UD1: Servicios necesarios para el despliegue web

1. DNS

1.1 Sistema de nombres de dominio.

¿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? ¿existen 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.

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 en 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**.

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 muchos TLDs 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:

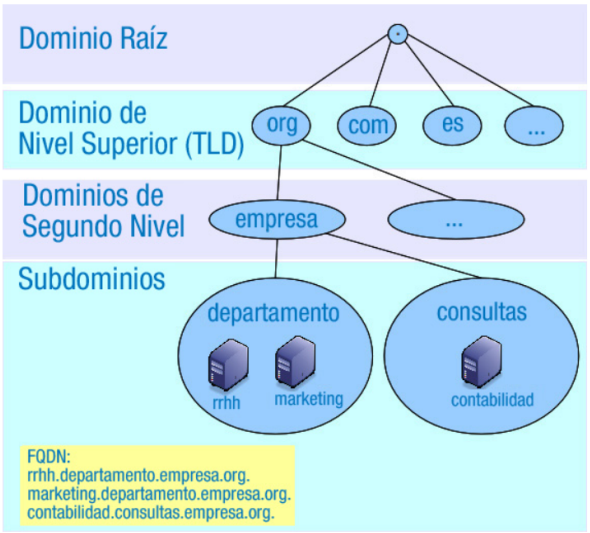
- La empresa agrupa sus nodos en el dominio de primer nivel "`org`". Éste es un TLD.

