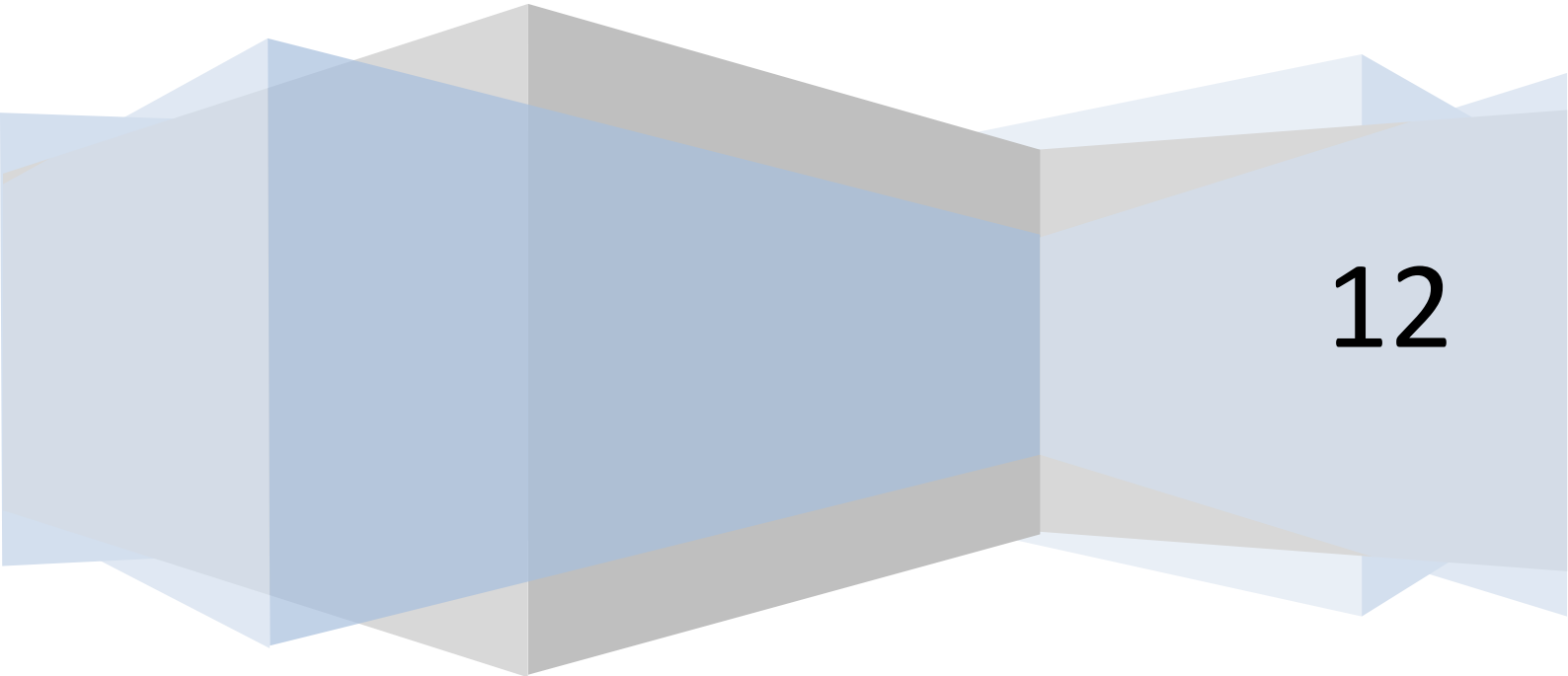


DESPLIEGUE DE APLICACIONES WEB

Desarrollo de Aplicaciones Web

José Luis Comesaña



12

ÍNDICE

1.- Aspectos generales de arquitecturas web.	- 2 -
1.1.- Evolución de los servicios web.	- 3 -
1.2.- Tecnologías asociadas a las aplicaciones web.	- 4 -
1.3.- Tipos de aplicaciones web.	- 5 -
1.4.- Arquitecturas web. Modelos.	- 6 -
1.5.- Plataformas web libres y propietarias.	- 8 -
1.6.- Escalabilidad.	- 9 -
Escalabilidad vertical.	- 9 -
Escalabilidad horizontal.	- 9 -
Cluster.	- 10 -
2.- Servidor web Apache.	- 11 -
2.1.- Instalación y configuración.	- 11 -
2.2.- Iniciar Apache.	- 14 -
3.- Aplicaciones web y servidores de aplicaciones.	- 15 -
3.1.- El servidor de aplicaciones Tomcat.	- 16 -
3.1.1.- Instalación y configuración básica.	- 17 -
3.1.2.- Iniciar Tomcat.	- 18 -
4.- Estructura y despliegue de una aplicación web.	- 20 -
4.1.- Archivos WAR.	- 20 -
4.2.- Despliegue de aplicaciones con Tomcat.	- 21 -
4.3.- Descriptor de despliegue.	- 22 -

Implantación de arquitecturas web.

Caso práctico

Juan, con su perfil de Técnico Superior en Desarrollo de Aplicaciones Informáticas, va a trabajar para la empresa **BK programación**. Por ese motivo, ha decidido documentar, en una wiki interna para los miembros de dicha empresa, los conocimientos que va a ir adquiriendo durante su vida laboral respecto a los trabajos que le toque realizar. De este modo, a los compañeros que le sucedan o que tengan que realizar labores similares les será útil esta información.

Juan ha pensado en incluir un tema titulado **Implantación de arquitecturas web** que estructurará en varios apartados.

En un principio, considera importante realizar un resumen claro y conciso del concepto de arquitecturas web en donde pretende abarcar como mínimo los siguientes puntos: evolución de los servicios web, tecnologías asociadas a las aplicaciones web, tipos de aplicaciones web, arquitecturas web, etc. También pretende incluir en otro de los puntos el servidor web Apache, abarcando los siguientes apartados: instalación, configuración básica e inicio de Apache, y está pensando documentar los mismos puntos para el servidor Tomcat.

Finalmente, piensa crear un último punto en donde explicar el proceso de despliegue de una aplicación web; explicando el despliegue de contenido estático y el de una aplicación web Java, la estructura de carpetas y recursos de una aplicación web y el descriptor del despliegue.

Para realizar este trabajo Juan ha contado con la colaboración de sus amigos Carlos y Ana, estudiantes ambos de Ciclos de Informática.

1.- Aspectos generales de arquitecturas web.

Caso práctico

Juan ha decidido empezar la documentación de la wiki explicando los aspectos básicos de la arquitectura web, ya que considera que es un medio que es necesario conocer con precisión si se pretende realizar el despliegue de una aplicación web. Para ello ha pensado en definir conceptos básicos de las interfaces web.

La arquitectura World Wide Web (WWW) de Internet provee un modelo de programación sumamente poderoso y flexible. Las aplicaciones y los contenidos son presentados en formatos de datos estándar y son localizados por aplicaciones conocidas como "web browsers", que envían requerimientos de objetos a un servidor y éste responde con el dato codificado según un formato estándar. Los estándares WWW especifican muchos de los mecanismos necesarios para construir un ambiente de aplicación de propósito general, por ejemplo:

- ✓ **Modelo estándar de nombres:** todos los servidores, así como el contenido de la WWW se denominan según un Localizador Uniforme de Recursos (Uniform Resource Locator: URL).
- ✓ **Contenido:** a todos los contenidos en la WWW se les especifica un determinado tipo permitiendo de esta forma que los browsers (navegadores) los interpreten correctamente.
- ✓ **Formatos de contenidos estándar:** todos los navegadores soportan un conjunto de formatos estándar, por ejemplo HTML, ECMA, JavaScript, etc.
- ✓ **Protocolos estándar:** éstos permiten que cualquier navegador pueda comunicarse con cualquier servidor web. El más comúnmente usado en WWW es HTML (Protocolo de Transporte de Hipertexto), que opera sobre el conjunto de protocolos TCP/IP.

