.normes-internet.com/normes.php?rfc=rfc3596&lang=es

Ten en cuenta que, si el servidor DNS no puede responder el nombre de la consulta inversa, se puede utilizar la resolución DNS normal (ya sea la recursividad o la iteración) para localizar un servidor DNS con autoridad para la zona de búsqueda inversa y que contenga el nombre consultado. En este sentido, el proceso de resolución de nombres utilizado en una búsqueda inversa es idéntico al de una búsqueda directa.

1.9.- Cómo funcionan los DNS preferidos y alternativos.

El servidor DNS preferido es aquel con el que el cliente prueba en primer lugar. También es el servidor en el que el cliente DNSactualiza sus registros de recursos. Si el servidor DNS preferido falla, el cliente prueba con el servidor DNS alternativo.

Opcionalmente, puedes especificar una lista completa de servidores DNS alternativos. Los servidores DNS preferidos y alternativos especificados se consultan en el orden que aparezcan en la lista.

Sin un servidor DNS preferido, el cliente DNS no puede consultar un servidor DNS. Sin un DNS alternativo, las consultas no se resolverán si el servidor DNS preferido falla.

Los pasos siguientes indican el proceso para entrar en contacto con servidores DNS preferidos y alternativos:

1. El servidor DNS preferido responde primero a una consulta DNS o a una actualización DNS.
2. Si el servidor DNS preferido no responde a una consulta DNS o a una actualización DNS, la consulta o actualización se redirige al servidor DNS alternativo.
3. Si el servidor DNS alternativo no responde y el cliente DNS está configurado con las direcciones IP adicionales de servidores DNS, el cliente DNS envía la consulta o actualización al siguiente servidor DNS de la lista.
4. Si alguno de los servidores DNS (un servidor preferido, un servidor alternativo o cualquier otro de la lista) no responde, dicho servidor se quita temporalmente de la lista.
5. Si ninguno de los servidores DNS responden, la consulta o actualización del cliente DNS no se realiza.

En los equipos tipo GNU/Linux puedes configurar estos servidores en el archivo `/etc/resolv.conf` e incluso puedes realizar balanceo de carga entre ellos, así como la modificación del tiempo de espera efectuado desde que un servidor falla hasta que se prueba con otro.

La configuración sería algo así:

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

En este caso si `8.8.8.8` falla la resolución se realizará a través de `8.8.4.4`. El problema es que por defecto el valor de tiempo de espera (`timeout`) asignado es 5 segundos, por lo que tardará un tiempo en detectar que tiene que utilizar el segundo DNS y todo irá muy lento. Para solucionarlo, tienes que usar la directiva "`options`" y modificar el `timeout`. Así, puedes poner 1 segundo como se demuestra en el siguiente ejemplo:

```
nameserver 8.8.8.8
nameserver 8.8.4.4
options timeout:1
```

Otra opción también interesante es "`rotate`", que permite distribuir la carga entre todos los servidores listados y evitar que todas las peticiones vayan siempre al primero:

```
nameserver 8.8.8.8
nameserver 8.8.4.4
options timeout:1 rotate attempts:1
```

Configurando estas opciones aseguramos que en caso del fallo del servidor DNS preferido el rendimiento de la máquina no se degrade.

Ten en cuenta que es posible y más que probable que el fichero `/etc/resolv.conf` sea modificado cuando configuras la red mediante un gestor de conexión de redes, como: NetworkManager, wicd … Por lo tanto, revisa este fichero.
Es conveniente que le des una visita al manual de resolv.conf: `man resolv.conf`.

Es recomendable que visites el siguiente enlace sobre los servidores DNS públicos de Google: 8.8.8.8 y 8.8.4.4.
http://code.google.com/intl/es/speed/public-dns/index.html

1.10.- Comandos (I).

A la hora de saber si tienes conectividad con alguna máquina en Internet, o en red local, se suele utilizar el comando `ping`, el cual

```
alumno@servidor-fp:~> nslookup ftp.rediris.es
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
ftp.rediris.es  canonical name = zeppo.rediris.es.
Name:   zeppo.rediris.es
Address: 130.206.1.5

alumno@servidor-fp:~> host ftp.rediris.es
ftp.rediris.es is an alias for zeppo.rediris.es.
zeppo.rediris.es has address 130.206.1.5
alumno@servidor-fp:~>
alumno@servidor-fp:~> dig ftp.rediris.es

; <<>> DiG 9.7.3-P3 <<>> ftp.rediris.es
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 486
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;ftp.rediris.es.                        IN      A

;; ANSWER SECTION:
ftp.rediris.es.         4088    IN      CNAME   zeppo.rediris.es.
zeppo.rediris.es.       5364    IN      A       130.206.1.5

;; Query time: 82 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Mon Oct  3 03:51:59 2011
;; MSG SIZE  rcvd: 68

alumno@servidor-fp:~>
```

indica, según su respuesta, si posees conectividad con la máquina en cuestión. El comando **ping** lo puedes utilizar para consultar direcciones IP o nombres de dominios.

Por lo tanto, el comando **ping** debe ser capaz de consultar información sobre el sistema de nombres de dominio; es un resolutor, un programa cliente capaz de consultar información sobre el sistema de nombres de dominio. Normalmente, un resolutor trabaja discretamente en segundo plano y los usuarios no conocen su presencia, es decir, que toda consulta de un cliente DNS a su servidor suele realizarla el programa que invocamos (**ping**, **ftp**, **telnet**, **mail**, **navegador web**, etc.). Por ejemplo, si solicitas una conexión ftp a **ftp.rediris.es**, la aplicación ftp que emplees llama a un programa resolutor local que busca la dirección IP de ese ordenador **130.206.1.5** sin que tengas conciencia de ello, esto es, para ti el proceso es transparente. Además de este trabajo en segundo plano, el usuario puede conectarse directamente al programa resolutor enviando consultas y resolviendo respuestas. Comandos resolutores típicos en sistemas operativos GNU/Linux son: **nslookup**, **host** y **dig**.

El comando **nslookup**, en algunas distribuciones GNU/Linux ya no está soportado pues está obsoleto(*deprecated*). Por lo tanto, hoy en día, se suelen utilizar el comando **host** para consulta de direcciones IP y el comando **dig** para consulta de servidores DNS activos. ¿Cómo funcionan todos estos comandos? Veamos:

Ejemplos de resolución directa: Resolución de nombre a IP.

1. **Comando nslookup:**
   Para consultar la dirección IP del ordenador **ftp.rediris.es**, basta con ejecutar:

```
nslookup ftp.rediris.es
alumno@servidor-fp:~$ nslookup ftp.rediris.es
Server:          8.8.8.8
Address:     8.8.8.8#53

Non-authoritative answer:
ftp.rediris.es     canonical name = zeppo.rediris.es.
Name:     zeppo.rediris.es
Address: 130.206.1.5
```

   Donde puedes ver que **ftp.rediris.es** es un alias (**CNAME**) de **zeppo.rediris.es** cuya dirección IP es **130.206.1.5**
2. **Comando host:**
   Para consultar la dirección IP del ordenador **ftp.rediris.es**, basta con ejecutar:

```
host ftp.rediris.es
alumno@servidor-ftp:~$ host ftp.rediris.es
ftp.rediris.es is an alias for zeppo.rediris.es.
zeppo.rediris.es has address 130.206.1.5
```

   Donde puedes ver que ftp.rediris.es es un alias (**CNAME**) de **zeppo.rediris.es** cuya dirección IP es **130.206.1.5**
3. **Comando dig:**
   Para consultar la dirección IP del ordenador **ftp.rediris.es**, basta con ejecutar:

```
dig ftp.rediris.es
alumno@servidor-ftp:~$ dig ftp.rediris.es

; <<>> DiG 9.7.3 <<>> ftp.rediris.es
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31214
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;ftp.rediris.es.                IN      A

;; ANSWER SECTION:
ftp.rediris.es.          7200     IN     CNAME     zeppo.rediris.es.
zeppo.rediris.es.     5195     IN     A     130.206.1.5

;; Query time: 76 msec
```

```
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Jul 29 11:13:44 2011
;; MSG SIZE rcvd: 68
```

Donde puedes ver que **ftp.rediris.es** es un alias (**CNAME**) de **zeppo.rediris.es** cuya dirección IP es **130.206.1.5**

```
alumno@servidor-fp:~> nslookup 130.206.1.5
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
5.1.206.130.in-addr.arpa        name = zeppo.rediris.es.

Authoritative answers can be found from:

alumno@servidor-fp:~> host 130.206.1.5
5.1.206.130.in-addr.arpa domain name pointer zeppo.rediris.es.
alumno@servidor-fp:~>
alumno@servidor-fp:~>  dig -x 130.206.1.5

; <<>> DiG 9.7.3-P3 <<>> -x 130.206.1.5
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25712
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;5.1.206.130.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
5.1.206.130.in-addr.arpa. 6327  IN      PTR     zeppo.rediris.es.

;; Query time: 85 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Mon Oct  3 04:03:38 2011
;; MSG SIZE  rcvd: 72

alumno@servidor-fp:~>
```

1.10.1.- Comandos (II).

Ejemplos de resolución inversa: Resolución de IP a nombre.

1. **Comando `nslookup`:**

   Para consultar el nombre de la IP **130.206.1.5**, basta con ejecutar:

```
nslookup 130.206.1.5
alumno@servidor-fp:~$ nslookup 130.206.1.5
Server:          80.58.61.254
Address:     80.58.61.254#53

Non-authoritative answer:
5.1.206.130.in-addr.arpa     name = zeppo.rediris.es.

Authoritative answers can be found from:
```

   Donde puedes ver que la IP **130.206.1.5** corresponde con el nombre de dominio **zeppo.rediris.es**

2. **Comando `host`:**

   Para consultar el nombre de la IP **130.206.1.5**, basta con ejecutar:

```
host 130.206.1.5
     alumno@servidor-ftp:~$ host 130.206.1.5
5.1.206.130.in-addr.arpa domain name pointer zeppo.rediris.es.
```

   Donde puedes ver que la IP **130.206.1.5** corresponde con el nombre de dominio **zeppo.rediris.es**

3. **Comando `dig`:**

   Para consultar el nombre de la IP **130.206.1.5**, basta con ejecutar:

```
dig -x 130.206.1.5
alumno@servidor-fp:~$ dig -x 130.206.1.5

; <<>> DiG 9.7.3 <<>> -x 130.206.1.5
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 38384
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;5.1.206.130.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
5.1.206.130.in-addr.arpa. 7200      IN      PTR     zeppo.rediris.es.

;; Query time: 73 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Jul 29 12:03:30 2011
;; MSG SIZE rcvd: 72
```

   Donde puedes ver que la IP **130.206.1.5** corresponde con el nombre de dominio **zeppo.rediris.es**, e incluso el registro de recursos empleado: **PTR**.

Es conveniente que le des una visita al manual de los comandos `nslookup`, `host` y `dig`.

DNSstuff (http://www.dnsstuff.com/): en su página web ofrece herramientas DNS, herramientas de red, herramientas de correo electrónico, información de DNS y recopilación de información IP que te pueden ser muy útil para gestionar, monitorizar y analizar tu servidor DNS.

1.11.- Instalación del servidor DNS BIND.

Para una instalación del servidor DNS BIND en Debian 6 (Squeeze) realiza el siguiente procedimiento como usuario `root`, teniendo en cuenta que el servidor está identificado como sigue:

- ✓ Hostname: `debian-servidor-fp`.
- ✓ IP: `192.168.200.250`.

1. Actualiza los repositorios del sistema operativo.
```
root@debian-servidor-fp:~# apt-get update
```
***NOTA**: es necesario para el buen funcionamiento del comando que tengas configurado correctamente la conexión a Internet.*
2. Actualiza el sistema operativo.
```
root@debian-servidor-fp:~# apt-get upgrade
```
3. Instala los paquetes necesarios para el funcionamiento de BIND (bind9).
```
root@debian-servidor-fp:~# apt-get install bind9 bind9utils
```
***NOTA**: La instalación crea el usuario bind que ejecuta el servicio dns denominado named.*
4. Verifica que el servidor **bind9** está activo.
```
root@debian-servidor-fp:~# service bind9 status
bind9 is running.
root@debian-servidor-fp:~# /etc/init.d/bind9 status
bind9 is running.
```
5. Verifica en qué puertos (*Número utilizado en las comunicaciones cliente/servidor, en transmisiones TCP o UDP, comprendido entre 1 y 65535, que indica por dónde tiene lugar la conexión con un servidor. Están estandarizados, esto es, un servidor suele estar activo siempre por definición en un puerto determinado, pero éste puede que sea modificado en la configuración del servidor. Por ejemplo, un servidor web espera en el puerto TCP 80*) TCP y UDP está activo el servidor **bind9**, para ello comprueba el servicio **named**:
```
root@debian-servidor-fp:~# netstat -natp | grep named
tcp 0 0 192.168.200.250:53 0.0.0.0:* LISTEN 1442/named
tcp 0 0 127.0.0.1:53 0.0.0.0:* LISTEN 1442/named
tcp 0 0 127.0.0.1:953 0.0.0.0:* LISTEN 1442/named
tcp6 0 0 :::53 :::* LISTEN 1442/named
tcp6 0 0 ::1:953 :::* LISTEN 1442/named
root@debian-servidor-fp:~# netstat -naup | grep named
udp 0 0 192.168.200.250:53 0.0.0.0:* 1442/named
udp 0 0 127.0.0.1:53 0.0.0.0:* 1442/named
udp6 0 0 :::53 :::* 1442/named
```

Para mantenerte informado y actualizado es recomendable que visites la página oficial del servidor BIND.

http://www.isc.org/software/bind

1.11.1.- Archivos de configuración del servidor DNS.