- La empresa tiene un subdominio, dominio de segundo nivel "`empresa`" bajo "`org`". Así "`empresa`" es un dominio de segundo nivel, hijo del TLD "`org`".
- 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 , 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.
```

❓ ¿Tiene sentido que haya un cuarto o un quinto subnivel de dominio?

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:



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 (): dominios asociados con países y territorios. Hay más de 240 ccTLD. Están formados por 2 letras, por ejemplo: `es`, `uk`, `en`, y `jp`.

  ccTLD
- Dominios de nivel superior genéricos (): están formados por 3 o más letras. A su vez se subdividen en:

  gTLD

- Dominios de internet patrocinados (): 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`.

  sTLD

- Dominios de internet no patrocinados (). Sin una organización detrás que establezca las reglas para optar a dicho dominio. La lista de gTLD incluye: `com`, `net`, `org`, `biz`, `info`.

  uTLD

1.2 Ventajas del DNS

¿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 ya que puede ocurrir que desconozcamos la IP que tiene cierto equipo pero sí conocer su nombre en el dominio, que será invariable.

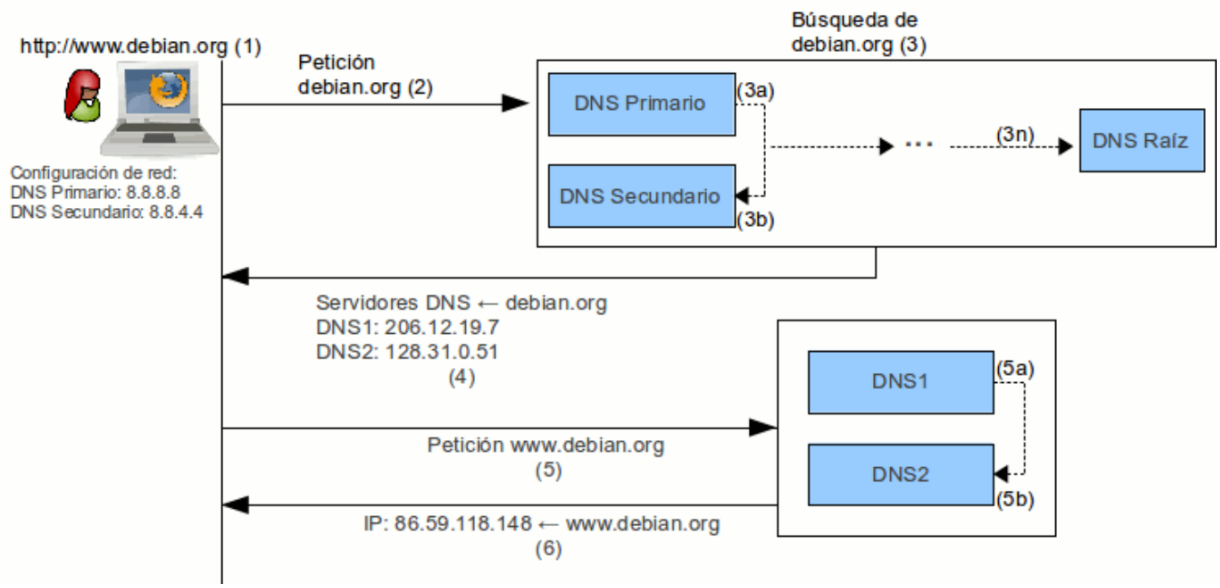
Un DHCP (Dynamic Host Configuration Protocol) es un protocolo de red que permite a los dispositivos obtener automáticamente una configuración de red, incluyendo una dirección IP, máscara de subred, puerta de enlace predeterminada, y servidores DNS. El DHCP simplifica la administración de la red al asignar de manera dinámica direcciones IP a los dispositivos en lugar de configurarlas manualmente en cada dispositivo de la red. Esto facilita la conectividad y la administración de la red, especialmente en entornos con un gran número de dispositivos.