Esta infraestructura permite a los usuarios acceder a una gran cantidad de aplicaciones y servicios de terceros. También permite a los desarrolladores crear aplicaciones y servicios para una gran comunidad de clientes.

Los aspectos generales a destacar en una arquitectura web son los siguientes:

- ✓ Escalabilidad.
- ✓ Separación de responsabilidades.
- ✓ Portabilidad.
- ✓ Utilización de componentes en los servicios de infraestructura.
- ✓ Gestión de las sesiones del usuario.
- ✓ Aplicación de patrones de diseño.

El esquema de funcionamiento de los servicios web requiere de tres elementos fundamentales:

1. **Proveedor del servicio web**, que es quien lo diseña, desarrolla e implementa y lo pone disponible para su uso, ya sea dentro de la misma organización o en público.
2. **Consumidor del servicio**, que es quien accede al componente para utilizar los servicios que éste presta.
3. **Agente del servicio**, que sirve como enlace entre proveedor y consumidor para efectos de publicación, búsqueda y localización del servicio.

De forma genérica podríamos decir que la arquitectura web es un modelo compuesto de tres capas:

1. **Capa de Base de Datos**, donde estaría toda la documentación de la información que se pretende administrar mediante el servicio web y emplearía una plataforma del tipo MySQL, PostgreSQL, etc.
2. En una segunda capa estarían los **servidores de aplicaciones web**, ejecutando aplicaciones de tipo Apache, Tomcat, Resin, etc.
3. En una tercera capa estarían los **clientes del servicio web** al que accederían mediante un navegador web como Firefox, Internet Explorer, Opera, etc.

1.1.- Evolución de los servicios web.

Caso práctico

Juan ha enviado un correo electrónico a Ana solicitándole la ayuda para poder documentar la evolución de los servicios web en la wiki de su empresa.

Ana ha accedido encantada a su petición ya que está muy interesada en colaborar con la empresa en la que Juan trabaja, llegando incluso a pedirle a éste que le cree un usuario para poder acceder a la wiki interna de **BK programación** y ella misma documentar parte de los puntos que Juan tiene pensado redactar.

La evolución del uso de Servicios web en las organizaciones está fuertemente ligada al desarrollo de Internet como red prestadora de servicios. Entre los factores que han impulsado el uso de los servicios web se encuentran:

- ✓ El **contenido se está volviendo más dinámico**: Los sitios web actuales proporcionan contenidos "instantáneos". Un Servicio web debe ser capaz de combinar contenido proveniente de fuentes muy diferentes.
- ✓ El **ancho de banda es menos costoso**: Actualmente un Servicio web puede entregar tipos variables de contenidos como vídeo o audio. A medida que crezca el ancho de banda, los servicios web deben adaptarse a nuevos tipos de contenidos.
- ✓ El **almacenamiento es más barato**: Un Servicio web debe ser capaz de manejar cantidades masivas de datos, y debe poder hacerlo de forma inteligente.
- ✓ El **éxito de la computación extendida** se está volviendo más importante: Con cientos de millones de dispositivos como teléfonos móviles, agendas electrónicas, etc. existentes actualmente, estamos llegando a un momento en el cual las computadoras están dejando de ser el dispositivo más común en Internet. A medida que las plataformas se hacen más diversas, tecnologías como XML se volverán más importantes. Un servicio web no puede exigir que los usuarios ejecuten, por ejemplo, un navegador web tradicional en alguna versión de Microsoft Windows; por el contrario, los servicios web deben servir a todo tipo de dispositivos, plataformas y navegadores, entregando contenido sobre una amplia variedad de tipos de conexión.

Estos factores, unidos a los beneficios proporcionados por los servicios web en la organización y los buenos productos disponibles para su desarrollo, han hecho que su utilización se extienda sin mayores obstáculos.

En términos generales, cuando se empiezan a utilizar servicios web en una organización, estos se desarrollan e implementan como servicios simples, que poco a poco se van integrando hasta llegar a servicios web mucho más complejos.

En los orígenes del mundo web nos situábamos ante un entorno estático, con páginas en formato HTML que raramente sufrían modificaciones o actualizaciones y en las que apenas había interacción con el usuario.

La **Web 2.0** es la transición que se ha dado desde las aplicaciones tradicionales hacia aplicaciones que funcionan a través de la web y que están fuertemente enfocadas al usuario final. En este nuevo entorno existen una serie de nuevas tecnologías que, en general, tienen como objetivo:

- ✓ Transformar software de escritorio hacia la web.
- ✓ Separar hojas de estilo.
- ✓ Potenciar el trabajo colaborativo y la utilización de redes sociales.
- ✓ Dar control total a los usuarios en el manejo de su información.

1.2.- Tecnologías asociadas a las aplicaciones web.

Caso práctico

Carlos, debido a su afición por el diseño web, ha ofrecido a su amigo Juan realizar un estudio sobre las tecnologías que se emplean actualmente en esta materia. A Juan le ha parecido una idea magnífica puesto que le va a servir para el desarrollo de su wiki.

Las aplicaciones web emplean páginas dinámicas, éstas se ejecutan en un servidor web y se muestran en el navegador de un equipo cliente que es el que ha realizado previamente la solicitud. Cuando una página web llega al navegador, es posible que también incluya algún programa o fragmento de código que se deba ejecutar. Ese código, normalmente en lenguaje JavaScript, **lo ejecutará el propio navegador**. Es por ello que en este apartado nos centraremos en las tecnologías asociadas a las aplicaciones web que se ejecutarán tanto del lado del servidor como del cliente, especificando lo que corresponda en cada uno de los casos.