Tras la instalación del servidor DNS BIND (**bind9**) existe la ruta `/etc/bind`, la cual contiene sus ficheros de configuración. Una estructura tipo de `/etc/bind` que puedes encontrar al instalar bind sería similar a la que se muestra en la siguiente imagen:

```
root@debian-servidor-fp:~# tree /etc/bind
/etc/bind
├── bind.keys
├── db.0
├── db.127
├── db.255
├── db.empty
├── db.local
├── db.root
├── named.conf
├── named.conf.default-zones
├── named.conf.local
├── named.conf.options
├── rndc.key
└── zones.rfc1918

0 directories, 13 files
root@debian-servidor-fp:~#
```

El servidor DNS BIND (**bind9**) posee por defecto en su instalación el fichero `/etc/bind/named.conf`, que contiene la configuración principal, de la que beben todos los demás ficheros de configuración. En su contenido puedes ver las siguientes líneas, que añaden la configuración de

determinados ficheros a la configuración principal, dedicados a particularizar la misma:

```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

Donde,

/etc/bind/named.conf.options: hace referencia al archivo de configuración que posee opciones genéricas.

/etc/bind/named.conf.local: hace referencia al archivo de configuración para opciones particulares.

/etc/bind/named.conf.default-zones: hace referencia al archivo de configuración de zonas.

Dentro de cada uno de estos archivos encontrarás partes de código agrupadas entre llaves que finalizan con el carácter punto y coma (**;**), conocidos como declaraciones, las cuales indicarán secciones de ejecución. Cualquier código en un archivo de configuración que comience con los caracteres doble barra (**//**), almohadilla (**#**) o aparezca encerrado entre barra asterisco (**/***) y asterisco barra (***/**) son considerados comentarios y por lo tanto no se ejecuta.

Puedes modificar los ficheros de configuración a tu antojo. Así, puedes crear incluso nuevos ficheros de configuración que sean llamados desde otros mediante la directiva **include**.

Elementos que puedes emplear en los ficheros de configuración, de la versión BIND 9.7 empleada en esta unidad, los puedes encontrar en la documentación oficial en formato HTML sobre BIND

http://ftp.isc.org/isc/bind9/cur/9.7/doc/arm/Bv9ARM.ch06.html

Puedes realizar una verificación de los ficheros de configuración y de zona por posibles fallos mediante los comandos "**named-checkconf**" y "**named-checkzone**" respectivamente. Estos comandos suelen ejecutarse con la siguiente sintaxis:

```
named-checkconf [-p] {filename}
```

donde,

named-checkconf → comprueba la sintaxis pero no la semántica de un fichero de configuración **named**. El fichero se analiza y comprueba por errores de sintaxis, junto con todos los archivos incluidos en él. Si no se especifica ningún fichero, por defecto se comprueba **/etc/named.conf**.

-p → imprime la salida de **named.conf** y los ficheros incluidos en forma canónica si no fueron detectados errores.

filename → El nombre del archivo de configuración que desea comprobar. Si no se especifica, por defecto es **/etc/named.conf**.

```
named-checkzone {zonename} {filename}
```

donde,

named-checkzone → comprueba la sintaxis y la integridad de un archivo de zona. Realiza las mismas comprobaciones que **named** hace al cargar una zona. Esto hace que sea útil para comprobar los archivos de zona antes de configurarlos en un servidor de nombres.

zonename → El nombre de dominio de la zona que se comprueba.

filename → El nombre del archivo de zona.

Ejemplos de ejecución:

1. Verificar archivo de configuración

```
/etc/bind/named.conf:
root@debian-servidor-fp:/etc/bind# named-checkconf -p  /etc/bind/named.conf
```

2. Verificar el dominio de zona ejemplo.com en el archivo de zona

```
/var/lib/bind/master/db.ejemplo.com.hosts
root@debian-servidor-fp:/etc/bind#              named-checkzone              ejemplo.com
/var/lib/bind/master/db.ejemplo.com.hosts
```

1.11.2.- Arranque y parada del servidor DNS.

En un sistema operativo Debian 6.0 (Squeeze) puedes comprobar el estado del servicio bind mediante el comando `service` o mediante el comando `/etc/init.d/bind`:

✓ **Comando `service`:**

1. Comprobar las opciones del comando:

```
root@debian-servidor-fp:~# service bind9
Usage: /etc/init.d/bind9 {start|stop|reload|restart|force-reload|status}.
```

Donde,

`start` → opción que permite arrancar el servicio.

`stop` → opción que permite apagar el servicio.

`reload` → opción que permite recargar la configuración del servicio sin tener que reiniciarlo.

`restart` → opción que permite reiniciar el servicio.

`force-reload` → opción que permite forzar la recarga de configuración del servicio.

`status` → opción que permite comprobar si el servicio está activo o inactivo.

2. Arrancar el servidor DNS:

```
root@debian-servidor-fp:~# service bind9 start
Starting domain name service...: bind9.
```

3. Parar el servidor DNS:

```
root@debian-servidor-fp:~# service bind9 stop
Stopping domain name service...: bind9 waiting for pid 1989 to die.
```

4. Comprobar el estado activo/inactivo del servicio:

```
root@debian-servidor-fp:~# service bind9 status
could not access PID file for bind9 ... failed!
```

Como puedes comprobar la salida del comando determina que el servicio está inactivo, entonces lo arrancamos:

```
root@debian-servidor-fp:~# service bind9 start
Starting domain name service...: bind9.
```

Se vuelve a lanzar el comando para comprobar de nuevo el estado:

```
root@debian-servidor-fp:~# service bind9 status
bind9 is running.
```

Ahora puedes comprobar que la salida del comando determina que el servicio está activo.

✓ **Comando `/etc/init.d/bind9`:**

1. Comprobar las opciones del comando:

```
root@debian-servidor-fp:~# /etc/init.d/bind9
Usage: /etc/init.d/bind9 {start|stop|reload|restart|force-reload|status}.
```

2. Arrancar el servidor DNS:

```
root@debian-servidor-fp:~# /etc/init.d/bind9 start
Starting domain name service...: bind9.
```

3. Parar el servidor DNS:

```
root@debian-servidor-fp:~# /etc/init.d/bind9 stop
Stopping domain name service...: bind9 waiting for pid 2061 to die.
```

4. Comprobar el estado activo/inactivo del servicio:

```
root@debian-servidor-fp:~# /etc/init.d/bind9 status
could not access PID file for bind9 ... failed!
```

Como puedes comprobar la salida del comando determina que el servicio está inactivo, entonces lo arrancamos:

```
root@debian-servidor-fp:~# /etc/init.d/bind9 start
Starting domain name service...: bind9.
```

Se vuelve a lanzar el comando para comprobar de nuevo el estado:

```
root@debian-servidor-fp:~# /etc/init.d/bind9 status
bind9 is running.
```

Ahora puedes comprobar que la salida del comando determina que el servicio está activo.

1.11.3.- Configuración como caché DNS.

"Buena memoria es la escritura, pues para siempre dura."

Proverbio español

Todos los servidores DNS son servidores caché, pero no por ello deben ser maestro o esclavo. Así, existe la posibilidad que un servidor DNS funcione solamente como servidor caché, sin que sea maestro o esclavo.

En GNU/Linux Debian 6.0 (Squeeze) la configuración de un servidor DNS BIND (**bind9**) como caché viene establecida en el archivo `/etc/bind/named.conf.options`, donde se indica: el directorio de caché y los servidores DNS a reenviar las peticiones que no se pueden resolver de forma local mediante la caché: los servidores **forwarders**, para que luego estas consultas se vayan guardando en la caché.

El directorio de caché, `/var/cache/bind`, está configurado y habilitado por defecto tras la instalación y los servidores DNS a reenviar las peticiones que no se pueden resolver de forma local mediante la caché, los servidores **forwarders**, aparecen en una sección del mismo nombre y que por defecto está comentada, esto es, deshabilitada.

Para activar la caché debes realizar el siguiente procedimiento:

1. Verifica que el contenido del fichero `/etc/bind/named.conf.options`, tras la instalación, es el siguiente:

```
options {
 directory "/var/cache/bind";

 // If there is a firewall between you and nameservers you want
 // to talk to, you may need to fix the firewall to allow multiple
 // ports to talk. See http://www.kb.cert.org/vuls/id/800113

 // If your ISP provided one or more IP addresses for stable
 // nameservers, you probably want to use them as forwarders.
 // Uncomment the following block, and insert the addresses replacing
 // the all-0's placeholder.

 //forwarders {
 0.0.0.0;
 };

 auth-nxdomain no; # conform to RFC1035
 listen-on-v6 { any; };
};
```

2. Modificas el fichero `/etc/resolv.conf` para que solamente tenga activa la siguiente línea:

```
nameserver 127.0.0.1
```

   De tal forma que ahora el servidor DNS activo solamente es el local, que tienes configurado como caché.

3. Una vez efectuados los cambios recargas el servidor con el comando: `service bind9 reload` ó `/etc/init.d/bind9 reload`.

Puedes realizar una comprobación del funcionamiento de la caché, si una vez realizado lo expuesto, sigues el procedimiento que encontrarás en el

ANEXO II - Comprobar funcionamiento servidor DNS BIND

El borrado de la caché DNS la puedes realizar en el cliente DNS y en el propio servidor DNS. Así, para un sistema operativo GNU/Linux podrías realizar los siguientes comandos según el caso:

1. Borrado de la caché del cliente DNS:

```
# /etc/init.d/nscd restart
```

2. Borrado de la caché del servidor DNS BIND:

```
# /usr/sbin/rndc flush
```

1.11.4.- Configuración como DNS maestro.

"El maestro sabe lo que hace."

Proverbio español

En GNU/Linux Debian 6.0 (Squeeze) puedes configurar un servidor DNS BIND como maestro modificando el archivo **`/etc/bind/named.conf.local`** realizando el siguiente procedimiento:

1. Configuras el fichero `/etc/bind/named.conf.local` para indicar: qué zonas son servidas por el servidor, qué zonas son servidas como master y el fichero donde se guarda el contenido de la zona. Por ejemplo:

```
//zonas creadas tipo master
zone "ejemplo.com" {
 type master;
 file "/var/lib/bind/master/db.ejemplo.com.hosts";
};
```

   En este ejemplo, el servidor sirve el dominio "`ejemplo.com`" como master, y la zona se guarda en el fichero **`/var/lib/bind/master/db.ejemplo.com.hosts`**.
   Habrá una entrada de este tipo por cada zona servida.
   Normalmente los ficheros de zona están situados en la ruta `/var/lib/bind`. Entonces, para una mayor comprensión y entendimiento, y para facilidad de uso en posteriores momentos, estaría bien que crearas los directorios master y slave dentro de esa ruta. Así, los ficheros con zonas maestras se pueden encontrar en `/var/lib/bind/master/db.*.hosts` y los ficheros con zonas esclavas se pueden encontrar en `/var/lib/bind/slave/db.*.hosts`.
2. Configuras el fichero `/var/lib/bind/master/db.ejemplo.com.hosts` para agregar los registros RR a la zona, por ejemplo:

```
;
; BIND Database file for ejemplo.com zone
;

@ IN SOA ejemplo.com. hostmaster.ejemplo.com. (
 2011091601 ; serial number
 3600 ; refresh
 600 ; retry
 1209600 ; expire
 3600 ) ; default TTL
;
 IN NS ns.ejemplo.com.
 IN MX 10 mail.ejemplo.com.
 IN TXT ( "v=spf1 mx ~all" )
;

localhost A 127.0.0.1
ns A 192.168.200.250
mail A 192.168.200.251
www A 192.168.200.252
```

3. Recargas el servidor con el comando: `service bind9 reload` ó `/etc/init.d/bind9 reload`.
4. Realizas la siguiente consulta: `dig ejemplo.com` obteniendo una salida similar a la siguiente:

```
; <<>> DiG 9.7.3 <<>> ejemplo.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25588
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;ejemplo.com. IN A

;; AUTHORITY SECTION:
ejemplo.com. 3600 IN SOA ejemplo.com. hostmaster.ejemplo.com. 2011091601 3600 600 1209600
3600

;; Query time: 3 msec
```

```
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Sep 27 11:48:36 2011
;; MSG SIZE rcvd: 76
```

¿Cuál sería mediante `dig` el comando que deberías ejecutar para obtener la IP del servidor de correo correspondiente al dominio `ejemplo.com`, si el servidor de correo posee el nombre `mail`?

```
dig mail.ejemplo.com
```

1.11.5.- Configuración como DNS esclavo.

"Hace mal quien lo secundario hace principal."

Proverbio español

En GNU/Linux Debian 6.0 (Squeeze) puedes configurar un servidor DNS BIND como esclavo modificando el archivo **`/etc/bind/named.conf.local`** realizando el siguiente procedimiento:

1. Configuras el fichero `/etc/bind/named.conf.local` del servidor esclavo para indicar: qué zonas son servidas por el servidor, qué zonas son servidas como slave, la IP del servidor master (*de donde se transferirá la zona cuando se reciba una notificación de cambio, o se supere el TTL de la zona*) y el fichero donde se guarda el contenido de la zona. Por ejemplo:

```
//zonas creadas tipo esclavo
zone "ejemplo.com" {
 type slave;
 masters {
 192.168.200.250;
 };
 file "/var/lib/bind/slave/db.ejemplo.com.hosts";
};
```

   En este ejemplo, el servidor sirve el dominio "**`ejemplo.com`**" como slave, y la zona se guarda en el fichero **`/var/lib/bind/slave/db.ejemplo.com.hosts`**.
   Habrá una entrada de este tipo por cada zona servida.
   Normalmente los ficheros de zona están situados en la ruta `/var/lib/bind`. Entonces, para una mayor comprensión y entendimiento, y para facilidad de uso en posteriores momentos, estaría bien que crearas los directorios master y slave dentro de esa ruta. Así, los ficheros con zonas maestras se pueden encontrar en `/var/lib/bind/master/db.*.hosts` y los ficheros con zonas esclavas se pueden encontrar en `/var/lib/bind/slave/db.*.hosts`.
2. En el servidor maestro configuras la sección correspondiente al servidor master en el fichero `/etc/bind/named.conf.local`:
   1. Para indicar qué servidores tienen permitido la transferencia de los ficheros de zona, mediante la directiva `allow-transfer`. Por ejemplo:

```
allow-transfer{192.168.200.100;192.168.210.100;10.10.42.41;10.10.42.42;};
```

      En este listado deberán estar incluidos todos los servidores slave que tengan configurado a éste como servidor master, y adicionalmente alguna IP que debiera tenerlo permitido por alguna razón.
   2. Mediante la directiva `notify-yes` se consigue enviar automáticamente una notificación de cambio de zona del maestro, cuando ésta se produce, a los servidores DNS especificados en la zona mediante el registro de recurso NS.
      Adicionalmente, se puede enviar una notificación de cambio de zona a servidores esclavos que no aparecen en la misma, mediante la directiva **`also-notify`**:

```
also-notify {192.168.200.100;10.10.42.41;};
```

      Por ejemplo, una zona tipo master con las directivas anteriores podría ser la siguiente:

```
//zonas creadas tipo master
   zone "ejemplo.com" {
      type master;
      file "/var/lib/bind/master/db.ejemplo.com.hosts";
      allow-transfer{192.168.200.100;192.168.210.100;10.10.42.41;10.10.42.42;};
```