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 siguiente imagen animada 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 la URL 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 (): 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).

   ISP
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 servidor DNS 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 **DynDNS**, **No-IP** y **FreeDNS**. 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?

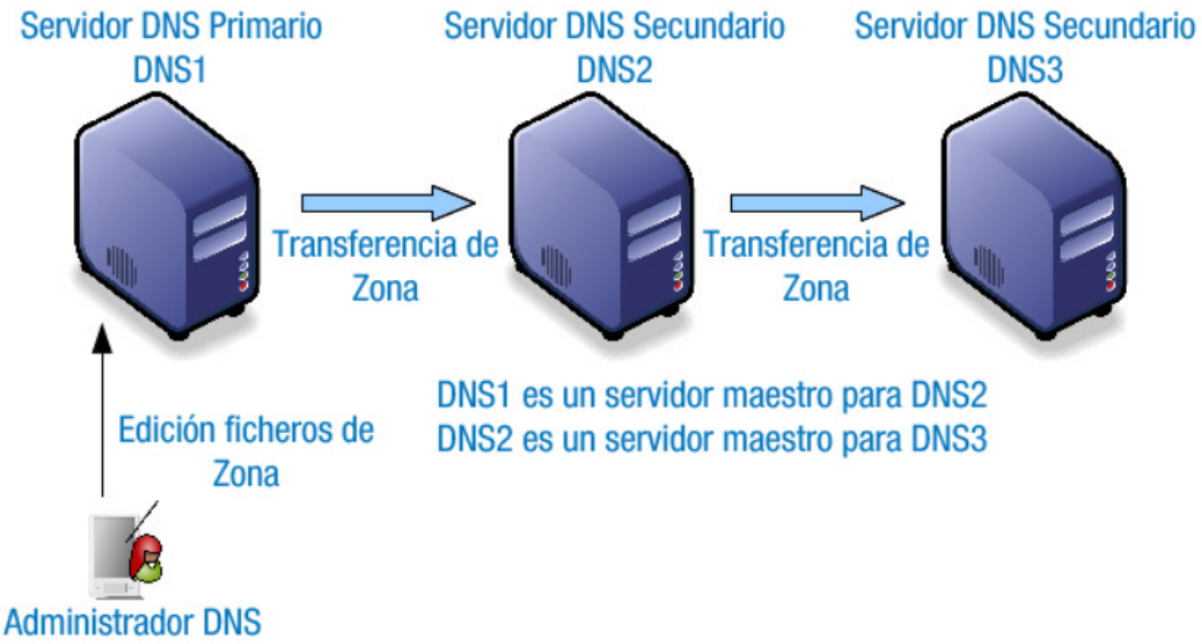
Si no tienes configurado un servidor DNS con las entradas de dominio necesarias, puedes generar estas entradas modificando el archivo `/etc/hosts`, añadiéndolas al final del mismo:

- #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 de la misma:

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 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.

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.

1.7 Tipos de registro 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:

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 un TTL 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.

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**: Canonical 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**: Name Server (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**: Mail eXchange (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**: Pointer (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**: Start Of Authority (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**: Sender Policy Framework. 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"`.