- ✓ **ASP (Active Server Pages):** Las "Páginas Activas" se ejecutan del lado del servidor, de este modo se forman los resultados que luego se mostrarán en el navegador de cada equipo cliente que ha realizado la solicitud. Un buen ejemplo de ello son los buscadores, donde un usuario realiza una petición de información y el servidor nos entrega un resultado a medida de nuestra petición.
- ✓ Existen versiones de ASP para Unix y Linux, a pesar de que fue una tecnología desarrollada por Microsoft para la creación dinámica de páginas web ofrecida junto a su servidor IIS.
- ✓ **CGI (Common Gateway Interface):** La "Interface Común de Entrada" es uno de los estándares más antiguos en Internet para trasladar información desde una página a un servidor web. **Este estándar** es utilizado para bases de datos, motores de búsqueda, formularios, generadores de email automático,
- ✓ foros, comercio electrónico, rotadores y mapas de imágenes, juegos en línea, etc.
- ✓ Las rutinas de CGI son habitualmente escritas en lenguajes interpretados como Perl o por lenguajes compilados como C.
- ✓ **CSS (Cascading Style Sheets):** Las "Hojas de Estilo en Cascada" se usan para formatear las páginas web; se trata de separar el contenido de un documento de su presentación. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS.
- ✓ **Java:** Este es un lenguaje que trabaja en el cliente, es decir: se ejecuta en el navegador del equipo cliente y no en el servidor. Es un lenguaje eficiente y muy poderoso, que se caracteriza por:
 - ➔ Una misma aplicación puede funcionar en diversos tipos de ordenadores y sistemas operativos: Windows, Linux, Solaris, MacOS, etc., así como en otros dispositivos inteligentes.
 - ➔ Los programas Java pueden ser aplicaciones independientes (que corren en una ventana propia) o "applets", que son pequeños programas interactivos que se encuentran incrustados en una página web y pueden funcionar con cualquier tipo de navegador: Explorer, Netscape, Ópera, etc.
 - ➔ Se trata de un lenguaje "orientado a objetos". Esto significa que los programas se construyen a partir de módulos independientes, y que estos módulos se pueden transformar o ampliar fácilmente. Un equipo de programadores puede partir de una aplicación existente para extenderla con nuevas funcionalidades.
 - ➔ Desarrollado por la empresa Sun Microsystems, pero posteriormente liberado bajo licencia GNU GPL (*La Licencia Pública General de GNU, o más conocida por su nombre en inglés GNU General Public License es una licencia creada por la "Free Software Foundation" y está orientada, principalmente, a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios. El proyecto GNU (GNU es un acrónimo recursivo para "GNU No es Unix"). Comenzó en 1984 a desarrollar un sistema operativo completo con la principal propiedad de ser Software Libre*), con lo cual es un software libre.
- ✓ **JavaScript:** Lenguaje que se interpreta y se ejecuta en el cliente. Útil para realizar tareas como mover imágenes por la pantalla, crear menús de navegación interactivos, utilizar algunos juegos,



etc. En las páginas web suele preferirse JavaScript porque es aceptado por muchos más navegadores que VBScript (creado por Microsoft)

- ✓ **PHP (Hypertext Preprocessor):** Este lenguaje es, como ASP, ejecutado en el lado del servidor. PHP es similar a ASP y puede ser usado en circunstancias similares. Es muy eficiente, permitiendo el acceso a bases de datos empleando servidores como MySQL (*potente gestor de bases de datos relacional, sencillo de usar e increíblemente rápido. También es uno de los motores de bases de datos más usados en Internet, la principal razón de esto es que se distribuye bajo la licencia GNU GPL para aplicaciones no comerciales*) y, por lo tanto, suele utilizarse para crear páginas dinámicas complejas.
- ✓ **VBScript (Visual Basic Scripting):** La respuesta de Microsoft a JavaScript. VBScript es una buena herramienta para cualquier sitio destinado a ser mostrado exclusivamente en el navegador Microsoft Internet Explorer. El código en VBScript puede, además, estar diseñado para su ejecución en el lado del cliente o en el del servidor, la diferencia es que un código que se ejecuta en el lado del servidor no es visible en el lado del cliente. Éste recibe los resultados, pero no el código.



¿Podemos ver una página web sin que intervenga un servidor web?



Sí



No

Podemos ver páginas web con extensión .htm, .html o .xhtml que tengamos almacenadas en nuestro equipo simplemente abriéndolas con el navegador. En este caso la única utilidad del servidor web es enviar la página que solicitamos a nuestro equipo.

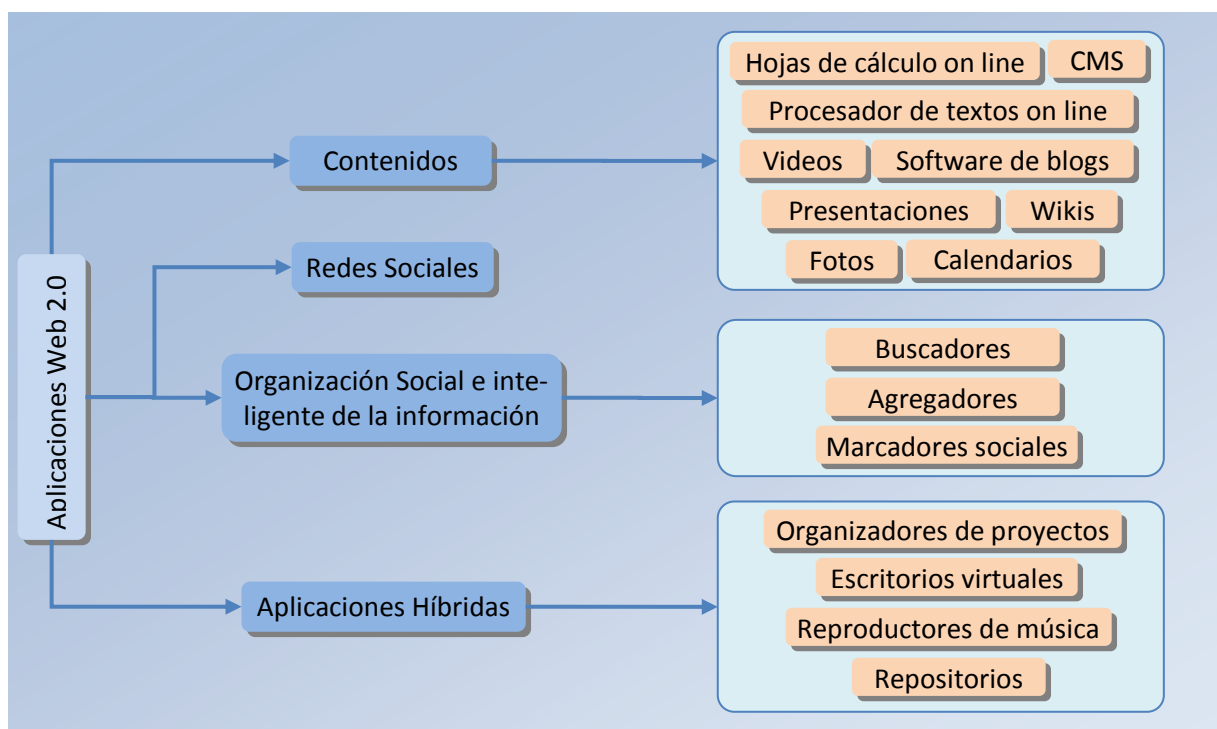
1.3.- Tipos de aplicaciones web.

Caso práctico

Tras una gran labor de documentación acerca de las tecnologías de aplicaciones web, Carlos es consciente de que existe una gran diversidad de aplicaciones disponibles en la web. Por ello, ha decidido documentar un apartado denominado "Tipos de aplicaciones web" que puede ser de utilidad para la wiki que está creando su amigo Juan.



Cualquier proyecto que se quiera desarrollar en Internet, bien sea comercio electrónico, reservas de billetes de vuelo on-line, información meteorológica, registro de usuarios, simuladores de hipotecas, etc, conlleva el desarrollo de una aplicación web. En definitiva, una aplicación web es una plataforma orientada a automatizar los procesos de servicios que se quieran ofrecer a usuarios.



Establecer una clasificación de los tipos de aplicaciones web es una tarea compleja debido a la dificultad existente para poder establecer algún parámetro en función del cual establecer dicha clasificación, junto con la innumerable cantidad de aplicaciones existentes en el actual entorno **web 2.0**.