```
        notify-yes;
        also-notify {192.168.200.100;10.10.42.41;};
    };
```

Mediante la directiva `also-notify` se mantienen los servidores DNS sincronizados. Así, el servidor DNS esclavo podrá satisfacer las peticiones DNS al igual que lo haría el maestro. Esto implica que se garantiza la disponibilidad del servicio DNS puesto que aunque el servidor maestro deje de funcionar, el servidor esclavo podrá seguir ofreciendo el servicio. Además, en caso de recibir múltiples conexiones concurrentes, siendo, por tanto, el número de peticiones muy elevado, la carga se distribuye entre los servidores.

Puedes visitar el siguiente enlace donde encontrarás la documentación ofrecida sobre el servidor DNS BIND en su página oficial.

http://www.isc.org/software/bind/documentation

2.- Servicio de directorio.

Caso práctico

*A BK Programación, una empresa, con la que ya han trabajado anteriormente en proyectos asignados a **Juan**, les ha encargado un proyecto con las siguientes especificaciones para el departamento de atención al cliente:*

1. *Controlar el acceso de usuarios a los equipos de la empresa, de tal forma que, independientemente del ordenador con el que trabajen en la empresa, mediante autenticación de usuario y contraseña, puedan tener acceso al mismo.*
2. *Controlar el acceso de usuarios a la herramienta de gestión de incidencias y proyectos.*

Para ello BK Programación ha determinado realizar una autenticación por LDAP mediante OpenLDAP, puesto que aunque la configuración y el tiempo empleado va a ser más costoso que empleando otras alternativas, determina que la empresa necesita una centralización de esa base de datos de usuarios para que la aplicación de gestión de incidencias y proyectos, y los equipos ofrecidos por la empresa a sus trabajadores, puedan beber de la misma fuente: la base de datos de OpenLDAP.

Seguramente habrás utilizado más de una vez algún tipo de directorio o servicio de directorio, como por ejemplo: una guía telefónica impresa en papel o una revista con la programación televisiva.

Los directorios, por lo tanto, permiten localizar información y para ello definen qué información se almacenará y en qué modo.

Los directorios anteriormente comentados presentan una serie de problemas, en contra de los directorios electrónicos, a saber:

1. Son estáticos:
    1. Cuando buscas un teléfono en la guía telefónica, la información más actualizada es la de la fecha de edición impresa de la guía. Esto quiere decir que si una persona modifica sus datos o da de alta una nueva línea no aparecerán los cambios hasta la próxima edición impresa.
    2. En el caso de la programación televisiva, el tiempo de renovación de la información se reduce, posiblemente sea semanalmente. Pero, ¿y si existe algún cambio de última hora en el medio de la semana? La información también quedaría obsoleta.
       Por contra, los directorios electrónicos pueden ser consultados/actualizados en tiempo real, por lo que su fiabilidad es mucho mayor.
2. Son inflexibles: en el contenido y en su organización:
    1. ¿Qué pasaría si en la guía telefónica quisieras introducir una nueva información sobre el propietario de un teléfono? Está claro que la visualización del nuevo contenido no es instantáneo, habría que modificarlo y el usuario debería esperar a la nueva edición.
    2. ¿Qué pasaría si en la programación televisiva se quisiera incorporar un nuevo logotipo de una cadena de televisión? Pues, lo mismo que comentado anteriormente.
       Por contra, los directorios electrónicos pueden modificar cualquier contenido y éste se verá reflejado al instante.
    3. ¿Qué pasaría si quisieras buscar en la guía telefónica un teléfono por la calle donde vive el usuario?
    4. ¿Qué pasaría si en la programación televisiva quisieras buscar todas las películas sobre acción que se emiten a una determinada hora, pero solamente los días a la semana que te interese? Puede ser que encuentres esa información pero, desde luego, no es muy flexible la búsqueda de la misma. Por contra, los directorios electrónicos permiten que la búsqueda de información sea localizada de distintas maneras, gracias a cómo está organizada.
3. Son inseguros: dificultad para controlar el acceso a la información.
    1. ¿Cómo impides que un usuario no pueda buscar un teléfono en la guía telefónica?
    2. ¿Cómo impides que un menor pueda leer cierto contenido sobre, por ejemplo, un programa no educativo?

Los directorios electrónicos sí permiten controlar el acceso a la información: solamente aquel que disponga de las claves de acceso obtendrá la información.

4. Difícilmente configurable:
   1. ¿Cómo hacer en la guía telefónica para realizar una búsqueda solamente sobre un segundo apellido, de una zona urbana y con teléfonos que poseen dos números que tú determines?
   2. ¿Cómo hacer en la programación televisiva para realizar una búsqueda sobre cadenas por satélite para Europa que emitan en franjas horarias determinadas deportes y, a poder ser, torneos o ligas profesionales?
      Bien, parece ser que manejar tanta cantidad de información para ser ofrecida en ese tipo de búsquedas la hace no impresa e incluso inmanejable. Por contra, los directorios electrónicos pueden establecer la información que recibe una persona en función de sus necesidades.

2.1.- ¿Para qué usar un servicio de directorio?

"Siempre deseé que mi computadora fuera tan fácil de usar como mi teléfono. Mi deseo se ha hecho realidad: ya no sé usar mi teléfono."

Bjarne Stroustrup

Por lo visto anteriormente los directorios electrónicos permiten, de forma eficiente:

1. Encontrar información:
   Los directorios electrónicos a diferencia de los clásicos permiten acceder a la información contenida en los mismos de múltiples formas. Así, comparando con la guía telefónica tradicional, un directorio electrónico permite realizar búsquedas, no solamente por orden alfabético, sino también por: apellido: dirección, teléfono… ¿Cómo realizarías una búsqueda por teléfono en una guía telefónica tradicional?
   Es más, podrías sumar campos de búsqueda, como por ejemplo: dirección y apellido.
2. Gestionar información:
   En los directorios electrónicos pueden existir varios usuarios que en tiempo real estén realizando modificaciones, como agregar/editar/eliminar distintos usuarios con sus correspondientes campos. Además, esta información ya estaría visible para todas aquellas aplicaciones que accedan a la misma. Centralizar así los datos en un directorio evita tener que sincronizar varios directorios, con el consiguiente riesgo que esto provoca, pues: ¿qué pasaría si la sincronización no tuvo lugar y una aplicación accede a los datos? Pues sí, obtendría los datos no actualizados, o error en los mismos.
   Un caso muy común es el de los servidores Web con autenticación: si solamente dispones de un servidor web la solución es sencilla, puesto que solamente se necesitaría actualizar una base de datos de usuarios, pero ¿y si dispones de más de un servidor web que debe acceder a la misma base de datos? Entonces, la cosa se complica, puesto que debes sincronizar a los distintos servidores. Es más, y si esa base de datos la quisiéramos aprovechar para ofrecer otro servicio distinto del de los servidores web? Pues, todo el trabajo no sería aprovechable, y por lo tanto sería mejor desde un principio adaptar este sistema a los servicios de directorios.
3. Control de seguridad:
   Los servicios de directorios no simplemente permiten delimitar el acceso a los usuarios, sino que también proporcionan una solución al problema de gestión de certificados digitales. Así, permiten:
   1. **Su creación**: Incorporar a los certificados los datos contenidos en el directorio.
   2. **Su distribución**: Tener accesibles mediante un protocolo estándar los certificados.
   3. **Su destrucción**: Revocar los certificados de forma sencilla simplemente borrando el certificado del directorio.
   4. **Su ubicación**: Los usuarios pueden acceder a través del directorio a los certificados de los restantes usuarios, de forma muy sencilla y fácil de integrar con las aplicaciones.

Por todo ello las aplicaciones prácticas que poseen los servicios de directorio son muy diversas y ventajosas, como por ejemplo: autenticación de usuarios: en aplicaciones web, correo electrónico, RADIUS (*protocolo de autenticación y autorización para aplicaciones de acceso a la red o movilidad IP*)..., sistemas de control de entradas a edificios, bases de datos comunes en organizaciones, en sistemas operativos: gestión de cuentas de acceso, servidores de certificados, libretas de direcciones compartidas...

2.2.- Directorio vs DNS.

Tanto un servicio de directorio como un servicio DNS proporcionan acceso a una base de datos jerárquica, pero difieren en:

1. Los servidores de directorio no están particularizados a una acción concreta, sino orientados de forma más general, mientras que el servicio DNS está dedicado a la traducción de nombres de dominios a direcciones IP.
2. La información almacenada en el servicio de directorio no es fija, mientras que en el servicio DNS tiene una estructura fija.
3. El servicio de directorio permite actualizaciones, mientras que el servicio DNS no las permite, ¿o puedes actualizar a tu antojo los servidores raíz DNS?
4. Los servicios de directorio suelen utilizar protocolos orientados a conexión (**TCP**), mientras que el servicio DNS opera con protocolos no orientados a conexión (**UDP**).

Pero, no por ello, poseen el impedimento de trabajar juntos, es más, usualmente los encontrarás unidos de la mano en aplicaciones web con distintas funcionalidades, como: servidores de correo, gestión de proyectos e incidencias, servidores RADIUS, etc. Así, suele ser necesario acceder a las URL de las aplicaciones web mediante nombres de dominio DNS y una vez en ellas autenticarse por medio de LDAP.

Antes de intentar configurar una aplicación web con autenticación LDAP deberías probar la instalación, configuración y autenticación por medio de una base de datos SQL.
En el siguiente anexo encontrarás información detallada sobre el procedimiento de instalación, configuración y autenticación por medio de una base de datosSQL de la aplicación web OpenCart.

ANEXO III - Ejemplo despliegue aplicación web OpenCart

2.3.- Organización del directorio LDAP.

El servicio de directorio puede estar centralizado o distribuido:

- ✓ **Centralizado**: En este caso un único servidor ofrece todo el servicio de directorio respondiendo a todas las consultas de los clientes.
- ✓ **Distribuido**: Si el directorio está distribuido, varios servidores proporcionan el servicio de directorio. Cuando está distribuido, los datos pueden estar fraccionados y/o replicados:
  - ➔ Cuando está 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 está 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.

En 1988, la CCITT (ahora ITU-T) creó el estándar X.500 (http://x500standard.com/) sobre servicios de directorio, el cual 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 protocolo DAP, pero DAP es un protocolo a nivel de aplicación, por lo que, tanto al cliente como el servidor debían implementar completamente la torre de protocolos OSI.

LDAP surge como una alternativa a DAP. Las claves del éxito de LDAP en comparación con DAP de X.500 son:

- ✓ El modelo funcional de LDAP es más simple y ha eliminado opciones raramente utilizadas en X.500, siendo más fácil de comprender e implementar.
- ✓ LDAP representa la información mediante cadenas de caracteres en lugar de complicadas estructuras ASN.1.

El directorio LDAP tiene una estructura en forma de árbol denominado `DIT`. Cada entrada del directorio describe un **objeto**: persona, impresora, etc. La ruta completa a una entrada la identifica de modo inequívoco y se conoce como `DN` y está compuesto por una secuencia de partes más pequeñas llamadas `RDN`, de forma similar a como el nombre de un fichero consiste en un camino de directorios en muchos sistemas operativos.

Una **clase de objeto (objectClass)** es una descripción general de un tipo de objeto. Todos los objetos de LDAP deben tener el atributo `objectClass`. La definición de `objectClass` especifica qué atributos requiere un objeto LDAP, así como las clases de objetos que pueden existir. Los valores de este atributo los pueden modificar los clientes, pero el atributo `objectClass` en sí no puede eliminarse.

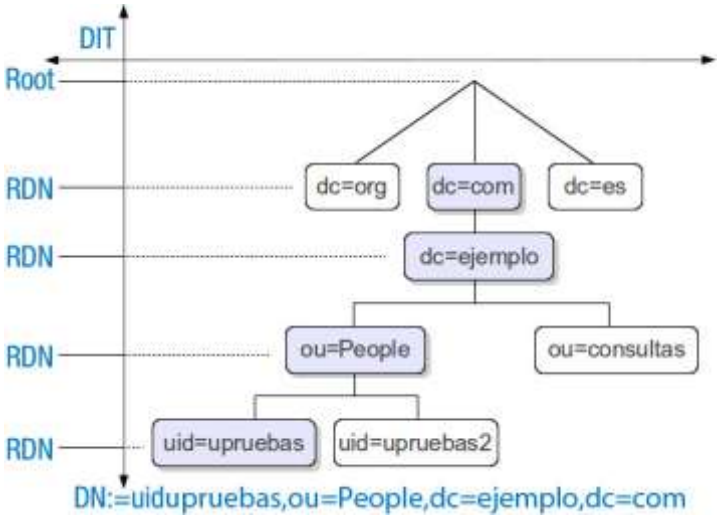
Un **esquema (schema)** define: qué clases de objetos se pueden almacenar en el directorio, qué atributos deben contener, qué atributos son opcionales y el formato de los atributos.

Por lo general, existen dos tipos de objetos:

- ✓ **Contenedor**: Este tipo de objeto puede contener a su vez otros objetos. Algunos ejemplos de estos elementos son: `Root` (elemento raíz del árbol de directorios que no existe en realidad), `c` (country), `ou` (OrganizationalUnit) y `dc` (domainComponent).
  La figura análoga al contenedor es el directorio (carpeta) de un sistema de archivos.
- ✓ **Hoja**: Este tipo de objeto se encuentra al final de una rama y carece de objetos subordinados. Algunos ejemplos son: `Person/InetOrgPerson` o `groupofNames`.

En la cúspide de la jerarquía del directorio se encuentra el elemento raíz `Root`. A este elemento le puede seguir en un nivel inferior `c` (country), `dc` (domainComponent) ó `o` (organization).

La siguiente imagen ilustra las relaciones jerárquicas dentro de un árbol de directorios LDAP:



La figura representa un `DIT` ficticio con entradas en cuatro niveles. Cada entrada se corresponde con una casilla en la figura. En este caso, el nombre válido completo `DN` del empleado ficticio `upruebas` es:

```
dn: uid=upruebas,ou=People,dc=ejemplo,dc=com
```

La definición global de qué tipo de objetos han de guardarse en el DIT se realiza mediante un ***esquema***. El tipo de objeto se determina mediante la *clase de objeto*. La clase de objeto especifica qué atributos *deben* o *pueden* ser asignados a un objeto determinado. Por lo tanto, un esquema

debe contener definiciones de todas las clases de objetos y atributos que van a utilizarse en el escenario de aplicación. Existen algunos esquemas de uso extendido (véase **RFC 2252** y **2256**). No obstante, si el entorno en el que va a utilizarse el servidor LDAP lo requiere, también pueden crearse nuevos esquemas en función del usuario o pueden combinarse varios esquemas entre sí.

2.4.- Integración del servicio de directorio con otros servicios.

De lo expuesto anteriormente puede deducirse que el servicio de directorio es importante en sí mismo, pero es fundamental para aglutinar información que puede ser fuente de objeto para desplegar nuevos servicios basados en la cooperación entre las distintas aplicaciones y el servicio de directorio.

Así, 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.

Puedes 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 acceso, 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, con la precisión que otras herramientas no pueden generar.

Debido a 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.

2.5.- El formato de intercambio de datos LDIF.

El formato LDIF es el estándar para representar entradas del directorio en formato texto ASCII (*código de caracteres basado en el alfabeto latino. ASCII es, en sentido estricto, un código de siete bits, lo que significa que usa cadenas de bits representables con siete dígitos binarios (que van de 0 a 127 en base decimal) para representar información de caracteres. El código ASCII define así una relación entre caracteres específicos y secuencias de bits*), que posee la siguiente sintaxis:

```
dn: <nombre distinguido>
<nombre_atributo>: <valor>
<nombre_atributo>: <valor>
<nombre_atributo>: <valor>
```

Entonces, una entrada del directorio en formato de intercambio de datos LDIF consiste en dos partes:

- ✓ El `DN` que debe figurar en la primera línea de la entrada y que se compone de la cadena `dn:` seguida del nombre distinguido (`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 existe ningún orden preestablecido para la colocación de los atributos, pero es conveniente listar primero el atributo **`objectclass`**, para mejorar la legibilidad de la entrada.

En un archivo LDIF puede haber más de una entrada definida, cada entrada se separa de las demás por una línea en blanco. A su vez, cada entrada puede tener una cantidad arbitraria de pares **`<nombre_atributo>: <valor>`**.

Este formato es útil tanto para realizar copias de seguridad de los datos de un servidor LDAP, como para importar pequeños cambios que se necesiten realizar manualmente en los datos, siempre manteniendo la independencia de la implementación LDAP y de la plataforma donde esté instalada.

A continuación puedes observar un ejemplo de una entrada para describir una cuenta de usuario en un servidor:

```
dn: uid=upruebas,ou=People,dc=ejemplo,dc=com
objectclass: account
objectclass: posixAccount
objectclass: topuid: upruebas
cn: Usuario Pruebas
loginshell: /bin/bash
uidnumber: 512
gidnumber: 300
homedirectory: /home/upruebas
gecos: Usuario Pruebas
userpassword: 123456
```

2.6.- Instalación de OpenLDAP.

El proceso de instalación de OpenLDAP en un sistema basado en Debian es sencillo, no tanto, como verás, será la configuración.

Para una instalación de OpenLDAP en Debian 6 (squeeze) realiza el siguiente procedimiento como usuario **`root`**, teniendo en cuenta que el servidor está identificado como sigue:

- ✓ Hostname: **`debian-servidor-fp`**
- ✓ IP: **`192.168.200.250`**

1. Actualiza los repositorios del sistema operativo.
   ```
   root@debian-servidor-fp:~# apt-get update
   ```
   NOTA: es necesario para el buen funcionamiento del comando que tengas configurado correctamente la conexión a Internet.
2. Actualiza el sistema operativo.
   ```
   root@debian-servidor-fp:~# apt-get upgrade
   ```
3. Instala los paquetes necesarios para el funcionamiento de OpenLDAP. La instalación te pedirá una contraseña, como puedes ver a
   ```
   root@debian-servidor-fp:~# apt-get install slapd ldap-utils
   Contraseña del administrador: admin
   Verificación de la contraseña: admin
   ```
4. Verifica que el servidor OpenLDAP está activo, por defecto, en el puerto **TCP 389**.
   ```
   root@debian-servidor-fp:~# netstat -natp | grep 389
   tcp 0 0 0.0.0.0:389 0.0.0.0:* LISTEN 1955/slapd
   tcp6 0 0 :::389 :::* LISTEN 1955/slapd
   ```

Es recomendable que visites el siguiente anexo que contiene como instalar y configurar un servidor OpenLDAP en un GNU/Linux basado enDebian.

ANEXO IV - INSTALACIÓN Y CONFIGURACIÓN DEL SERVIDOR OPENLDAP

2.6.1.- Configuración de OpenLDAP.

Una de las principales novedades de la versión 2.4 de OpenLDAP es que se incluye toda la configuración del servidor `slapd` en un directorio de base `cn=config`, (`/etc/ldap/slapd.d/cn=config`), en lugar del habitual fichero `/etc/ldap/slapd.conf`. Esto tiene la ventaja de que las modificaciones de configuración se pueden hacer sin tener que reiniciar el servicio.

Dentro del directorio `/etc/ldap/slapd.d/cn=config`, en una instalación limpia, puedes observar el objeto `cn=schema`, donde se encuentran los cuatro esquemas instalados por defecto: `core`, `cosine`, `nis` e `inetorgperson`.

En el siguiente enlace puedes encontrar información sobre las especificaciones del esquema (schema) en OpenLDAP.

http://www.openldap.org/doc/admin24/schema.html

Puedes encontrar más esquemas dentro de `/etc/ldap/schema`. Para añadir un esquema nuevo al directorio hay que subir un fichero `ldif` con el nuevo esquema al `dn: cn=schema,cn=config`.

La tabla siguiente ofrece un resumen de las clases de objetos utilizadas en el ejemplo de `core.schema` e `inetorgperson.schema` junto con los atributos obligatorios y los valores adecuados de atributo.

Clase de objeto	Significado	Entrada de ejemplo	Atributo obligatorio
dcObject	domainComponent (partes del nombre del dominio).	ejemplo.	dc
organizationalUnit	organizationalUnit (unidad organizativa).	People.	ou
inetOrgPerson	inetOrgPerson (datos sobre personal para Internet/intranet).	Usuario Pruebas Pruebas.	cn ; sn

El árbol completo LDAP se genera a partir de archivos esquema, en `/etc/ldap/schema`, que definen el árbol de clases y atributos permitidos para la organización.

La configuración de OpenLDAP puede resultar ardua, así que ármate de paciencia y procede como se te indica a continuación.

1. Configura el servidor OpenLDAP mediante el comando `dpkg-reconfigure slapd`. Los valores utilizados los puedes ver a continuación del comando:

```
root@debian-servidor-fp:~# dpkg-reconfigure slapd
 ¿Desea omitir la configuración del servidor OpenLDAP? No
 Introduzca su nombre de dominio DNS: proyecto-empresa.local
 Nombre de la organización: proyecto-empresa.local
 Contraseña del administrador: admin
 Verificación de la contraseña: admin
 Motor de base de datos a utilizar: HDB
 ¿Desea que se borre la base de datos cuando se purgue el paquete slapd? No
 ¿Desea mover la base de datos antigua? Sí
 ¿Desea permitir el protocolo LDAPv2? Sí
```

http://www.youtube.com/watch?feature=player_embedded&v=HvSgWdVb_1k

2. Continuación de la configuración del servidor OpenLDAP. Edita el archivo `/etc/ldap/slapd.d/cn=config/olcDatabase\=\{1\}hdb.ldif` y cambia todas las cadenas '`dc=nodomain`' por '`dc=proyecto-empresa,dc=local`', similar a como se expone a continuación:

```
root@debian-servidor-fp:~#  cat  /etc/ldap/slapd.d/cn=config/olcDatabase\=\{1\}hdb.ldif  |
sed -e "s/dc=nodomain/dc=proyecto-empresa,dc=local/g" > a.txt
root@debian-servidor-fp:~# mv a.txt /etc/ldap/slapd.d/cn=config/olcDatabase\=\{1\}hdb.ldif
```

Puedes revisar a continuación el contenido tipo del fichero **`/etc/ldap/slapd.d/cn=config/olcDatabase\=\{1\}hdb.ldif`**

```
dn: olcDatabase={1}hdb
objectClass: olcDatabaseConfig
objectClass: olcHdbConfig
olcDatabase: {1}hdb
olcDbDirectory: /var/lib/ldap
olcSuffix: dc=proyecto-empresa,dc=local
olcAccess: {0}to attrs=userPassword,shadowLastChange by self write by anonymou
 s auth by dn="cn=admin,dc=proyecto-empresa,dc=local" write by * none
olcAccess: {1}to dn.base="" by * read
olcAccess: {2}to * by self write by dn="cn=admin,dc=proyecto-empresa,dc=local"
  write by * read
olcLastMod: TRUE
olcRootDN: cn=admin,dc=proyecto-empresa,dc=local
olcRootPW:: e1NTSEF9SzJlYzZxSmRIRitKR2poU2U0cTl2Z1BleG5LZUVvUTM=
olcDbCheckpoint: 512 30
olcDbConfig: {0}set cachesize 0 2097152 0
olcDbConfig: {1}set_lk_max_objects 1500
olcDbConfig: {2}set_lk_max_locks 1500
olcDbConfig: {3}set_lk_max_lockers 1500
olcDbIndex: objectClass eq
structuralObjectClass: olcHdbConfig
entryUUID: 2723694a-809c-1030-958c-4d95d1efe948
creatorsName: cn=admin,cn=config
createTimestamp: 20111001171131Z
entryCSN: 20111001171131.699703Z#000000#000#000000
modifiersName: cn=admin,cn=config
modifyTimestamp: 20111001171131Z
```

2.6.2.- Arranque y parada del servidor LDAP.

En un sistema operativo Debian 6.0 (Squeeze) puedes comprobar el estado del servicio OpenLDAP mediante el comando service o mediante el comando **`/etc/init.d/slapd`**:

✓ **Comando `service`**:

1. Comprobar las opciones del comando:

```
root@debian-servidor-fp:~# service slapd
Usage: /etc/init.d/slapd {start|stop|reload|restart|force-reload|status}.
```

Donde:

`start` → opción que permite arrancar el servicio.

`stop` → opción que permite apagar el servicio.

`reload` → opción que permite recargar la configuración del servicio sin tener que reiniciarlo.

`restart` → opción que permite reiniciar el servicio.

`force-reload` → opción que permite forzar la recarga de configuración del servicio.

`status` → opción que permite comprobar si el servicio está activo o inactivo.

2. Arrancar el servidor OpenLDAP:

```
root@debian-servidor-fp:~# service slapd start
Starting OpenLDAP: slapd.
```

3. Parar el servidor OpenLDAP:

```
root@debian-servidor-fp:~# service bind9 stop
Stopping OpenLDAP: slapd.
```

4. Comprobar el estado activo/inactivo del servicio:

```
root@debian-servidor-fp:~# service slapd status
could not access PID file for slapd ... failed!
```

Como puedes comprobar la salida del comando determina que el servicio está inactivo, entonces lo arrancamos:

```
root@debian-servidor-fp:~# service slapd start
Starting OpenLDAP: slapd.
```

Se vuelve a lanzar el comando para comprobar de nuevo el estado:

```
root@debian-servidor-fp:~# service slapd status
slapd is running.
```

Ahora puedes comprobar que la salida del comando determina que el servicio está activo.

✓ Comando `/etc/init.d/slapd`:

1. Comprobar las opciones del comando:

```
root@debian-servidor-fp:~# /etc/init.d/slapd
Usage: /etc/init.d/slapd {start|stop|reload|restart|force-reload|status}.
```

2. Arrancar el servidor OpenLDAP:

```
root@debian-servidor-fp:~# /etc/init.d/slapd start
Starting OpenLDAP: slapd.
```

3. Parar el servidor OpenLDAP:

```
root@debian-servidor-fp:~# /etc/init.d/slapd stop
Stopping OpenLDAP: slapd.
```

4. Comprobar el estado activo/inactivo del servicio:

```
root@debian-servidor-fp:~# /etc/init.d/slapd status
could not access PID file for slapd ... failed!
```

Como puedes comprobar la salida del comando determina que el servicio está inactivo, entonces lo arrancamos:

```
root@debian-servidor-fp:~# /etc/init.d/slapd start
Starting OpenLDAP: slapd.
```

Se vuelve a lanzar el comando para comprobar de nuevo el estado:

```
root@debian-servidor-fp:~# /etc/init.d/slapd status
slapd is running.
```

Ahora puedes comprobar que la salida del comando determina que el servicio está activo.

El comando `slaptest` permite verificar la configuración del servidor OpenLDAP.

2.6.3.- Administrando un servidor LDAP:

OpenLDAP ofrece una serie de comandos para la administración de datos en el directorio LDAP, contenidos en el paquete **`ldap-utils`**. Los cuatro comandos más importantes para añadir, modificar, buscar y eliminar son explicados a continuación.

http://www.youtube.com/watch?feature=player_embedded&v=vhRth4qvQOU

1. Añadir entradas: comando `ldapadd`.
   a. Crea la estructura básica del dominio LDAP mediante la ejecución de un fichero **`estructura_basica.ldif`**.

```
# Objetos raiz del dominio
dn: dc=proyecto-empresa,dc=local
objectClass: top
objectClass: dcObject
objectclass: organization
o: proyecto-empresa.local
dc: proyecto-empresa
description:  Raiz de dominio

# Usuarios
dn: ou=usuarios,dc=proyecto-empresa,dc=local
objectClass: organizationalUnit
ou: usuarios

# Grupos
dn: ou=grupos,dc=proyecto-empresa,dc=local
objectClass: organizationalUnit
ou: grupos
```

```
root@debian-servidor-fp:~# ldapadd -x -D cn=admin,dc=proyecto-empresa,dc=local -w
admin -f estructura basica.ldif
adding new entry "dc=proyecto-empresa,dc=local"
adding new entry "ou=usuarios,dc=proyecto-empresa,dc=local"
adding new entry "ou=grupos,dc=proyecto-empresa,dc=local"
```

   b. Añade un usuario a LDAP de nombre '`pruebas`' y contraseña '`123456`' mediante el archivo `usuario.ldif`.

```
# Usuario
```

```
dn: uid=upruebas,ou=usuarios,dc=proyecto-empresa,dc=local
objectClass: inetOrgPerson
objectClass: posixAccount
cn: Pruebas daw05
sn: daw05
loginShell: /bin/bash
uidNumber: 10001
gidNumber: 10001
homeDirectory: /home/upruebas
gecos: Pruebas DAW05
userPassword: 123456
mail: upruebas.daw05@proyecto-empresa.local
```

```
root@debian-servidor-fp:~# ldapadd -x -D cn=admin,dc=proyecto-empresa,dc=local -w
admin -f usuario.ldif
adding new entry "uid=upruebas,ou=usuarios,dc=proyecto-empresa,dc=local"
```

2. Modificar entradas: comando `ldapmodify`.
   a. Modificar la contraseña del usuario anterior '`pruebas`' mediante la ejecución del archivo **`cambiar_usuario.ldif`**.