1.8 Funcionamiento del cliente DNS

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**.

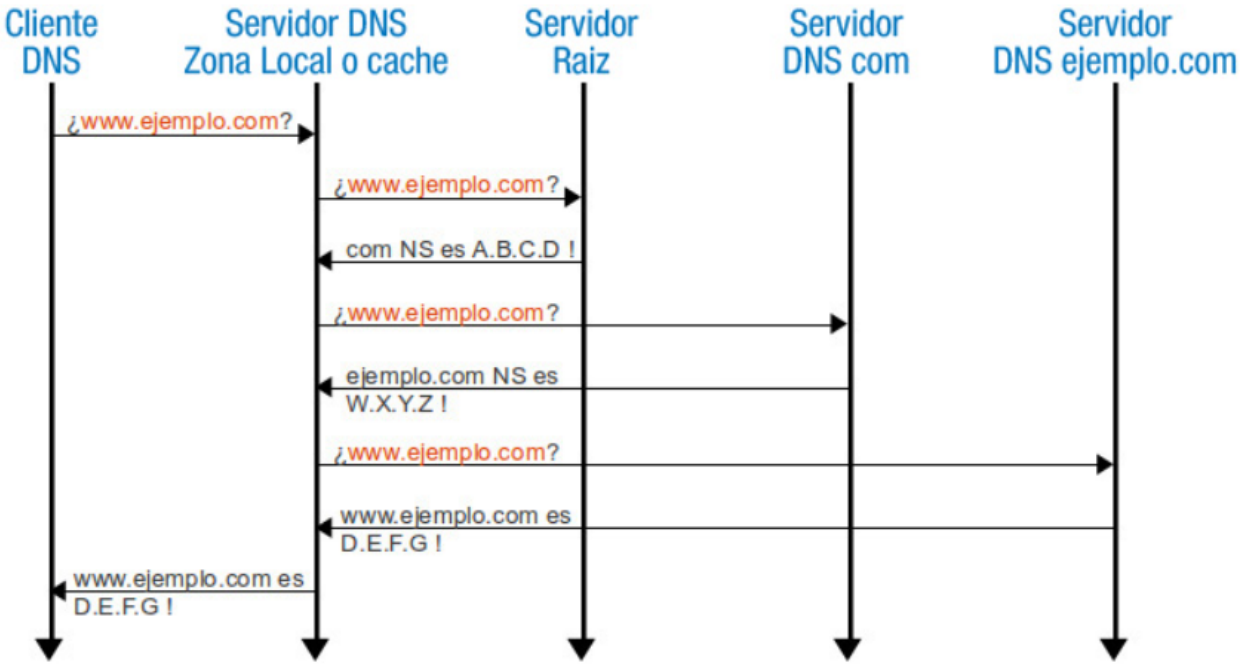
En general, el proceso de consulta DNS se realiza en dos partes:

- 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

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.



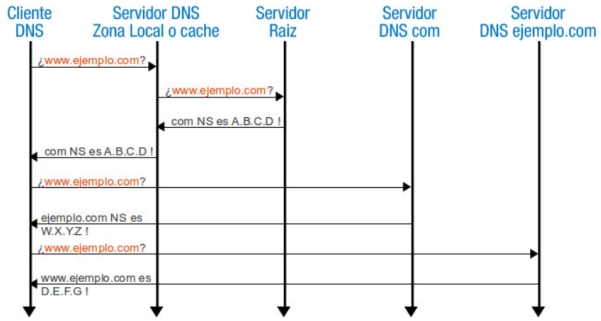
1. Para que el servidor DNS realice la recursividad correctamente, primero necesita información de contacto útil acerca de los otros servidores DNS del espacio de nombres de dominio DNS. Esta información se proporciona en forma de *sugerencias de raíz*, una lista de los registros de recursos preliminares 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

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 DNS pueda 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

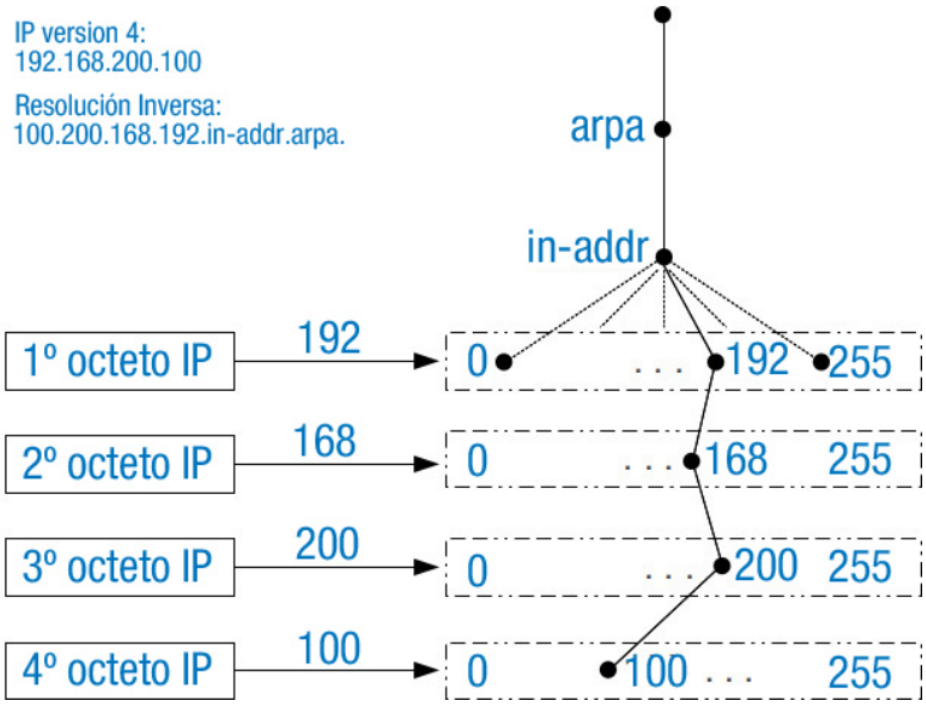
En la mayoría de la consultas DNS los clientes normalmente realizan una búsqueda directa. Este tipo de consulta espera recibir una dirección IP como respuesta a la consulta. Pero, DNS también proporciona un proceso de búsqueda inversa, es decir, buscar un nombre de host a través de una dirección IP. 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 IP 192.168.200.100 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`.

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 DNS actualiza 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`.

2. Comandos para verificar el DNS

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 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`.

En Debian / Ubuntu deberemos instalar estas utilidades mediante:

```
apt install dnsutils
```

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`

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 1`30.206.1.5` corresponde con el nombre de dominio `zeppo.rediris.es`, e incluso el registro de recursos empleado: `PTR`.

3. Instalación y configuración de bind9

3.1 Instalación

Para una instalación del servidor DNS BIND en Debian / Ubuntu, realiza el siguiente procedimiento como usuario **root** (o **sudo** en Ubuntu), 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.

   ```
   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.

   ```
   apt-get upgrade
   ```

3. Instala los paquetes necesarios para el funcionamiento de BIND (bind9).

   ```
   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.

```
/etc/init.d/bind9 status

systemctl status bind9

service bind9 status
```

5. Verifica en qué puertos y está activo el servidor **bind9**, para ello comprueba el servicio **named**:

```
root@debian-servidor-fp:~# netstat -natp | grep named
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN      490/named
tcp        0      0 192.168.200.250:53      0.0.0.0:*               LISTEN      490/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      490/named
tcp6       0      0 ::1:953                 :::*                    LISTEN      490/named
tcp6       0      0 :::53                   :::*                    LISTEN      490/named
```

3.2 Configuración

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

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`.

3.2.1 Comandos para verificar que funciona correctamente

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 de `/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
   ```

3.2.2 Parar y arrancar el servidor

En un sistema operativo Debian / Ubuntu, puedes comprobar el estado del servicio bind mediante el comando service o mediante el comando `/etc/init.d/bind`:

- **Comando service:** NOTA: En sistemas actuales Debian, como por ejemplo la versión 10, `service` no está habilitado, para ello deberemos añadir al archivo `~/.bashrc`, la línea:

  ```
  PATH=$PATH:/usr/sbin
  ```

  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.

- **Comando systemctl**

1. Arrancar el servidor DNS:

```
root@debian-servidor-fp:~# systemctl start bind9
```

2. Parar el servidor DNS:

```
root@debian-servidor-fp:~# systemctl stop bind9
```

3. Comprobar el estado activo/inactivo del servicio:

`root@debian-servidor-fp:~# systemctl status bind9`

3.3 Bind9 como caché DNS

> 💡 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 y Ubuntu 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;
  // };

  //========================================================================
  // If BIND logs error messages about the root key being expired,
  // you will need to update your keys.  See https://www.isc.org/bind-keys
  //========================================================================
  dnssec-validation auto;

  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: `systemctl reload bind9` ó `service bind9 reload` ó `/etc/init.d/bind9 reload`.

3.4 Bind9 como servidor principal

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: `systemctl reload bind9` ó `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 9.11.5-P4-5.1+deb10u1-Debian <<>> ejemplo.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15697
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;ejemplo.com.     IN  A

;; ANSWER SECTION:
ejemplo.com.    21599 IN  A 127.0.0.1

;; Query time: 135 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: dom may 31 14:32:38 CEST 2020
;; MSG SIZE  rcvd: 56
```

3.5 Bind9 como servidor secundario

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.

2. En el servidor maestro configuras la sección correspondiente al servidor master en el fichero `/etc/bind/named.conf.local`:

   a. 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.

   b. 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.

4. Servicio de directorio

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:
    a. 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.
    b. 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:
    a. ¿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.

b. ¿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.

c. ¿Qué pasaría si quisieras buscar en la guía telefónica un teléfono por la calle donde vive el usuario?

d. ¿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.

   a. ¿Cómo impides que un usuario no pueda buscar un teléfono en la guía telefónica?

   b. ¿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:

   a. ¿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?

   b. ¿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.

4.1 ¿Para qué usar un servicio de directorio?

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..., 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...

4.2 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 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:

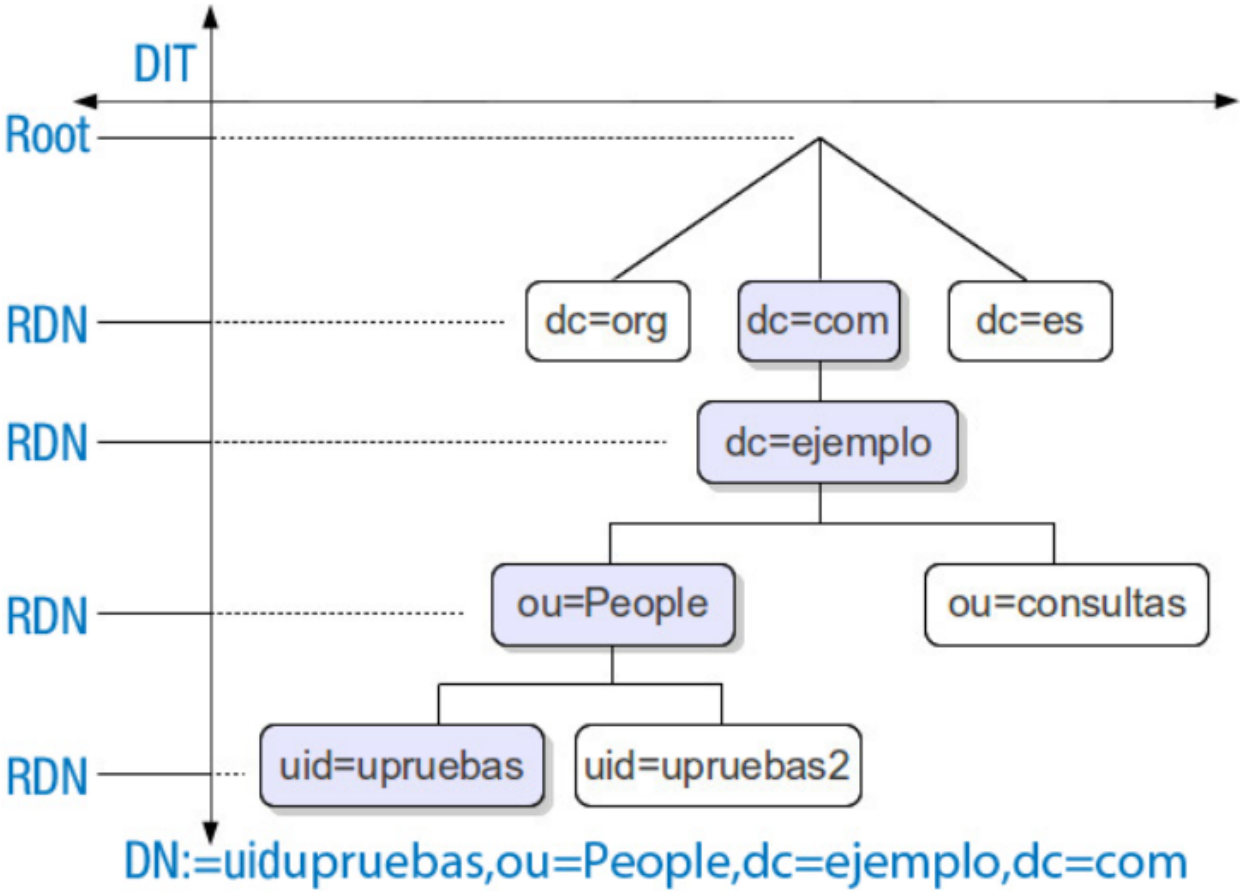
- Contenedor: Este tipo de objeto puede contener a su vez otros objetos. Algunos ejemplos de estos elementos son: `Root` (elemento raíz del árbol de directorios que no existe en realidad), `c` (country), `ou` (OrganizationalUnit) y `dc` (domainComponent).

  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
```