En función de cómo se presenta la aplicación web junto con el contenido que pretende mostrar, se ha establecido la siguiente clasificación:

- ✓ **Página web Estática.** Están implementadas en HTML y pueden mostrar en alguna parte de la página objetos en movimiento tales como banners, GIF animados, vídeos, etc.
- ✓ **Página web Animada.** Se realizan con la tecnología FLASH; ésta permite que una página web presente el contenido con ciertos efectos animados continuados. El uso de esta tecnología permite diseños más vanguardistas, modernos y creativos.
- ✓ **Página web Dinámica.** Existen muchos lenguajes de programación que son la base para la mayoría de páginas web dinámicas. Los que destacamos aquí son los lenguajes PHP y ASP. Estos lenguajes permiten una perfecta estructuración del contenido. Por una parte crearíamos la estructura de las páginas web y por otra, almacenaríamos el contenido en determinados archivos. A partir de ahí, crearíamos el código de llamada, que insertaría el contenido en la propia página web estructurada. Este es el principio básico que siguen los lenguajes de programación. A partir de aquí se desarrollan aplicaciones para poder gestionar el contenido a través de un panel de control.
- ✓ **Portal.** Es un sitio web que en su página principal permite el acceso a múltiples secciones que, por lo general, son foros, chats, cuentas de correo, buscador, acceso registrado para obtener ciertas ventajas, las últimas noticias de actualidad, etc.
- ✓ **Tienda virtual o comercio electrónico.** Sitio web que publica los productos de una tienda en Internet. Permite la compra on-line a través de tarjeta de crédito, domiciliación bancaria o transferencia bancaria en general. Ofrece al administrador un panel de gestión para poder subir los productos, actualizarlos, eliminarlos, etc.
- ✓ **Página web con "Gestor de Contenidos".** Se trata de un sitio web cuyo contenido se actualiza a través de un panel de gestión por parte del administrador del sitio. Este panel de gestión suele ser muy intuitivo y fácil de usar. En aquellas páginas web que requieran una actualización constante, se suele incorporar este panel de gestión para que la web pueda controlarse día a día por parte del cliente.

Cuando adquirimos un equipo informático nuevo, existen una serie de aplicaciones imprescindibles que es necesario instalar junto con los drivers de nuestro equipo para poder empezar a utilizarlo. Entre estas aplicaciones encontramos aplicaciones ofimáticas, antivirus, aplicaciones de mensajería, compresores, visualizadores, reproductores multimedia, etc.

¿En algún momento te has parado a pensar qué cantidad de aplicaciones web hay disponibles en Internet para substituir a las que tienes pensado instalar en el equipo?

1.4.- Arquitecturas web. Modelos.

Caso práctico

En la wiki que Juan, junto con sus amigos está desarrollando, ha decidido integrar un punto en el cual explicar los diversos tipos de modelos de arquitecturas web que se han utilizado a lo largo del tiempo. La finalidad es tener en cuenta las características de cada uno de ellos y, en función de las mismas, distinguir sus ventajas e inconvenientes.

Se puede establecer que la arquitectura de un sitio web comprende los sistemas de organización y estructuración de los contenidos junto con los sistemas de recuperación de información y navegación

que provea el sitio web, con el objetivo de servir de ayuda a los usuarios a encontrar y manejar la información.

Centraremos el estudio de los modelos de arquitectura web relacionados, en función de cómo implementan cada una de las capas establecidas en una aplicación web:

1. **Capa de presentación** es la encargada de la navegabilidad, validación de los datos de entrada, formateo de los datos de salida, presentación de la web, etc.; se trata de la capa que se presenta al usuario.
2. **Capa de negocio** es la que recibe las peticiones del usuario y desde donde se le envían las respuestas; en esta capa se verifican que las reglas establecidas se cumplen.
3. **Capa de acceso a datos** es la formada por determinados gestores de datos que se encargan de almacenar, estructurar y recuperar los datos solicitados por la capa de negocio.

La evolución experimentada por los medios informáticos en los últimos años ha convivido con otra evolución paralela, la evolución de la arquitectura de las aplicaciones web, que permite aprovechar las nuevas características que éstas ofrecen. De esta forma, el modelo arquitectónico de las aplicaciones de Internet ha sufrido dos grandes saltos, con algún paso intermedio, desde la aparición de los primeros portales web. Los distintos modelos de aplicación sobre los que se ha ido desarrollando, según diversos autores, se podrían clasificar del siguiente modo:

✓ **Modelo 1**

En este caso las aplicaciones se diseñan en un modelo web CGI, basadas en la ejecución de procesos externos al servidor web, cuya salida por pantalla era el HTML que el navegador recibía en respuesta a su petición. Presentación, negocio y acceso a datos se confundían en un mismo script perl (*Lenguaje de programación diseñado por Larry Wall en 1987. Perl toma características del lenguaje C, del lenguaje interpretado shell (sh), awk, sed, Lisp y, en un grado inferior, de muchos otros lenguajes de programación*).

✓ **Modelo 1.5**

Aplicado a la tecnología java (*Lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90, aunque a finales de 2006 liberó la mayor parte de sus tecnologías Java bajo la licencia GNU GPL*), se da con la aparición de las JSP y los servlets (*Objetos que se ejecutan dentro del contexto de un contenedor de "servlets", por ejemplo Tomcat y amplían su funcionalidad. La palabra servlet deriva de otra anterior, applet, que se refería a pequeños programas que se ejecutan en el contexto de un navegador web. Por contraposición, un servlet es un programa que se ejecuta en un servidor. El uso más común de los servlets es generar páginas web de forma dinámica a partir de los parámetros de la petición que envíe el navegador web*). En este modelo, las responsabilidades de presentación recaen en las páginas JSP, mientras que los beans (*Abreviatura científica del botánico Willian Jackson Bean (1863-1947). Un bean es un componente software que tiene la particularidad de ser reutilizable y así evitar la tediosa tarea de programar los distintos componentes uno a uno*) incrustados en las mismas son los responsables del modelo de negocio y acceso a datos.

✓ **Modelo 2**

Como evolución del modelo anterior, con la incorporación del patrón MVC en este tipo de aplicaciones, se aprecia la incorporación de un elemento controlador de la navegación de la aplicación. El modelo de negocio queda encapsulado en los javaBeans (*Modelo de componentes creado por Sun Microsystems para la construcción de aplicaciones en Java; se usan para encapsular varios objetos en un único objeto (bean), para hacer uso de un solo objeto en lugar de varios más simples. La especificación de JavaBeans los define como "componentes de software reutilizables que se puedan manipular visualmente en una herramienta de construcción"*) que se incrustan en las páginas JSP.

✓ **Modelo 2X**

Aparecen con el objetivo de dar respuesta a la necesidad, cada vez más habitual, de desarrollar aplicaciones multicanal, es decir, aplicaciones web que pueden ser atacadas desde distintos tipos de clientes remotos. Así, una aplicación web multicanal podrá ejecutarse desde una PDA, desde un terminal de telefonía móvil, o desde cualquier navegador HTML estándar. El medio para lograr publicar la misma aplicación para distintos dispositivos es emplear plantillas XSL para transformar los datos XML.

Esta web está pensada como un curso en español de Java básico. Pretende tener una interacción con los lectores, de forma que se puedan resolver las dudas que surjan.

<http://java-spain.com/bienvenido-java-spaincom>

1.5.- Plataformas web libres y propietarias.

Caso práctico

El diseño de aplicaciones web requiere de un entorno funcional, que permita el desarrollo de cada uno de los componentes que forman la aplicación web, junto con un entorno complejo de herramientas que ofrecen al cliente los servicios de las aplicaciones web; para todo ello, el mercado pone a disposición de los programadores un abanico de herramientas software, que Juan pretende analizar en la wiki de la empresa BK programación, estableciendo el criterio de clasificación que considera primordial: software libre o software propietario.