```
# Cambiar contraseña Usuario
dn: uid=upruebas,ou=usuarios,dc=proyecto-empresa,dc=local
changetype: modify
replace: userPassword
userPassword: 654321
```

```
root@debian-servidor-fp:~# ldapmodify -x -D cn=admin,dc=proyecto-empresa,dc=local -w
admin -f cambiar_usuario.ldif
modifying entry "uid=upruebas,ou=usuarios,dc=proyecto-empresa,dc=local"
```

3. Buscar entradas: comando `ldapsearch`.
   a. Buscar todos los usuarios cuyo nombre contenga los caracteres '`pru`':

```
root@debian-servidor-fp:~# ldapsearch -x -b dc=proyecto-empresa,dc=local "(cn=*pru*)"
```

   b. Buscar todos los usuarios cuyo nombre contenga los caracteres '`pru`' y cuyo correo electrónico contengan los caracteres '`daw05`':

```
root@debian-servidor-fp:~# ldapsearch -x -b dc=proyecto-empresa,dc=local
"(&(cn=*pru*)(mail=*05*))"
```

4. Eliminar entradas: comando `ldapdelete`.
   a. Eliminar el usuario `upruebas`:

```
root@debian-servidor-fp:~# ldapdelete -x -D cn=admin,dc=proyecto-empresa,dc=local -w
admin uid=upruebas,ou=usuarios,dc=proyecto-empresa,dc=local
```

Los comandos anteriores poseen la opción **`-h`** con la cual se puede indicar el host (nombre de dominio o IP) que identifica al servidor LDAP. Por ejemplo: **`ldapsearch -h 192.168.200.250 -x -b dc=proyecto-empresa,dc=local "(objectclass=*)"`** conectaría con el servidor LDAP en la IP **`192.168.200.250`** para buscar el DIT del dominio **`proyecto-empresa.local`**.
Existe un paquete de nombre ldapscripts que contiene una serie de scripts para administrar de forma sencilla los usuarios y grupos almacenados en el servidor LDAP. Puedes encontrar plantillas de ejemplo, formato LDIF, situadas en **`/usr/share/doc/ldapscripts/examples/`** cuando se instala el paquete **`ldapscripts`**.

Una forma más sencilla de interactuar con el servidor OpenLDAP sería la posibilidad de gestionar el servidor mediante alguna interface gráfica, éstas existen tanto de pago como libres. A continuación se recoge varios enlaces que ofrecen información sobre estas interfaces gráficas (exploradores de directorios LDAP):

ANEXO V - Exploradores de directorios LDAP.
ANEXO VI - Administración de usuarios y grupos con LDAP

2.6.4.- Configuración de los clientes. Instalación de librerías de autentificación.

Como ya hemos comentado anteriormente, una de las utilidades más importantes de un servidor LDAP es la de servidor de autentificación. Autentificarse suele ser lo común y necesario para entrar en un sistema GNU/Linux. También para acceder a algunos servicios como un servidor FTP o a páginas privadas en un servidor web.

A continuación verás las modificaciones que hay que realizar en un sistema GNU/Linux Debian 6.0 (squeeze) para que autentifique a los usuarios en un servidor LDAP, esto es, verás los pasos a seguir para configurar un equipo como cliente LDAP. Así, el equipo en lugar de utilizar los clásicos archivos `/etc/passwd`, `/etc/group` y `/etc/shadow`, tomará los usuarios y grupos del servidor LDAP, autentificando los usuarios que inicien sesión validándose contra el servidor LDAP.

Esta configuración debe ser replicada en todos los clientes LDAP pertenecientes al dominio, incluido el propio servidor LDAP, si se quiere que los clientes accedan al mismo.

Para ello realiza el siguiente procedimiento:

1. Instala y configura los paquetes

```
libnss-ldap, libpam-ldap y nscd
root@debian-servidor-fp:~# apt-get install libnss-ldap libpam-ldap nscd
URI del servidor de LDAP: ldap://192.168.200.250
El nombre distintivo (DN) de la base de búsquedas: dc=proyecto-empresa,dc=local
Versión de LDAP a utilizar: 3
Cuenta LDAP para root: cn=admin,dc=proyecto-empresa,dc=local
Contraseña para la cuenta LDAP de root: admin
nsswitch.conf no se gestiona automáticamente
Debe modificar su fichero <</etc/nsswitch.conf>> … Aceptar
¿Desea permitir que la cuenta del administrador de LDAP se comporte como el administrador
local? Sí
¿Hacer falta un usuario para acceder a la base de datos de LDAP? No
Cuenta del administrador de LDAP: cn=admin,dc=proyecto-empresa,dc=local
Contraseña del administrador de LDAP: admin
```

   Toda esta configuración se ha guardado en el fichero **`/etc/libnss-ldap.conf`**

2. Modifica en el archivo `/etc/nsswitch.conf`:

```
2. /etc/nsswitch.conf::
passwd: files ldap
group: files ldap
shadow: files ldap
```

3. Reinicia el servicio nscd para que se activen los cambios efectuados en el paso anterior, esto es, para que el sistema operativo recoja los usuarios en primer lugar de los ficheros locales de usuarios y grupos y a continuación del servidor LDAP.

```
root@debian-servidor-fp:~# service nscd restart
Restarting Name Service Cache Daemon: nscd.
```

4. Revisa mediante el comando `pam-auth-update` que los servicios: `Unix authentication` y `LDAP Authentication`, que el sistema operativo usa para autenticar los usuarios, están activados.

```
root@debian-servidor-fp:~# pam-auth-update
Perfiles PAM a habilitar:
[*] Unix authentication
[*] LDAP Authentication
```

5. Por último, prueba que la configuración del cliente es correcta:
   1. Mediante el comando `getent passwd`, que proporciona todos los usuarios del sistema operativo, en este caso los de `Unix authentication` y `LDAP Authentication`.

```
root@debian-servidor-fp:~# getent passwd | grep uprueba
upruebas:*:10001:10001:Pruebas DAW05:/home/upruebas:/bin/bash
upruebas2:*:10002:10001:upruebas2:/home/upruebas2:/bin/bash
```

   2. Iniciar sesión en una consola de texto en el equipo cliente con un usuario del LDAP. En esta caso, con el usuario `upruebas` o el usuario `upruebas2`.

2.6.5.- Probar la autenticación con pamtest.

Ahora que la autentificación de usuarios por LDAP está activada en el sistema operativo, es recomendable que efectúes algunas pruebas con la nueva configuración para comprobar si todo funciona correctamente.

El comando `pamtest` puede ayudarte a realizar estas pruebas. La instalación del mismo se efectúa realizando el siguiente comando:

```
root@debian-servidor-fp:~# apt-get install libpam-dotfile
```

El comando **`pamtest`** acepta dos parámetros: el primero es el nombre del servicio al cual se va a conectar para realizar la autentificación y el segundo es el nombre del usuario que se va a autentificar sobre dicho servicio. Veamos unos ejemplos:

1. Intentar autentificar al usuario `upruebas2` en el servicio `passwd` mediante una clave correcta:

```
root@debian-servidor-fp:~# pamtest passwd upruebas2
Trying to authenticate <upruebas2> for service <passwd>.
Password:
Authentication successful.
```

2. Intentar autentificar al usuario `upruebas2` en el servicio `passwd` mediante una clave incorrecta:

```
root@debian-servidor-fp:~# pamtest passwd upruebas2
Trying to authenticate <upruebas2> for service <passwd>.
Password:
Failed to authenticate: Authentication failure
```

3. Intentar autentificar al usuario `upruebas2` en el servicio `ssh` mediante una clave correcta:

```
root@debian-servidor-fp:~# pamtest ssh upruebas2
Trying to authenticate <upruebas2> for service <ssh>.
Password:
Authentication successful.
```

4. Intentar autentificar al usuario `upruebas2` en el servicio `ssh` mediante una clave incorrecta:

```
root@debian-servidor-fp:~# pamtest ssh upruebas2
Trying to authenticate <upruebas2> for service <ssh>.
Password:
Failed to authenticate: Authentication failure
```

5. Intentar autentificar al usuario `upruebas2` en el servicio `ftp` mediante una clave correcta:

```
root@debian-servidor-fp:~# pamtest ftp upruebas2
Trying to authenticate <upruebas2> for service <ftp>.
Password:
Authentication successful.
```

6. Intentar autentificar al usuario `upruebas2` en el servicio `ftp` mediante una clave incorrecta:

```
root@debian-servidor-fp:~# pamtest ftp upruebas2
Trying to authenticate <upruebas2> for service <ftp>.
Password:
Failed to authenticate: Authentication failure
```

Una vez se ha llegado a este punto, el sistema ya está preparado para autentificar a los usuarios a través de LDAP.

ANEXO I - Servidores raíz DNS

```
;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       (e.g. reference this file in the "cache  .  <file>"
;       configuration file of BIND domain name servers).
;
;       This file is made available by InterNIC
;       under anonymous FTP as
;           file                /domain/named.cache
;           on server           FTP.INTERNIC.NET
;       -OR-                    RS.INTERNIC.NET
;
;       last update:    Jan 3, 2013
;       related version of root zone:   2013010300
;
; formerly NS.INTERNIC.NET
;
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
A.ROOT-SERVERS.NET.      3600000      AAAA  2001:503:BA3E::2:30
;
; FORMERLY NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     192.228.79.201
;
; FORMERLY C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
;
; FORMERLY TERP.UMD.EDU
;
.                        3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A     199.7.91.13
D.ROOT-SERVERS.NET.    3600000      AAAA  2001:500:2D::D
;
; FORMERLY NS.NASA.GOV
;
.                        3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
;
; FORMERLY NS.ISC.ORG
;
.                        3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000      A     192.5.5.241
F.ROOT-SERVERS.NET.      3600000      AAAA  2001:500:2F::F
;
; FORMERLY NS.NIC.DDN.MIL
;
.                        3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.      3600000      A     192.112.36.4
;
; FORMERLY AOS.ARL.ARMY.MIL
;
.                        3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000      A     128.63.2.53
H.ROOT-SERVERS.NET.      3600000      AAAA  2001:500:1::803F:235
;
; FORMERLY NIC.NORDU.NET
;
.                        3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.      3600000      A     192.36.148.17
I.ROOT-SERVERS.NET.      3600000      AAAA  2001:7FE::53
;
; OPERATED BY VERISIGN, INC.
;
.                        3600000      NS    J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.      3600000      A     192.58.128.30
J.ROOT-SERVERS.NET.      3600000      AAAA  2001:503:C27::2:30
;
; OPERATED BY RIPE NCC
;
.                        3600000      NS    K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.      3600000      A     193.0.14.129
```

```
K.ROOT-SERVERS.NET.      3600000      AAAA  2001:7FD::1
;
; OPERATED BY ICANN
;
.                        3600000      NS    L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.      3600000      A     199.7.83.42
L.ROOT-SERVERS.NET.      3600000      AAAA  2001:500:3::42
;
; OPERATED BY WIDE
;
.                        3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.      3600000      A     202.12.27.33
M.ROOT-SERVERS.NET.      3600000      AAAA  2001:DC3::35
; End of File
```

ANEXO II - Comprobar funcionamiento servidor DNS BIND

Procedimiento para comprobar el funcionamiento del servidor DNS BIND como servidor caché:
Prerequisitos: Haber realizado anteriormente lo expuesto en los puntos: **1.12. Instalación del servidor BIND** y **1.12.3. Configuración como caché DNS**.

1. Ejecutas el comando:
```
dig www.debian.org
```
El cual te muestra una salida similar a la siguiente:
```
; <<>> DiG 9.7.3 <<>> www.debian.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16236
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL: 0
;; QUESTION SECTION:
;www.debian.org. IN A
;; ANSWER SECTION:
www.debian.org. 300 IN A 86.59.118.148
www.debian.org. 300 IN A 82.195.75.97
;; AUTHORITY SECTION:
www.debian.org. 28800 IN NS geo1.debian.org.
www.debian.org. 28800 IN NS geo3.debian.org.
www.debian.org. 28800 IN NS geo2.debian.org.
;; Query time: 401 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Sep 12 08:20:34 2011
;; MSG SIZE rcvd: 121
```
donde, `401 msec` significa el tiempo de resolución consumido de la petición DNS en milisegundos.
2. Ejecutas de nuevo el comando anterior: `dig www.debian.org` obteniendo una salida similar a la siguiente:
```
; <<>> DiG 9.7.3 <<>> www.debian.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10876
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL: 0
;; QUESTION SECTION:
;www.debian.org. IN A
;; ANSWER SECTION:
www.debian.org. 295 IN A 82.195.75.97
www.debian.org. 295 IN A 86.59.118.148
;; AUTHORITY SECTION:
www.debian.org. 28795 IN NS geo3.debian.org.
www.debian.org. 28795 IN NS geo1.debian.org.www.debian.org. 28795 IN NS geo2.debian.org.
;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Sep 12 08:20:39 2011
;; MSG SIZE rcvd: 121
```
donde, `4 msec` significa el tiempo de resolución consumido de la petición DNS en milisegundos.
3. Ahora, deberías obtener una salida con un número inferior indicando un menor tiempo de resolución en la petición DNS. En este procedimiento puedes observar que el tiempo disminuye de `401 msec` en el paso 1 a `4 msec` en el paso 2, lo que indica que el tiempo de resolución consumido por la petición fue menor, puesto que la misma fue resuelta por la caché DNS, es decir, la primera consulta realizada en el paso 1 fue escalada a otro/s servidor/es DNS, empezando la búsqueda a través de los servidores raiz que se encuentran en el archivo `/etc/bind/db.root`, mientras que la segunda consulta, realizada en el paso 2 fue realizada en el propio servidor DNS y no fue escalada a otro/s servidor/es DNS.

ANEXO III - Ejemplo despliegue aplicación web OpenCart

"El movimiento se demuestra andando."

Diógenes de Sínope

Procede con el siguiente ejemplo: **Instalación de OpenCart**

1. **Descarga y descomprime la aplicación:**
   - ✓ En la página de descarga de OpenCart(http://www.opencart.com/index.php?route=download/download) puedes ver los requisitos para la instalación de Opencart: Web Server (preferiblemente Apache) , PHP (al menos la 5.2), MySQL , Curl , Fsock
   - ✓ Descarga el último paquete estable de Opencart de la página web de descarga en `/tmp/pruebas`

     ```
     mkdir /tmp/pruebas
     wget -c http://opencart.googlecode.com/files/opencart_v1.4.9.5.zip
     ```
   - ✓ Descomprime el paquete

     ```
     cd /tmp/pruebas
     apt-get install unzip
     unzip opencart_v1.4.9.5.zip
     ```
2. **Lee el fichero de instalación `install.txt`.**
3. **Crea el virtualhost para Opencart:**
   - ✓ Copia la carpeta `upload` en el servidor web. Para ello genera en `/etc/apache2/sites-available/` un virtualhost de nombre `tienda-virtual` como el siguiente:

     ```
     <VirtualHost 192.168.200.250:80>
        DocumentRoot /var/www/tienda-virtual
        ServerName www.tienda-virtual.empresa-proyecto.com
        ErrorLog /var/log/apache2/error tienda-virtual.log
        CustomLog /var/log/apache2/access tienda-virtual.log "%h %l %u %t \"%r\" %>s %b
     \"%{Referer}i\" %I %O"
     </VirtualHost>
     ```
   - ✓ Ahora mueve la carpeta `upload` con el nombre `tienda-virtual` en `/var/www/tienda-virtual`
   - ✓ Activa el sitio nuevo `tienda-virtual`: `a2ensite tienda-virtual`
   - ✓ Recarga la configuración de Apache: `/etc/init.d/apache2 reload`
   - ✓ Verifica que los siguientes ficheros y carpetas tengan permisos de escritura en `/var/www/tienda-virtual/`:
     `chmod 0755 ó 0777` para: `image/`, `image/cache/`, `image/data/`, `system/cache/`, `system/logs/`, `download/`, `config.php`, `admin/config.php`
4. **Crea la base de datos para OpenCart y el usuario con permisos en la misma**:
   Asegúrate que posees una base de datos mysql para Opencart y un usuario distinto de root con permisos en la misma:
   - ✓ Primero, debes crear una nueva base de datos para tu sitio Opencart:

     ```
     /usr/bin/mysql -h127.0.0.1 -uroot -p -e "CREATE DATABASE db_opencart;"
     ```

     donde:
     `root` es el usuario administrador de MySQL y por lo tanto tiene los privilegios para crear una base de datos.
     `db_opencart` es el nombre de la base de datos de opencart que acabas de crear.
     MySQL te pide la contraseña del usuario root y luego crea los archivos iniciales de la base de datos.
   - ✓ Segundo, creas el usuario con privilegios en la base de datos (*de nuevo se requiere la contraseña de root*).

     ```
     /usr/bin/mysql -h127.0.0.1 -uroot -p -e "GRANT
     SELECT,UPDATE,INSERT,DELETE,DROP,INDEX,ALTER,CREATE ON "db_opencart".* TO
     "db_user_opencart"@localhost IDENTIFIED BY 'opencart';"
     ```

     donde:
     '`db_opencart`' es el nombre de tu base de datos

'`db_user_opencart@localhost`' es el nombre de usuario de MySQL que posee los privilegios en la base de datos '`db_opencart`'.

'`opencart`' es la contraseña requerida para iniciar sesión como el usuario '`db_user_opencart`' en MySQL

- ✓ Tercero, para activar los nuevos cambios ejecuta:

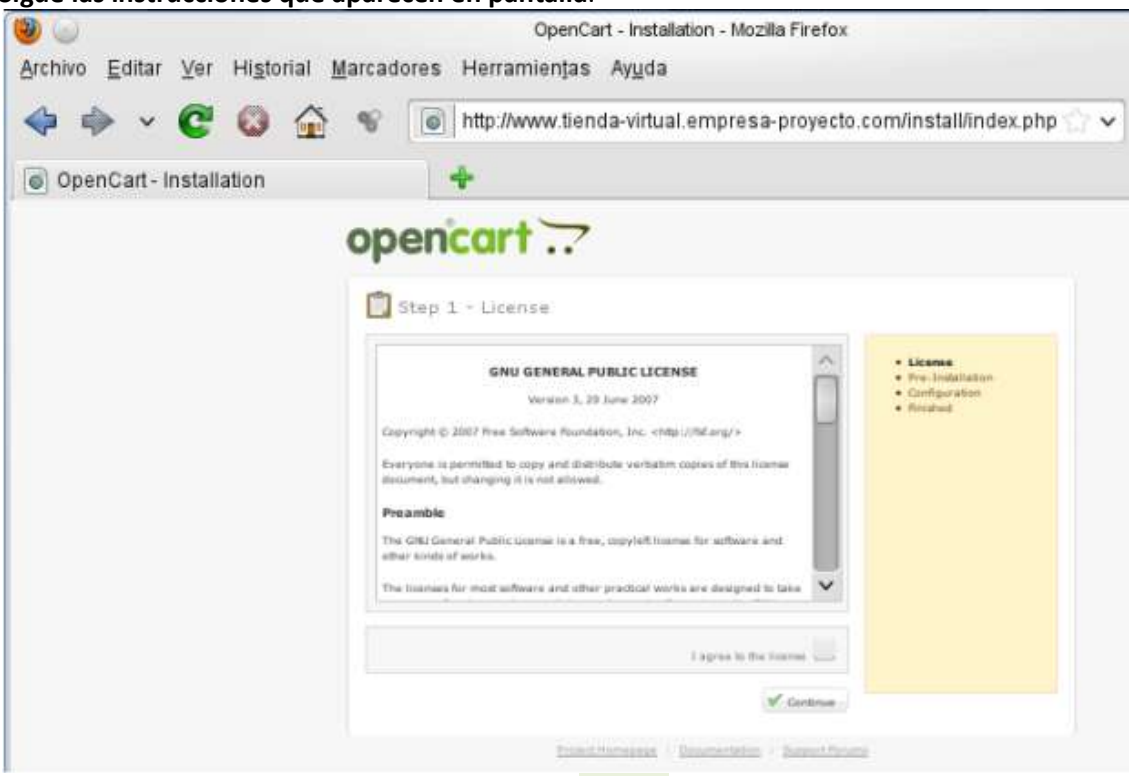
```
/usr/bin/mysql -h127.0.0.1 -uroot -p -e "flush privileges;"
```

Alternativamente puedes usar, si lo posees, tu panel de control Web o bien phpMyAdmin para crear la base de datos '`db_opencart`' y el usuario '`db_user_opencart`'

5. **Visita la página principal de tu Opencart**, por ejemplo:

```
http://www.tienda-virtual.empresa-proyecto.com/
```

6. **Sigue las instrucciones que aparecen en pantalla**.



7. **Una vez acabada la instalación borra la carpeta `install`.**
8. **Puedes ya visitar tu tienda online en**: `http://www.tienda-virtual.empresa-proyecto.com/` y tu panel de administración en: `http://www.tienda-virtual.empresa-proyecto.com/admin/`

ANEXO IV - Instalación y configuración de OpenLDAP

Para simplificar la administración de los usuarios del sistema es ideal utilizar una base de datos accesible mediante LDAP. Almacenar las cuentas de usuario de forma centralizada en un único repositorio facilitará la creación, modificación y eliminación de cuentas de usuario y grupos de usuarios. Será necesario configurar los PCs de la red para que utilicen el servidor LDAP como servidor de autentificación.

Instalación de OpenLDAP

El servidor OpenLDAP está disponible en el paquete `slapd` por tanto, lo instalaremos utilizando `apt-get`. También nos conviene instalar el paquete `ldap-utils` que contiene utilidades adicionales:

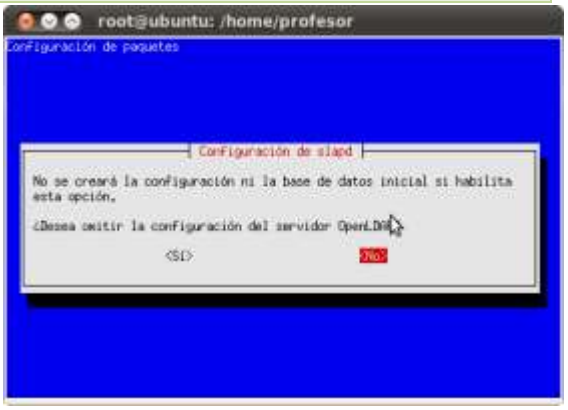
```
// Instalación del servidor LDAP
sudo apt-get install slapd ldap-utils
```

Configuración inicial de OpenLDAP

Los archivos de configuración del servidor LDAP se almacenan en la carpeta `/etc/ldap/`. En lugar de editar manualmente dichos archivos, es mejor lanzar el asistente de configuración de slapd. Para ello debemos ejecutar el siguiente comando:

```
//Lanzar el asistente de configuración de slapd
sudo dpkg-reconfigure slapd
```

Lo primero que nos pregunta el asistente es si deseamos omitir la configuración del servidor LDAP:



Obviamente responderemos que no, ya que precisamente lo que queremos es configurar el servidor LDAP.

Después nos preguntará si queremos que se elimine la base de datos cuando quitemos slapd. Para evitar confusiones con bases de datos anteriores, lo mejor es responder Sí:

Luego nos preguntará si deseamos utilizar LDAP versión 2, respondemos que no ya que apenas se utiliza.

Con esto habremos concluido la configuración inicial del servidor LDAP.

Arranque y parada manual del servidor LDAP

El servidor LDAP, al igual que todos los servicios en Debian, dispone de un script de arranque y parada en la carpeta `/etc/init.d`.

```
// Arrancar o reiniciar el servidor LDAP
sudo /etc/init.d/slapd restart

// Parar el servidor LDAP
sudo /etc/init.d/slapd stop
```

La configuración del servidor LDAP se guarda en `/etc/ldap` pero...
...es mejor no tocar manualmente los archivos de configuración

Administración de OpenLDAP

Introducción

Una vez instalado y configurado el servidor LDAP, la siguiente tarea es la del diseño de la estructura y la introducción de datos en el directorio.

Puesto que la finalidad de nuestro servidor LDAP es que sirva de almacén de usuarios y grupos para autentificar sistemas linux y servicios como ftp y web, deberemos crear una estructura que parta de la base de nuestro directorio, para almacenar dicha información. Tal y como se explica más abajo, crearemos una unidad organizativa (`ou`) llamada **`groups`**, para almacenar los grupos de usuarios y crearemos otra unidad organizativa llamada **`users`** para almacenar a los usuarios.

Paso 1: Cargar plantillas

Al instalar el servidor LDAP, se instalan también unas plantillas que nos serviran para crear el esquema básico para almacenamiento de usuarios unix para LDAP, lo que nos permitirá almacenar en nuestro directorio, cuentas de usuario. Para instalar las plantillas necesarias, debemos ejecutar los siguientes comandos:

```
// Instalar plantillas para almacenamiento de usuarios unix
sudo ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/ldap/schema/cosine.ldif
sudo ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/ldap/schema/nis.ldif
sudo ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/ldap/schema/inetorgperson.ldif
```

Paso 2: Archivo de configuración del esquema básico

Después crearemos un archivo en formato ldif con la configuración de nuestro esquema básico. En dicho archivo debemos configurar:

- ✓ **Base del directorio:** Se configura en el parámetro `olcSuffix` del archivo de configuración del esquema básico. En nuestro ejemplo usaremos: `dc=ieslapaloma,dc=com`
- ✓ **Nombre de usuario administrador**: Se configura en el parámetro `olcRootDN` del archivo de configuración del esquema básico. En nuestro ejemplo usaremos: `cn=admin,dc=ieslapaloma,dc=com`
- ✓ **Contraseña**: Se configura en el parámetro `olcRootPW` del archivo de configuración del esquema básico. En nuestro ejemplo usaremos: ldapadmin
- ✓ **Permiso de acceso a contraseñas:** Se configura en el parámetro `olcAccess: to attrs=userPassword`. Daremos al usuario administrador permiso de escritura y a cada usuario para cambiar su propia contraseña
- ✓ **Permiso de acceso global al directorio:** Se configura en el parámetro `olcAccess: to *`. Daremos al usuario administrador permiso de escritura y a todos los usuarios, permisos de lectura

Almacenaremos el archivo en la carpeta temporal porque una vez procesado se debería borrar, ya que contiene la contraseña de administrador en texto plano.

```
#  ---------- Archivo /tmp/ldapcurso-esquema-basico.ldif ----------
# Load dynamic backend modules
dn: cn=module,cn=config
objectClass: olcModuleList
cn: module
olcModulepath: /usr/lib/ldap
olcModuleload: back hdb

# Database settings
dn: olcDatabase=hdb,cn=config
objectClass: olcDatabaseConfig
objectClass: olcHdbConfig
olcDatabase: {1}hdb
olcSuffix: dc=ieslapaloma,dc=com
olcDbDirectory: /var/lib/ldap
olcRootDN: cn=admin,dc=ieslapaloma,dc=com
olcRootPW: ldapadmin
olcDbConfig: set_cachesize 0 2097152 0
olcDbConfig: set_lk_max_objects 1500
olcDbConfig: set_lk_max_locks 1500
olcDbConfig: set_lk_max_lockers 1500
olcDbIndex: objectClass eq
olcLastMod: TRUE
olcDbCheckpoint: 512 30
olcAccess: to attrs=userPassword by dn="cn=admin,dc=ieslapaloma,dc=com" write by anonymous
auth by self write by * none
olcAccess: to attrs=shadowLastChange by self write by * read
olcAccess: to dn.base="" by * read
olcAccess: to * by dn="cn=admin,dc=ieslapaloma,dc=com" write by * read
# ---------------------------------
```

Ahora habrá que cargar el servidor ldap con el archivo de configuración creado:

```
// Cargar en ldap el archivo ldapcurso-esquema-basico.ldif
sudo ldapadd -Y EXTERNAL -H ldapi:/// -f /tmp/ldapcurso-esquema-basico.ldif
```

Paso 3: Creación de unidades organizativas para almacenar cuentas unix

Para que nuestro directorio LDAP pueda almacenar cuentas unix, necesitamos crear una unidad organizativa (`dn: ou=users`) para los usuarios y otra (`dn: ou=groups`) para los grupos de usuarios. Antes debemos crear la base del directorio (`dn: dc=ieslapaloma,dc=com`) y el usuario administrador (`dn: cn=admin,dc=ieslapaloma,dc=com`). Después podemos crear usuarios y grupos para hacer pruebas. Crearemos los usuarios javier, joaquin y miguel en el grupo profesores y los usuarios jessica y joel en el grupo alumnos.

```
# ---------- Archivo /tmp/ldapcurso-usuarios.ldif ----------
dn: dc=ieslapaloma,dc=com
objectClass: top
objectClass: dcObject
objectClass: organization
dc: ieslapaloma
o: ieslapaloma

dn: cn=admin,dc=ieslapaloma,dc=com
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword:: e2NyeXB0fXdSVDNLMEpKSlQydmM=

dn: ou=users,dc=ieslapaloma,dc=com
objectClass: organizationalUnit
objectClass: top
ou: users

dn: ou=groups,dc=ieslapaloma,dc=com
objectClass: organizationalUnit
objectClass: top
ou: groups

dn: cn=Francisco Javier,ou=users,dc=ieslapaloma,dc=com
objectClass: inetOrgPerson
```

```
objectClass: organizationalPerson
objectClass: person
objectClass: posixAccount
objectClass: top
cn: Francisco Javier
gidNumber: 1001
homeDirectory: /home/javier
loginShell: /bin/bash
sn: Corcuera Ruiz
uid: javier
uidNumber: 1001

dn: cn=Joaquin,ou=users,dc=ieslapaloma,dc=com
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: posixAccount
objectClass: top
cn: Joaquin
gidNumber: 1001
homeDirectory: /home/joaquin
loginShell: /bin/bash
sn:: R8OzbWV6
uid: joaquin
uidNumber: 1002

dn: cn=Miguel Angel,ou=users,dc=ieslapaloma,dc=com
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: posixAccount
objectClass: top
cn: Miguel Angel
gidNumber: 1001
homeDirectory: /home/miguel
loginShell: /bin/bash
sn: Martinez
uid: miguel
uidNumber: 1003

dn: cn=Jessica,ou=users,dc=ieslapaloma,dc=com
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: posixAccount
objectClass: top
cn: Jessica
gidNumber: 1002
homeDirectory: /home/jessica
loginShell: /bin/bash
sn: Perez
uid: jessica
uidNumber: 1004

dn: cn=Joel Javier,ou=users,dc=ieslapaloma,dc=com
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: posixAccount
objectClass: top
cn: Joel Javier
gidNumber: 1002
homeDirectory: /home/joel
loginShell: /bin/bash
sn: Moreno
uid: joel
uidNumber: 1005

dn: cn=profesores,ou=groups,dc=ieslapaloma,dc=com
objectClass: posixGroup
objectClass: top
cn: profesores
gidNumber: 1001
memberUid: javier
memberUid: joaquin
memberUid: miguel
```

```
dn: cn=alumnos,ou=groups,dc=ieslapaloma,dc=com
objectClass: posixGroup
objectClass: top
cn: alumnos
gidNumber: 1002
memberUid: jessica
memberUid: joel
# ---------------------------------
```

Ahora habrá que cargar el servidor ldap con el archivo de usuarios creado:

```
// Cargar en ldap el archivo ldapcurso-usuarios.ldif (cuando pida la contraseña: ldapadmin)
sudo ldapadd -c -x -D cn=admin,dc=ieslapaloma,dc=com -W -f /tmp/ldapcurso-usuarios.ldif
```

A partir de este momento ya tendremos un servidor LDAP apto para almacenar usuarios y grupos de cuentas unix.

ANEXO V - Explorador de directorios LDAP

Aunque LDAP permite trabajar con comandos y archivos ldif, para acceder al directorio LDAP y poder crear y modificar elementos en dicho directorio, es más práctico utilizar un explorador de directorios LDAP (LDAP browser). Existen muchos exploradores LDAP tanto de pago como libres. Entre las aplicaciones libres destacamos **gq**, **phpldapadmin** (aplicación web) y **JXplorer**.

Para instalar `gq`, podemos utilizar `apt-get install gq`. Una vez instalada, para ejecutar `gq` tan solo debemos pulsar `alt+f2` y escribir `gq`.

Instalar phpldapadmin

Para instalar phpldapadmin, al igual que otras aplicaciones web, deberemos descargarla desde http://phpldapadmin.sourceforge.net/ y descomprimirla dentro del `DocumentRoot` de apache, es decir, dentro de la carpeta `/var/www`, por ejemplo en `/var/www/phpldapadmin`. Para ejecutarla, si la hemos descomprimido en la carpeta anterior, debemos ir a `http://ip del servidor web/phpldapadmin/` con el navegador y veremos la página principal de la aplicación:

JXplorer - Explorador LDAP en java

Por su calidad superior, utilizaremos **JXplorer** para administrar el directorio LDAP.
Previo a instalar **jxplorer**, es necesario instalar la máquina virtual java de Sun, para lo cual utilizaremos `apt-get`, pero antes debemos activar los repositorios -partner- de Ubuntu