4.3 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.

4.4 LDIF

El formato LDIF es el estándar para representar entradas del directorio en formato texto ASCII, 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 mas 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
```

5. Instalación y configuración de openLDAP

5.1 Instalación

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 / Ubuntu, realiza el siguiente procedimiento como usuario **root** (o **sudo** en Ubuntu), 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      2400/slapd
tcp6       0      0 :::389                  :::*                    LISTEN      2400/slapd
```

5.2 Configuración

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`.

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
```

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:

```
cat /etc/ldap/slapd.d/cn=config/olcDatabase\=\{1\}hdb.ldif \
| sed -e "s/dc=nodomain/dc=proyecto-empresa,dc=local/g" > a.txt

mv a.txt /etc/ldap/slapd.d/cn=config/olcDatabase\=\{1\}hdb.ldif
```

5.3 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.

5.4 Administración del 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.

1. Añadir entradas: comando `ldapadd`.

   a. Crea la estructura básica del dominio LDAP mediante la ejecución de un fichero estructuraBasica.ldif

   ```
   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

   ```
   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 cambiarUsuario.ldif

```
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':

```
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':

```
ldapsearch -x -b dc=proyecto-empresa,dc=local "(&(cn=*pru*)(mail=*05*))"
```

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

```
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`

El contenido de los ficheros es el siguiente:

```
###        INICIO         ###
### estructuraBasica.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

###          FIN          ###

###        INICIO         ###
###      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

###          FIN          ###

###        INICIO         ###
###      cambiar.ldif     ###

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

###          FIN          ###
```

5.5 Configuración de los clientes

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:`

   ```
   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:
   a. 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
```

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

5.5.1 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.