Una plataforma web es el entorno de desarrollo de software empleado para diseñar y ejecutar un sitio web. En términos generales, una plataforma web consta de cuatro componentes básicos:

1. El **sistema operativo**, bajo el cual opera el equipo donde se hospedan las páginas web y que representa la base misma del funcionamiento del computador. En ocasiones limita la elección de otros componentes.
2. El **servidor web** es el software que maneja las peticiones desde equipos remotos a través de la Internet. En el caso de páginas estáticas, el servidor web simplemente provee el archivo solicitado, el cual se muestra en el navegador. En el caso de sitios dinámicos, el servidor web se encarga de pasar las solicitudes a otros programas que puedan gestionarlas adecuadamente.
3. El **gestor de bases de datos** se encarga de almacenar sistemáticamente un conjunto de registros de datos relacionados para ser usados posteriormente.
4. Un **lenguaje de programación interpretado** que controla las aplicaciones de software que corren en el sitio web.

Diferentes combinaciones de los cuatro componentes señalados, basadas en las distintas opciones de software disponibles en el mercado, dan lugar a numerosas plataformas web, aunque, sin duda, hay dos que sobresalen del resto por su popularidad y difusión: LAMP y WISA.

La plataforma LAMP trabaja enteramente con componentes de **software libre** y no está sujeta a restricciones propietarias. El nombre **LAMP** surge de las iniciales de los componentes de software que la integran:

- ✓ **Linux**: Sistema operativo.
- ✓ **Apache**: Servidor web.
- ✓ **MySQL**: Gestor de bases de datos.
- ✓ **PHP**: Lenguaje interpretado PHP, aunque a veces se sustituye por Perl o Python.

La plataforma **WISA** está basada en tecnologías desarrolladas por la compañía Microsoft; se trata, por lo tanto, de **software propietario**. La componen los siguientes elementos:

- ✓ **Windows**: Sistema operativo.
- ✓ **Internet Information Services**: servidor web.
- ✓ **SQL Server**: gestor de bases de datos.
- ✓ **ASP o ASP.NET**: como lenguaje para scripting del lado del servidor.

Existen otras plataformas, como por ejemplo la configuración Windows-Apache-MySQL-PHP que se conoce como **WAMP**. Es bastante común pero sólo como plataforma de desarrollo local.

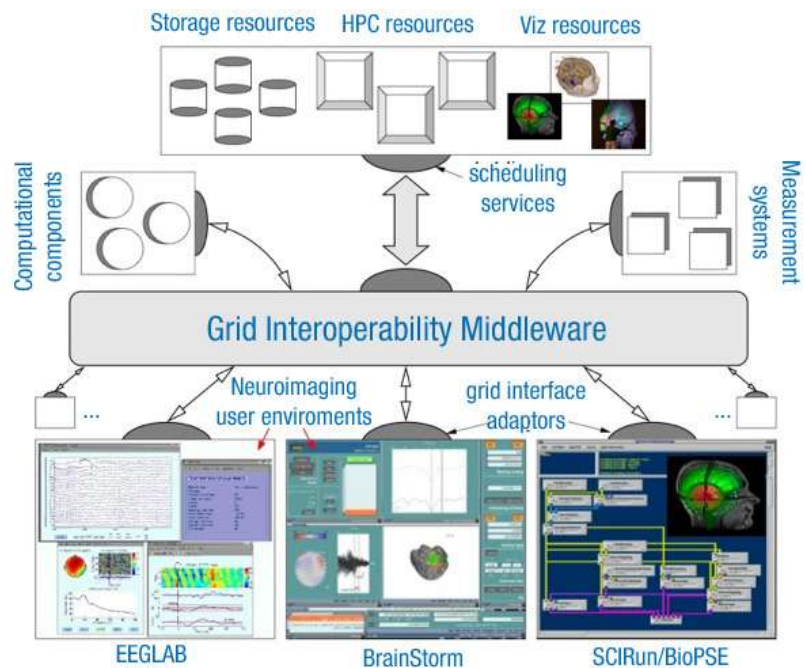
De forma similar, un servidor Windows puede correr con MySQL y PHP. A esta configuración se la conoce como plataforma **WIMP**.

Existen muchas otras plataformas que trabajan con distintos sistemas operativos (Unix, MacOS, Solaris), servidores web (incluyendo algunos que se han cobrado relativa popularidad como Lighttpd y LiteSpeed), bases de datos (Postgre SQL) y lenguajes de programación.

1.6.- Escalabilidad.

Las aplicaciones web se ejecutan en un entorno donde el número de clientes que solicitan el servicio puede variar en gran medida en función del momento. Es por ello que hay una característica de esencial importancia como es la escalabilidad, al que Juan ha dedicado un apartado de su wiki para documentar esta característica.

En el entorno en que se ubican las aplicaciones web, uno de los principales factores que puede afectar al rendimiento de las mismas es el número de usuarios, ya que éste puede verse incrementado de forma vertiginosa en un periodo de tiempo relativamente corto. El éxito o el fracaso de un sitio web orientado al usuario común vendrá determinado, entre otros aspectos, por el dimensionamiento del sistema sobre el que se instala y soporta el software que sustenta dicho sitio. En consecuencia, uno de los requisitos fundamentales de una aplicación web es que sea completamente escalable sin que un aumento de los recursos dedicados a la misma suponga modificación alguna en su comportamiento o capacidades.



La escalabilidad de un sistema web puede ser:

- ✓ Verticalmente: de manera ascendente "upgrades" a cada nodo.
- ✓ Horizontalmente: consiste en aumentar el número de nodos.
- ✓ Cluster: consiste en crear agrupaciones de servidores.

Escalabilidad vertical.

Habitualmente, la separación lógica en capas se implementa de tal forma que se permita una separación física de las mismas. Interponiendo elementos conectores que actúen de middlewares es posible distribuir la aplicación de forma vertical (una máquina por cada capa del sistema), e incluso si esto no fuera suficiente, distribuyendo los elementos de una misma capa entre distintas máquinas servidoras.

Escalabilidad horizontal.

Se trata de clonar el sistema en otra máquina de características similares y balancear la carga de trabajo mediante un dispositivo externo. El balanceador de carga puede ser:

- ✓ **Balanceador Software:** Por ejemplo, habitualmente encontramos un servidor web apache junto con el módulo **mod_jk**, que permite la redirección de las peticiones http que a tal efecto sean configuradas entre las distintas máquinas que forman la granja de servidores. Este tipo de balanceadores examinan el paquete http e identifican la sesión del usuario, guardando registro de cuál de las máquinas de la granja se está encargando de servir a dicha sesión. Este aspecto es importante, dado que nos permite trabajar (de cara al diseño de la aplicación) apoyándonos en el objeto sesión propio del usuario y almacenando información relativa a la sesión del mismo, puesto que tenemos la garantía de que todas las peticiones de una misma sesión http van a ser redireccionadas hacia la misma máquina.