```
// Instalación de Java (previamente activar repositorios partner)
sudo apt-get install sun-java6-bin sun-java6-jre sun-java6-plugin sun-java6-fonts
```

El comando anterior instalará java en la carpeta `/usr/lib/jvm/java-6-sun/jre/bin/`. Posteriormente tendremos que editar el archivo `/root/.bashrc` y añadir las variables que permitan al shell encontrar los binarios del JRE:

```
// Añadir en /root/.bashrc
CLASSPATH=/usr/lib/jvm/java-6-sun/jre/bin/
JAVA HOME=/usr/lib/jvm/java-6-sun/jre/bin/
PATH=/usr/lib/jvm/java-6-
sun/jre/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/X11:/usr/local/sbin:/usr/local/bin
```

Una vez instalado el java y establecidas las variables `CLASSPATH`, `JAVA HOME` y `PATH` en el archivo `/root/.bashrc`, debes cerrar el terminal y volver a abrirlo, para que cargue nuevamente las variables de entorno. Si ejecutas el comando `set` en el terminal, podrás comprobar que ha cargado las variables de entorno y podrás instalar JXplorer. JXplorer no está disponible en los repositorios de paquetes de debian, pero se puede descargar desde:

http://enebro.pntic.mec.es/arug0000/servicio/jxplorer3.2_linux.bin

Debemos copiar el archivo en la carpeta `/tmp` de nuestro sistema y ejecutar:

```
// Instalar JXplorer (como usuario, no como root)
sh /tmp/jxplorer3.2_linux.bin
```

Se iniciará un sencillo asistente de instalación que al finalizar habrá creado la carpeta JXplorer en nuestra carpeta `home` y el script de inicio `jxplorer.sh` dentro de ella, por lo tanto para ejecutarlo debemos escribir:

```
// Ejecutar JXplorer: Entran en la carpeta de instalación y ejecutar:
~/JXplorer/jxplorer.sh
```

Veremos la pantalla principal de JXplorer:

Conexión con el servidor LDAP

La conexión con el servidor LDAP podemos hacerla como **usuario anónimo** o como **usuario administrador**. Si conectamos de forma anónima solo podremos visualizar los elementos pero no podremos hacer cambios. Si conectamos como administrador, podremos crear, modificar y eliminar elementos de cualquier tipo.

Para conectar al servidor LDAP como administrador necesitamos la siguiente información:

- ✓ Dirección **IP** del servidor LDAP
- ✓ **Protocolo** del servidor (`LDAP v3` en nuestro caso)
- ✓ **Base del directorio** (`dc=ieslapaloma,dc=com` en nuestro caso)
- ✓ Nombre de usuario **administrador** (`cn=admin,dc=ieslapaloma,dc=com` en nuestro caso)
- ✓ **Contraseña** (`ldapadmin` en nuestro caso)

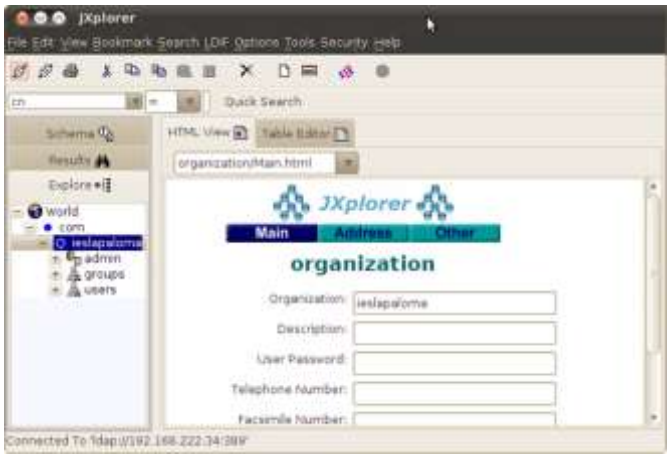
La base del directorio se suele denominar en inglés '**base DN**' o '*Nombre Distinguido de la base del directorio*'. Se corresponde con el parámetro '`suffix`' del archivo de configuración del servidor LDAP `/etc/ldap/slapd.conf`.

El nombre del usuario con el que nos conectamos se suele denominar en inglés '`user DN`' o también '`bind DN`'.

El nombre de usuario administrador por defecto suele ser `admin` y a menudo hay que proporcionar nombre y base del directorio: `cn=admin, dc=ieslapaloma, dc=com`

Al hacer clic en el botón '**conectar**' (*marcado con círculo rojo en la anterior figura*) nos aparecerá el diálogo de conexión para que introduzcamos los datos de la conexión. Para no tener que introducir dicha información cada vez que conectemos, podemos grabar los datos pulsando '**Save**'.

Si pulsamos **OK**, JXplorer conectará con el servidor LDAP y mostrará el directorio.



Vemos que en nuestro directorio ya tiene creada la organización llamada 'ieslapaloma', el usuario administrador llamado 'admin' y dos unidades organizativas: groups y users en las cuales se encuentran los grupos y los usuarios anteriormente creados.

Creación de usuarios y grupos con jxplorer

Anteriormente hemos creado el grupo alumnos y el grupo profesores mediante el archivo ldapcurso-usuarios.ldif. Ahora veremos cómo crear usuarios y grupos desde la herramienta jxplorer. Como ejemplo, crearemos un nuevo grupo y un nuevo usuario.

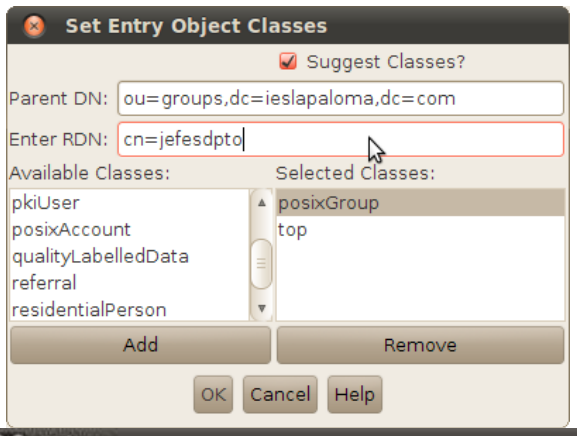
Crearemos el siguiente grupo:

- ✓ jefesdpto (gid=1003)

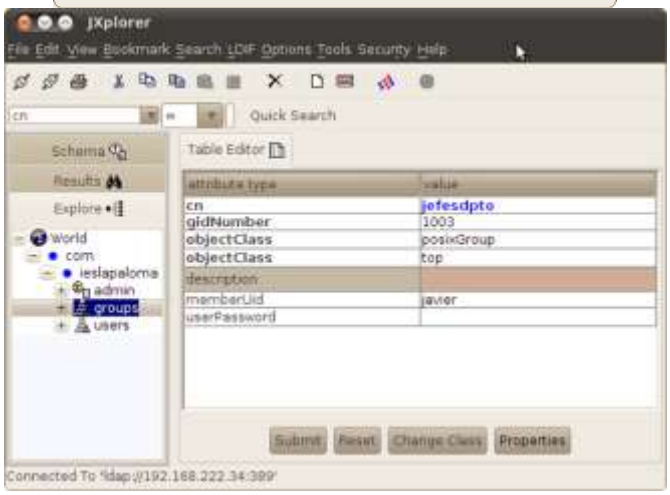
Además, crearemos un usuario nuevo:

- ✓ carlos (uid=1006, jefesdpto)

Para crear los grupos, haremos clic con el botón derecho en la unidad organizativa '`groups`' y haremos clic en '`New`'. Observamos en '`Selected Classes`' (clases seleccionadas) que está seleccionada la clase '`posixGroup`'. El nombre (RDN) será `jefesdpto`, por tanto debemos escribir '`cn=jefesdpto`' (**`cn= Common Name - Nombre Común`**):



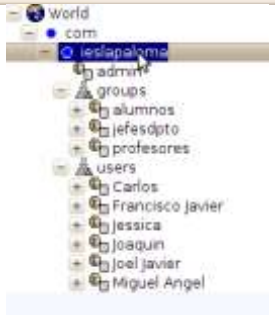
Al pulsar **OK** nos apacererá la siguente figura, en la cual observamos los atributos clásicos de un grupo posix. Debemos rellenar al menos el campo `gidNumber`. También podemos introducir miembros al grupo. En el parámetro `memberUid` añadimos `javier`. Luego, haciendo clic con el derecho en `javier > Add another value` podemos añadir más miembros.



Para crear los usuarios, haremos clic con el derecho en la unidad organizativa '`users`' y haremos clic en '`New`'. Observamos en '`Selected Classes`' (clases seleccionadas) que están seleccionadas las clases '`inetOrgPerson`', '`organizationalPerson`', '`person`' y '`posixAccount`'. Si su nombre es `Carlos`, podemos escribir en la casilla `RDN` '`cn=Carlos`'.

Al pulsar **OK** nos apacerá la siguiente figura, en la cual observamos los atributos de las tres tipologías de nuestro elemento: `persona`, `usuario de internet` y `cuenta posix`. Debemos rellenar al menos los campos `gidNumber` (grupo primario que será el 1003), `homeDirectory`, `uid` (identificador), `uidNumber` y `sn` (surname - apellidos). También podemos configurar la contraseña en el atributo `userPassword` escribiendo la nueva contraseña cifrada con MD5.

Lo mismo haremos con el resto hasta que tengamos creados los cinco usuarios. Al final nuestro servidor LDAP tendrá la siguiente información:



Ya tendríamos creada la estructura, los grupos y los usuarios que necesitamos para nuestro sistema.

Trabajar con herramientas gráficas como jxplorer o phpldapadmin resulta interesante cuando hay que realizar consultas o pequeñas modificaciones...
...pero cuando se trata de crear usuarios de forma masiva, lo mejor es utilizar archivos ldif y el comando ldapadd para cargarlos al servidor

ANEXO VI - Administración de usuarios y grupos con LDAP

Administración mediante scripts

El paquete `ldapscripts` incluye una serie de scripts para gestionar de forma sencilla, usuarios y grupos almacenados en el servidor LDAP. En primer lugar tenemos que instalar el paquete:

```
sudo apt-get install ldapscripts
```

A continuación tenemos que editar el fichero de configuración **`/etc/ldapscripts/ldapscripts.conf`** de acuerdo a las preferencias de nuestro servidor LDAP, descomentando y modificando los siguientes parámetros:

```
SERVER="ldap://localhost"
BINDDN="cn=admin,dc=iescalquera,dc=local"
BINDPWDFILE="/etc/ldapscripts/ldapscripts.passwd"
SUFFIX="dc=iescalquera,dc=local"
GSUFFIX="ou=grupos"
USUFFIX="ou=usuarios"
MSUFFIX="ou=maquinas"
CREATEHOMES="yes"
```

Para terminar la configuración del paquete, introduciremos en nuestro fichero **`/etc/ldapscripts/ldapscripts.passwd`** una contraseña para conectarse al servidor LDAP:

```
sudo sh -c "echo -n 'admin' > /etc/ldapscripts/ldapscripts.passwd"
sudo chmod 400 /etc/ldapscripts/ldapscripts.passwd
```

A continuación se muestra el uso de scripts del paquete para crear/cambiar contraseña y borrar un usuario, así como crear/borrar un grupo y añadir/eliminar usuarios a un grupo:

```
sudo ldapaddgroup alumnos
Successfully added group alumnos to LDAP
sudo ldapadduser pepe alumnos
Successfully added user pepe to LDAP
Successfully created home directory for user pepe
sudo ldapsetpasswd pepe
Changing password for user uid=pepe,ou=usuarios,dc=iescalquera,dc=local
New Password:
Retype New Password:
Successfully set password for user uid=pepe,ou=usuarios,dc=iescalquera,dc=local
sudo ldapaddusertogroup pepe profes
Successfully added user pepe to group profes
```

NOTAS:

- ✓ En **`/home`** del servidor creamos una carpeta personal para `pepe`, pero no para nuestros clientes, ya que eso se verá en la parte III del curso.
- ✓ Para comprobar el resultado, ahora podemos iniciar sesión, en modo consola, no en modo gráfico, que se verá en la parte III del curso, como usuario `pepe` desde un equipo configurado para manejar los usuarios de LDAP y utilizar el comando `id` para ver los grupos a los que pertenece:

```
$ id
uid=10001(pepe) gid=10001(alumnos) grupos=10000(profes),10001(alumnos)
```

Vamos ahora a ver cómo borrar el usuario y grupo creados:

```
sudo ldapdeleteuserfromgroup pepe profes
Successfully deleted user pepe from group profes
sudo ldapdeleteuser pepe
Successfully deleted user uid=pepe,ou=usuarios,dc=iescalquera,dc=local from LDAP
sudo ldapdeletegroup alumnos
Successfully deleted group cn=alumnos,ou=grupos,dc=iescalquera,dc=local from LDAP
```

***NOTA*: Observar como se elimina el usuario `pepe`, pero no así su carpeta personal del servidor en **`/home`**. El cliente no tenía ninguna carpeta.**

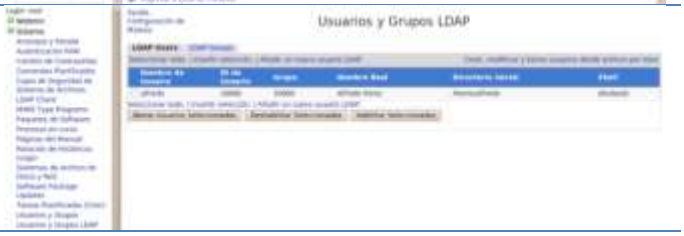
Una opción que puede ser muy útil con estos scripts es la de definir un modelo para los valores por defecto que tendrán los nuevos usuarios, grupos y máquinas. Estos modelos deben ser almacenados en ficheros con formato LDIF (en `/usr/share/doc/ldapscripts/examples` hay ejemplos de estos ficheros con la extensión `.template.sample`). En el fichero de configuración `/etc/ldapscripts/ldapscripts.conf` podemos indicar los ficheiros de modelos en los que queramos utilizar nuestros parámetros **`UTEMPLATE`** (usuarios), **`GTEMPLATE`** (grupos) e **`MTEMPLATE`** (máquinas).

Administración con webmin

El webmin incluye un módulo muy cómodo para hacer a gestión de usuarios y grupos de LDAP. Accederemos a la categoría de **Sistema**, como nombre de **Usuarios y Grupos LDAP**. Si no aparece aquí, tendremos que picar en la opción de **Refresh Modules** para que detecte ahora que un servidor LDAP está instalado y que este módulo ya tiene utilidades.

Configuración inicial del módulo de Usuarios y grupos LDAP

Si accedemos al módulo, veremos que hay un software para el uso del protocolo LDAP con scripts en PERL (que es un lenguaje de programación en el que está escrito el webmin) que no está instalado. Picando en el enlace **`Pulse aquí`** el webmin instalará, usando el comando `apt-get` los paquetes necesarios:

Página que muestra a webmin informando de la necesidad de instalar paquetes para el funcionamento del módulo	
Resultado de la correcta instalación de dos paquetes necesarios	
Una vez instalados los paquetes, ya podemos acceder al módulo y visualizar los usuarios y grupos de LDAP	

En este momento un módulo de gestión de usuarios y grupos LDAP de webmin ya es totalmente operativo y podemos agregar, editar y borrar usuarios y grupos en enuestro servidor LDAP, pero hemos de realizar un par de cambios en la configuración del módulo para afinar su funcionamiento. Veamos cuáles son los problemas…

Es muy común que las distribuciones de Linux comiencen a asignar identificadores de usuario a los nuevos usuarios locales con el número 500 ó 1000 (*este es el caso de Ubuntu*). Por lo tanto, es conveniente que los usuarios de LDAP no coincidan con el ID de usuario con estos usuarios, porque entonces al iniciar sesión en el equipo cliente se le asignarán los derechos y privilegios del usuario local al usuario del dominio (tenga en cuenta que la gestión de permisos que se hace en Linux es basado en el UID del usuario), y lo mismo podría decirse de los grupos. Así que lo que hacemos es establecer en el módulo Webmin para los nuevos usuarios y los grupos que se crean en LDAP es

asignar identificadores a partir del número 10000, y no hay identificadores coincidentes entre los usuarios locales del equipo y el dominio (si nos fijamos en el archivo de configuración `ldapscripts`, este es el identificador mínimo de los usuarios y grupos que se configuran de forma predeterminada).

Por otro lado, el módulo toma la rama de base de LDAP de usuarios y la rama base para grupos de fichero de configuración del cliente LDAP, que será en nuestro caso `dc = iescalquera, dc = local`, cuando queramos almacenar usuarios y grupos en subramas que no sea LDAP (`o = usuarios, dc = iescalquera, dc = local y ou = grupos, dc = iescalquera, dc = local`). Hay decir que esto no es necesario y podría funcionar perfectamente almacenando los usuarios y grupos directamente en la rama raíz de LDAP, pero para tener un poco más ordenado el directorio estructurarémoslo de esta manera.

Así que haga clic en el enlace **Configuración de módulos** que se encuentran en la parte superior de la página y accederemos a una página en la que se puede establecer una serie de parámetros acerca del comportamiento del módulo. En particular, vamos a modificar el texto siguiente:

- ✓ En la sección **Opciones del servidor LDAP**, en **base de usuarios** y en **Base para grupos**:

- ✓ Dentro del apartado de **Opciones para usuario nuevo** en **UID menor para nuevos usuarios** y en **GID menor para nuevos grupos**:

- ✓ Hacemos clic en el botón **Salvar** para guardar esta configuración

Administración de usuarios y grupos de LDAP con webmin

La administración de usuarios y grupos de LDAP con este módulo es muy simple, y sólo tendremos que usar los enlaces para la creación de nuevos usuarios y grupos, y picar sobre un nombre de usuario o un grupo para editar asus propiedades o eliminarlo. A continuación se muestran un par de ejemplos de la creación de un usuario y de un grupo:

Creación del usuario `felipe`, con contraseña `abc123.` e incluido en el grupo `profes`

Creación del grupo `profes-informatica`, e inclusión del usuario `felipe` en este grupo

Creación masiva de usuarios

El módulo de usuarios y grupos LDAP de webmin ofrece la opción de **Crear, modificar** y **borrar usuarios desde un archivo por lotes**. Con ella podemos subir al servidor un fichero de texto de datos de una serie de usuarios (una linea por cada usuario) y automatizar la creación de modificación masiva en LDAP. Esto es enormemente útil cuando el número de usuarios que hay que manejar es grande, y puede ahorrar mucho tiempo de administración.

Por ejemplo, un fichero para la creación de dos usuarios podría tener el siguiente contenido (ojo, las líneas deben comenzar por `create`, `modify` o `delete`, y no por `crear`, `modificar` y `borrar` como aparece en las instruciones traducidas al castellano):

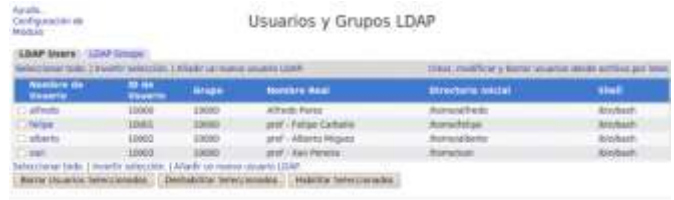
```
create:alberto:abc123.::10000:prof - Alberto Miguez:/home/alberto:/bin/bash:::::
create:xan:abc123.::10000:prof - Xan Pereira:/home/xan:/bin/bash:::::
```

Las instruciones de la página explican qué campos son necesarios y cuales se pueden dejar en blanco, como se hace con algunos campos en este ejemplo. Por su puesto, en cada caso concreto y dependiendo del formato del fichero que se nos proporcione para la creación de usuarios, habrá que buscar el método más o menos automatizado de crear un fichero con este formato, o bien escribiendo algún script o simplemente con algún programa de hoja de cálculo guardando el ficheiro resultante en formato *CSV* (fichero de texto separado por comas) estableciendo como separador de campo el carácter `:` en lugar da `,`.
Podemos ver a continuación un ejemplo donde se carga el fichero `usuarios.txt` con este contenido, y el resultado de su ejecución:

Página para la carga de un fichero para la creación masiva de usuarios.

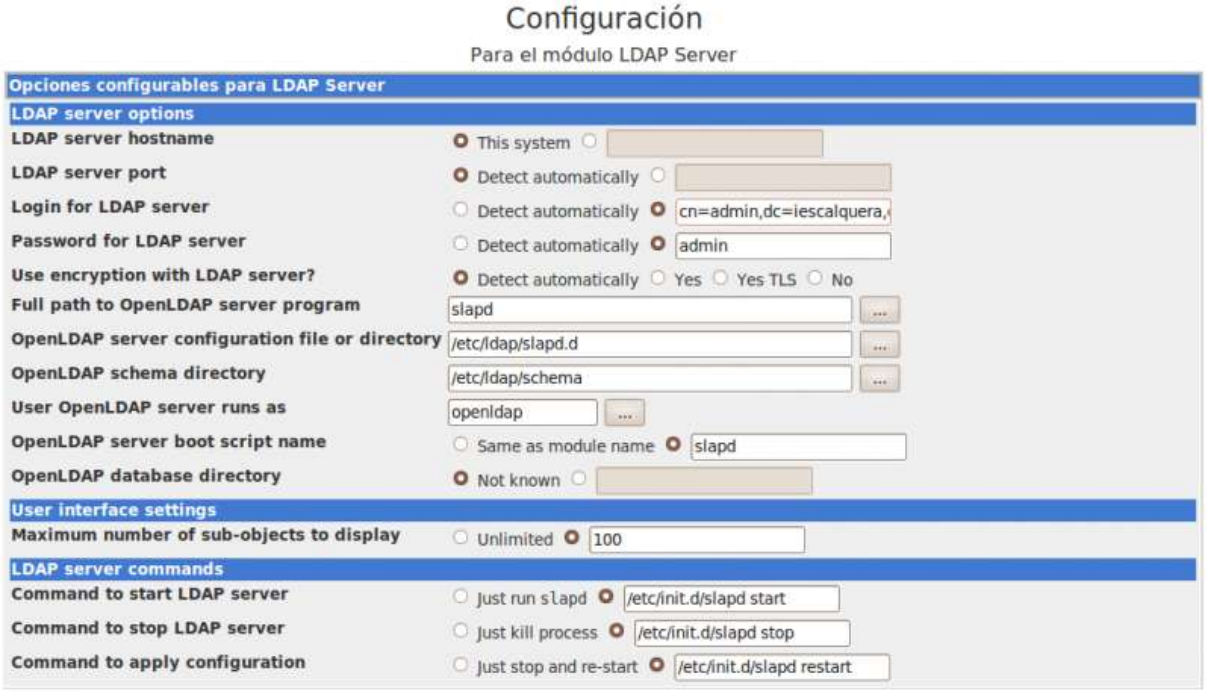
Resultado del proceso de creación de usuarios. Observar como en `/home` están las carpetas personales de usuarios creados.

Lista de usuarios de LDAP después de cargado el fichero

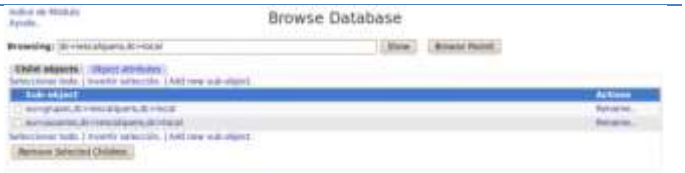


El módulo de servidor LDAP

Webmin también incluye un módulo **LDAP Server** (dentro de la categoría de **Servidores**), aunque no se utiliza para configurar el servidor LDAP en nuestro caso, puede ser útil para poder navegar por los datos almacenados en él. Antes de usarlo, tenemos que entrar en la configuración del módulo para introducir el usuario y la contraseña que usará para conectarse al servidor LDAP, que podrá ser un usuario normal si sólo queremos visualizar los datos almacenados o el administrador si queremos también poder realizar modificacións de los datos de calquier usuario del grupo:



Una vez guardados estos datos, clicamos en la opción **Browse Database**, introducimos una rama de LDAP que queremos explorar y picamos en el botón de **Show**. A continuación podemos ver algunas páginas de exploración de LDAP:

Vista del contenido de la rama base de LDAP	
Vista de las propiedades del usuario Alberto	

LDAP Account Manager

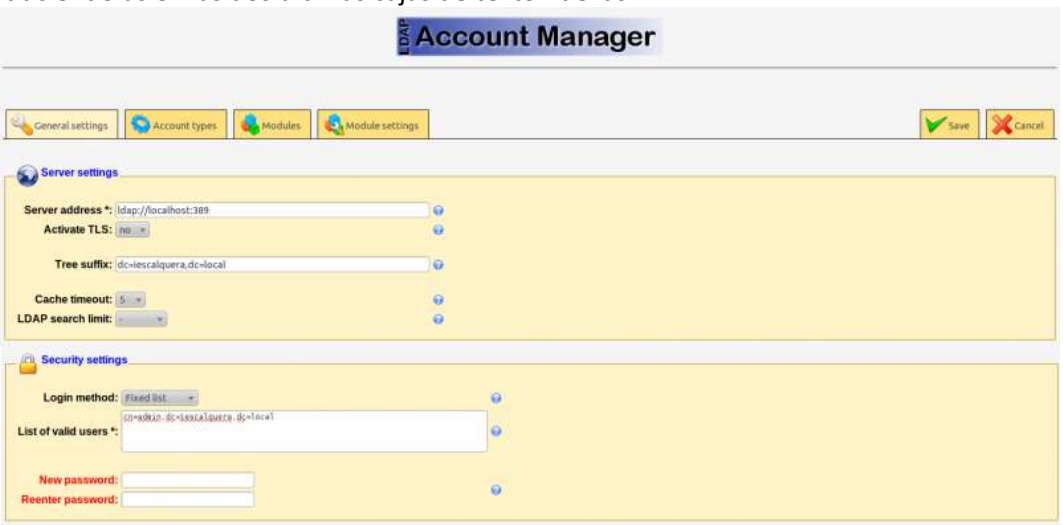
Otra herramienta que podemos utilizar para administrar los usuarios y grupos del servidor LDAP y LDAP Account Manager. En Ubuntu Server, instalamos el paquete `ldap-account-manager`, así que introduciremos el comando:

```
sudo apt-get install ldap-account-manager
```

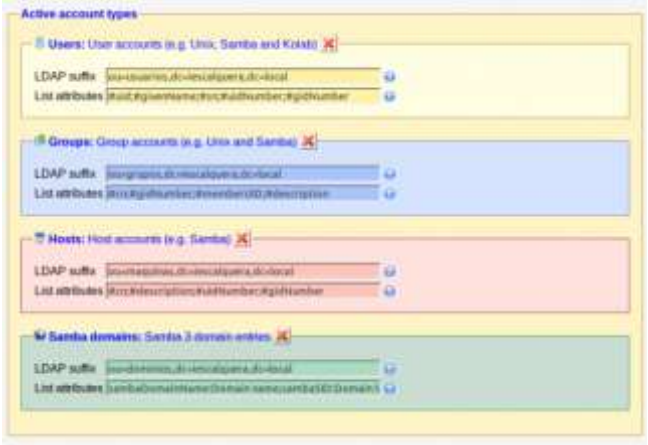
Con esto ya nos podemos conectar con un navegador desde un cliente introduciendo la dirección `http://direcciónIPServidor/lam` (si es un servidor real, sería muy recomendable configurar el servidor apache para recibir conexións seguras y usar **https** en lugar de **http**):

Picamos en el enlace de **`LAM configuration`** en el logo en **`Edit server profiles`** para configurar los parámetros de conexión a nuestro servidor LDAP. Introduciremos la contraseña por defecto (*lam*) e entramos en la página de configuración en la que modificaremos los parámetros:

- ✓ En la pestaña `General Settings`:
  - ➔ **`Tree suffix`**: Para introducir el sufijo de nuestro directorio (**`dc=iescalquera,dc=local`**).
  - ➔ **`Default language`**: Español.
  - ➔ **`List of valid users`**: Pondremos un DN de usuario administrador del LDAP (**`cn=admin,dc=iescalquera,dc=local`**)
  - ➔ Podremos cambiar la contraseña para acceder a esta página de configuración introduciendolas en las dos últimas cajas de texto nuevas.



- ✓ En la pestaña **`Account Types`**, dentro del apartado **`Active account types`**:
  - ➔ **Users -> LDAP suffix**: `ou=usuarios,dc=iescalquera,dc=local`
  - ➔ **Groups -> LDAP suffix**: `ou=grupos,dc=iescalquera,dc=local`
  - ➔ **Hosts -> LDAP suffix**: `ou=maquinas,dc=iescalquera,dc=local`
  - ➔ **Samba domains -> LDAP suffix**: `ou=dominios,dc=iescalquera,dc=local`



Picamos en el botón **Save** para guardar los cambios. Todos estos parámetros introducidos se almacenan en el fichero de configuración de lam (`/usr/share/ldap-account-manager/config/lam.conf`).
Ahora ya podemos entrar una herramienta introduciendo la contraseña de administrador de LDAP (`admin`):

Inicio de sesión	
Nos pregunta si queremos crear las ramas para almacenar las máquinas y los dominios en el directorio, ya que detecta que no existe aún.	
Vista del árbol de LDAP	
Vista de los usuarios	
Vista de los grupos	

IMPORTANTE: LAM se puede utilizar para crear usuarios y grupos, pero no va a crear carpetas personales en el servidor asociado con cada usuario.