- ✓ **Balanceador hardware:** Se trata de dispositivos que, respondiendo únicamente a algoritmos de reparto de carga (Round Robin, LRU, etc.), redireccionan una petición http del usuario a la máquina que, según dicho algoritmo, convenga que se haga cargo de la petición. Son mucho más rápidos que los anteriores, dado que se basan en conmutación de circuitos y no examinan ni interpretan el paquete http. Sin embargo, el no garantizar el mantenimiento de la misma sesión de usuario en la misma máquina, condiciona seriamente el diseño, dado que fuerza a que la información relativa a la sesión del usuario sea almacenada por el implementador del mismo, bien en cookies o bien en base de datos.
- ✓ **Balanceador hardware http:** Se trata de dispositivos hardware pero que examinan el paquete http y mantienen la relación usuario-máquina servidora. Mucho más rápidos que los balanceadores software, pero algo menos que los hardware, suponen hoy en día una de las soluciones más aceptadas en el mercado.

Cluster

Con la aparición de los servidores de aplicaciones en cluster se abre una nueva capacidad de escalabilidad que, dependiendo de cómo se aplique, podría clasificarse como vertical u horizontal. Un cluster de servidores de aplicaciones permite el despliegue de una aplicación web corriente, de forma que su carga de trabajo vaya a ser distribuida entre la granja de servidores que forman el cluster, de modo transparente al usuario y al administrador. El cluster, mediante el mecanismo de replicación de sesión, garantiza que sea cual sea la máquina que sirva la petición http, tendrá acceso a la sesión del usuario (objeto **HttpSession** en java). Este tipo de sistemas, debido precisamente a la replicación de sesión, suele presentar problemas de rendimiento.

¿En qué tipo de escalabilidad se emplean los balanceadores de carga?



Horizontal



Vertical

Correcto. Los balanceadores se encargan de repartir la carga entre las distintas máquinas; pudiendo existir tres tipos de balanceadores :

- ✓ Balanceador software: Centran su trabajo basándose en las sesiones establecidas por los usuarios de la aplicación.
- ✓ Balanceador hardware: Se basan en algoritmos de reparto de carga; redireccionando las peticiones http del usuario a la máquina indicada por el algoritmo.
- ✓ Balanceador hardware-http: Situación intermedia entre los dos anteriores.

2.- Servidor web Apache.

Caso práctico

En la empresa **BK programación**, Ada ha solicitado a María la instalación de un servidor web para albergar, entre otras cosas, la wiki que Juan, junto con sus amigos, están construyendo. María, por su parte, ha decidido instalar el servidor web Apache por ser uno de los más empleados y aportar a Juan la información que pueda interesarle acerca de este servidor web para su wiki.

Un servidor web es un programa que se ejecuta de forma continua en un ordenador (también se utiliza el término para referirse al ordenador que lo ejecuta), se mantiene a la espera de peticiones por parte de un cliente (un navegador de Internet) y contesta a estas peticiones de forma adecuada, sirviendo una página web que será mostrada en el navegador o mostrando el mensaje correspondiente si se detectó algún error.

Uno de los servidores web más populares del mercado y el más utilizado actualmente es Apache, de código abierto y gratuito, disponible para Windows y GNU/Linux, entre otros.

En cuanto a su arquitectura podemos destacar lo siguiente:

- ✓ Estructurado en módulos.
- ✓ Cada módulo contiene un conjunto de funciones relativas a un aspecto concreto del servidor.
- ✓ El archivo binario **httpd** contiene un conjunto de módulos que han sido compilados.
- ✓ La funcionalidad de estos módulos puede ser activada o desactivada al arrancar el servidor.
- ✓ Los módulos de Apache se pueden clasificar en tres categorías:
 - ➔ Módulos base: Se encargan de las funciones básicas.
 - ➔ Módulos multiproceso: Encargados de la unión de los puertos de la máquina, aceptando las peticiones y atendíéndolas.
 - ➔ Módulos adicionales: se encargan de añadir funcionalidad al servidor.

El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation. La licencia de software, bajo la cual el software de la fundación Apache es distribuido, es una parte distintiva de la historia de Apache HTTP Server y de la comunidad de código abierto.

La **Licencia Apache** permite la distribución de derivados de código abierto y cerrado a partir de su código fuente original.

Esta web sirve como manual de referencia, guía de usuario, tutoriales prácticos, etc., sobre el servidor web Apache. Se trata de la web oficial de Apache Software Foundation.

<http://httpd.apache.org/docs/2.0/es/>

2.1.- Instalación y configuración.

Caso práctico

En la empresa **BK programación**, Ada ha solicitado a María la instalación de un servidor web para albergar, entre otras cosas, la wiki que Juan, junto con sus amigos, está construyendo. María debe instalar el servidor Apache debido a que es uno de los más empleados. Se trata de una herramienta de código libre y que funciona en multitud de plataformas; en este caso, será instalado en una máquina Debian 6.0.1 Squeeze, quedando el servidor totalmente operativo.

Vamos a realizar la instalación de un servidor Apache en una máquina con la distribución Debian 6.0.1 Squeeze.



Empezamos por identificarnos en la máquina con el usuario **root** y, a continuación, ejecutamos:

```
# apt-get install apache2
```

Debido a que pretendemos montar una plataforma LAMP, por sus ventajas derivadas de las características del software libre, instalaremos también los siguientes componentes: MySQL y PHP.

```
# apt-get install php5 mysql-client mysql-admin mysql-query-browser phpmyadmin
```

Una vez instalado, para verificar si funciona, podemos hacerlo desde un navegador, escribiendo en la barra de direcciones :

```
http://localhost ó http://127.0.0.1
```

o bien, si accedemos desde otro equipo de la red a la dirección IP de esta máquina, deberíamos obtener una página con el mensaje "**It Works!**", confirmando así su correcto funcionamiento.

Otro método de operar es descargar el código fuente de la aplicación desde la web del proyecto Apache; luego descomprimir, compilar e instalar; realizar el proceso empleando los siguientes comandos:

```
cd /usr/local/src
wget http://apache.rediris.es/httpd/httpd-2.2.19.tar.gz
tar xvfz httpd-2.2.19.tar.gz
cd /usr/local/src/httpd-2.2.19
./configure --prefix=/usr/local/apache --enable-module=most --enable-mods-shared=most
make
make install
```

Por defecto, Apache sirve las páginas web que están en la carpeta "`/var/www`"; si nos situamos en esa carpeta, encontramos un archivo "`index.html`" que es el que contiene el "It Works!". En esta carpeta podemos crear nuevas carpetas en donde ubicaremos nuevas páginas web que deseamos servir, todas ellas accesibles a través del puerto 80.

Si la única pretensión es servir una página web, podemos integrar su contenido aquí. En caso que se pretenda servir más páginas web, es más recomendable la utilización de los **Hosts Virtuales**; para ello accedemos a la carpeta "`/etc/apache2/sites-enabled`", donde hay un fichero llamado "`000-default`", que nos va a servir de ejemplo para la creación de hosts virtuales, los cuales van a permitir servir varias web desde una sola dirección IP utilizando para cada una un puerto distinto.

Apache se configura colocando directivas en archivos de configuración de texto plano. El archivo principal de configuración se llama `apache2.conf`. Además, se pueden añadir otros archivos de configuración mediante la directiva "`Include`", y se pueden usar comodines para incluir muchos archivos de configuración. Todas las directivas deben colocarse en alguno de esos archivos de configuración. Apache2 sólo reconocerá los cambios realizados en los archivos principales de configuración cuando se inicie o se reinicie.

Como ya hemos comentado, el archivo de configuración predeterminado de Apache2 es `/etc/apache2/apache2.conf`. Se puede editar este archivo para configurar el servidor Apache2, para configurar el número de puerto, la raíz de documentos, los módulos, los archivos de registros, los hosts virtuales, etc. Pasamos a ver alguna de las principales directivas:

- ✓ `ServerTokens`, para configurar la cantidad de información que Apache aporta sobre sí mismo.
- ✓ `ServerSignature`, para indicar datos sobre Apache en el pie de los mensajes de error.
- ✓ `Alias` permite direccionar a una carpeta que puede estar fuera del árbol de directorios especificado en `DocumentRoot`.
- ✓ `UserDir` permite redireccionar al directorio personal del usuario si se recibe una solicitud de tipo `~usuario`.

Para modificar el servidor virtual predeterminado, editamos el archivo `/etc/apache2/sites-available/default`. En el caso de querer configurar un nuevo servidor o sitio virtual, copiaríamos ese archivo dentro del mismo directorio con el nombre que se haya elegido, y editaríamos el nuevo archivo para configurar el nuevo sitio usando algunas de las directivas que se describen a continuación:

- ✓ **ServerName**, en el caso de no tener un dominio registrado emplearíamos localhost.
- ✓ **CustomLog** define el archivo .log donde se guardan los logs de acceso.
- ✓ **ServerAdmin** especifica la dirección de correo del administrador del servidor. El valor por omisión es **webmaster@localhost**.
- ✓ **Listen** especifica el puerto (y, opcionalmente, la dirección IP) por el que escuchará Apache2. La directiva se puede encontrar y cambiar en su propio archivo de configuración, **/etc/apache2/ports.conf**.
- ✓ **DocumentRoot** especifica dónde Apache debe buscar los archivos que forman el sitio. El valor predeterminado es **/var/www**.
- ✓ **RedirectMatch** en **/etc/apache2/apache2.conf**, las peticiones se redirigirán a **/var/www/apache2-default**, que es donde reside el sitio predeterminado de Apache2. Cambiar este valor en el archivo de host virtual implica crear ese directorio si fuese necesario.



En vídeo práctico se explica cómo instalar y configurar un servidor web Apache en una máquina virtual con el sistema operativo Ubuntu:

http://www.youtube.com/watch?feature=player_embedded&v=p1kDRqd_JHg

Resumen del vídeo:

Se parte de un sistema operativo Ubuntu con la herramienta Webmin previamente instalada. Webmin es una herramienta de configuración de sistemas accesible vía web para OpenSolaris, GNU/Linux y otros sistemas Unix. Con él se pueden configurar aspectos internos de muchos sistemas operativos, como usuarios, cuotas de espacio, servicios, archivos de configuración, apagado del equipo, etcétera, así como modificar y controlar muchas aplicaciones libres, como el servidor web Apache, PHP, MySQL, entre otros.

En primer lugar, se busca el paquete de Apache en el buscador de la herramienta Webmin, se selecciona el paquete correspondiente y, a continuación, la opción de instalar.

Se comprueba que el servidor web Apache ha sido correctamente instalado, posteriormente se comprueba que es capaz de mostrar una página web que nosotros deseamos que el Apache sirva. Para ello debemos copiar la página a la carpeta **/var/www**, estableciendo, en primer lugar, los permisos correspondientes para trabajar con dicha carpeta.

Una vez copiada la página a la carpeta anterior, se para el servicio correspondiente a Apache y se vuelve a arrancar, se comprueba el funcionamiento correcto del servidor introduciendo en el navegador la URL: **http://127.0.0.1**

Otra parte del vídeo demuestra cómo crear un nuevo Virtual Host y que escuche por el puerto 81, para ello es necesario configurar el archivo **/etc/apache2/ports.conf** para indicar al servidor que escuche por dicho puerto

2.2.- Iniciar Apache.

Caso práctico

En la empresa **BK programación**, Ada ha solicitado a María la instalación de un servidor web para albergar, entre otras cosas, la wiki que Juan, junto con sus amigos, está construyendo.

María debe instalar el servidor Apache debido a que es uno de los más empleados. También debe indicar todos y cada uno de los pasos para arrancar el servidor, así como las indicaciones oportunas para la accesibilidad a las páginas de la wiki desde el servidor Apache.

Si hemos instalado Apache en la ruta `/usr/local/apache`, podemos probar su configuración por defecto e intentar iniciar el servicio de la siguiente forma:

```
/usr/local/apache/bin/apachectl configtest
```



Si todo está correcto debería devolver un mensaje del tipo `"Syntax OK"`

`usr/local/apache/bin/apachectl start` y el servidor debería estar arrancado, con lo cual, si en un navegador introducimos la URL: `http://localhost` veríamos la página de bienvenida de Apache.

Si el puerto especificado en la directiva Listen del fichero de configuración es el que viene por defecto, es decir, el puerto 80 (o cualquier otro puerto por debajo del 1024), entonces es necesario tener privilegios de usuario root (superusuario) para iniciar Apache, de modo que pueda establecerse una conexión a través de esos puertos privilegiados. Una vez que el servidor Apache se ha iniciado y ha completado algunas tareas preliminares, tales como abrir sus ficheros log, lanzará varios procesos, procesos hijo, que hacen el trabajo de escuchar y atender las peticiones de los clientes. El proceso principal, `httpd`, continúa ejecutándose como root, pero los procesos hijo se ejecutan con menores privilegios de usuario.

El demonio `httpd` se debería invocar empleando el script de control `apachectl`, que es el que se encarga de fijar variables de entorno y pasa al demonio (`httpd`) cualquier opción que se le pase como argumento por línea de comandos.

El script `apachectl` es capaz de interpretar los argumentos `start`, `restart`, y `stop` y traducirlos en las señales apropiadas para `httpd`.

Si en cualquier momento deseásemos parar, reiniciar o arrancar el servidor, podríamos emplear los siguientes comandos respectivamente:

```
# /etc/init.d/apache2 stop
# /etc/init.d/apache2 restart
# /etc/init.d/apache2 start
```

Una vez instalado el servidor Apache, es necesario acceder a su funcionalidad y gestionarlo como si de un servicio se tratase, de modo, que cuando establecemos cambios en su configuración, los mismos se vean reflejados.

¿Será necesario reiniciar el servicio Apache si, mediante la creación de un host virtual, hemos cambiado el puerto por el que escucha?

3.- Aplicaciones web y servidores de aplicaciones.

Caso práctico

Hoy en día existen innumerables aplicaciones web. Por eso, en la wiki que está construyendo, Juan ha decidido dedicar un apartado a dos conceptos de vital importancia: Aplicación web y Servidor de Aplicaciones; ambos conceptos estrechamente relacionados.



Se define una aplicación web como una aplicación informática que se ejecuta en un entorno web, de forma que se trata de una aplicación cliente-servidor junto con un protocolo de comunicación previamente establecido:

- ✓ Cliente: navegador.
- ✓ Servidor: servidor web
- ✓ Comunicación: protocolo HTTP

Un **servidor de aplicaciones** es un software que proporciona aplicaciones a los equipos o dispositivos cliente, por lo general, a través de Internet y utilizando el protocolo http. Los servidores de aplicación se distinguen de los servidores web en el uso extensivo del contenido dinámico y por su frecuente integración con bases de datos.

Un servidor de aplicaciones también es una máquina en una red de computadores que ejecuta determinadas aplicaciones, gestionando la mayor parte de las funciones de acceso a los datos de la aplicación.

Las principales ventajas de la tecnología de los servidores de aplicaciones es la **centralización y disminución** de la complejidad en el desarrollo de las aplicaciones, ya que no necesitan ser programadas, sino que son ensambladas desde bloques provistos por el servidor de aplicación.

Otra de las ventajas es la **integridad de datos y código** ya que, al estar centralizada en una o un pequeño número de máquinas servidoras, las actualizaciones están garantizadas para todos los usuarios.

El término servidor de aplicaciones se aplica a todas las plataformas. Dicho término se utiliza para referirse a los servidores de aplicaciones basadas en web, como el control de las plataformas de comercio electrónico integrado, sistemas de gestión de contenido de sitios web y asistentes o constructores de sitios de Internet.

Uno de los ejemplos destacados es el de Sun Microsystems, la plataforma J2EE. Los servidores de aplicaciones Java se basan en la Plataforma Java™ 2 Enterprise Edition (J2EE™). J2EE utiliza un modelo de este tipo y en general, incluye un nivel Cliente, un nivel Medio, y un EIS. El servidor de tipo Cliente puede contener una o más aplicaciones o navegadores. La Plataforma J2EE es del Nivel Medio y consiste en un servidor web y un servidor EJB. (Estos servidores son también llamados "contenedores".) También podría haber subniveles adicionales en el nivel intermedio. El nivel del SistemaEnterprise Information System (EIS, o "Sistema de Información Empresarial") contiene las aplicaciones existentes, archivos y bases de datos.

Esta web detalla 120 aplicaciones disponibles gratuitamente vía web en donde se especifica la función de cada una de ellas.

<http://especial.wetpaint.com/page/120+soluciones+gratis+web+2.0>

3.1.- El servidor de aplicaciones Tomcat.

Caso práctico

Instalar un servidor de aplicaciones web para la empresa **BK programación** ha sido una solicitud que Ada había propuesto hace tiempo a María, quien, llegado el momento, y habiendo estudiado las posibles opciones, se ha decidido por instalar el servidor Tomcat



Tomcat es el servidor web (incluye el servidor Apache) y de aplicaciones del proyecto Yakarta, con lo cual, gestiona las solicitudes y respuestas http y, además, es servidor de aplicaciones o contenedor de Servlets y JSP.

Incluye el compilador Jasper, que compila JSP convirtiéndolas en servlets.

Tomcat es un contenedor de servlets con un entorno JSP. Un contenedor de servlets es un shell de ejecución que maneja e invoca servlets por cuenta del usuario. Podemos dividir los contenedores de servlets en:

1. Contenedores de servlets **stand-alone** (independientes): Estos son una parte integral del servidor web. Este es el caso en el que se usa un servidor web basado en Java, por ejemplo, el contenedor de servlets es parte de JavaWebServer (actualmente sustituido por **iPlanet**). Por defecto Tomcat trabaja en este modo, sin embargo, la mayoría de los servidores no están basados en Java.
2. Contenedores de servlets **dentro-de-proceso**: El contenedor servlets es una combinación de un plugin para el servidor web y una implementación de contenedor Java. El plugin del servidor web abre una JVM (Máquina Virtual Java) dentro del espacio de direcciones del servidor web y permite que el contenedor Java se ejecute en él. En el caso de que una petición debiera ejecutar un servlet, el plugin toma el control sobre la petición y lo pasa al contenedor Java (usando JNI). Un contenedor de este tipo es adecuado para servidores multi-thread de un sólo proceso y proporciona un buen rendimiento pero está limitado en escalabilidad.
3. Contenedores de servlets **fuera-de-proceso**: El contenedor servlets es una combinación de un plugin para el servidor web y una implementación de contenedor Java que se ejecuta en una JVM fuera del servidor web. El plugin del servidor web y el JVM del contenedor Java se comunican usando algún mecanismo IPC (normalmente sockets TCP/IP). Si una cierta petición tuviera que ejecutar un servlet, el plugin toma el control sobre la petición y lo pasa al contenedor Java (usando IPCs). El tiempo de respuesta en este tipo de contenedores no es tan bueno como el anterior, pero obtiene mejores rendimientos en otras cosas (escalabilidad, estabilidad, etc.).

Tomcat puede utilizarse como un contenedor solitario (principalmente para desarrollo y depuración) o como plugin para un servidor web existente (actualmente soporta los servidores Apache, IIS). Esto significa que siempre que desplaguemos Tomcat tendremos que decidir cómo usarlo y, si seleccionamos las opciones 2 o 3, también necesitaremos instalar un adaptador de servidor web.

Las funciones del Servidor Apache y las funciones del servidor Tomcat, ¿son equivalentes?



Sí



No

Básicamente, el servidor Apache es únicamente un servidor web, mientras que el servidor Tomcat es un servidor de aplicaciones.

3.1.1.- Instalación y configuración básica.

En primer lugar, destacar que para instalar cualquier versión de Tomcat es necesario tener instalado JDK (Kit de desarrollo de Java), ya que el objetivo es que las peticiones a Apache se redirijan a Tomcat empleando un conector proporcionado por Java en este caso.



Empezamos buscando el paquete de Java que nos pueda interesar. Con el siguiente comando obtendríamos la lista del entorno Java, debido a que Debian proporciona varias implementaciones, cada uno de estos paquetes tiene un entorno de desarrollo (JDK) y un tiempo de ejecución conocido (JRE o Java Virtual Machines, JVM):

```
aptitude search "?provides(java-runtime)"
```

Instalamos el siguiente paquete por ser el que más se adapta a nuestras necesidades:

```
apt-get install default-jre
```

Una vez instalado el paquete anterior, es necesario crear una variable de entorno para indicar en dónde se ha instalado, y añadir a la variable **PATH** el directorio en donde se encuentran los archivos binarios para que puedan ser invocados desde cualquier parte; para ello, añadimos en nuestro caso las siguientes líneas al archivo `/etc/profile`:

```
JAVA_HOME=/usr/lib/jvm/java-6-openjdk/jre/
PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME
```

Actualizamos las variables de entorno mediante el comando:

```
source /etc/profile
```

Llegado este punto descargamos Tomcat, para ello abrimos en un navegador la URL: <http://apache.rediris.es/tomcat/tomcat-6/>, una vez allí comprobaremos cuál es la última versión estable de Tomcat y, desde la carpeta "bin", copiamos el link de descarga al `apache-tomcat-x.xx.x.tar.gz`. En nuestro caso el link sería: <http://apache.rediris.es/tomcat/tomcat-6/v6.0.32/bin/apache-tomcat-6.0.32.tar.gz> con lo cual podemos emplear el siguiente comando para descargarlo:

```
# wget http://apache.rediris.es/tomcat/tomcat-6/v6.0.32/bin/apache-tomcat-6.0.32.tar.gz
```

Descomprimos el archivo descargado:

```
# tar xvfz apache-tomcat-6.0.32.tar.gz
```

Movemos a la carpeta de destino :

```
# mv -drfv apache-tomcat-6.0.32 /usr/local/
```

Podemos hacer un link para hacer más cómodas las actualizaciones:

```
# ln -s /usr/local/apache-tomcat-6.0.32/ /usr/local/tomcat
```

En Tomcat, la gestión del servicio se realiza a través del script incluido llamado **catalina**, al que le podemos proporcionar los parámetros "start" y "stop", con lo que arrancaríamos o pararíamos el servicio manualmente.

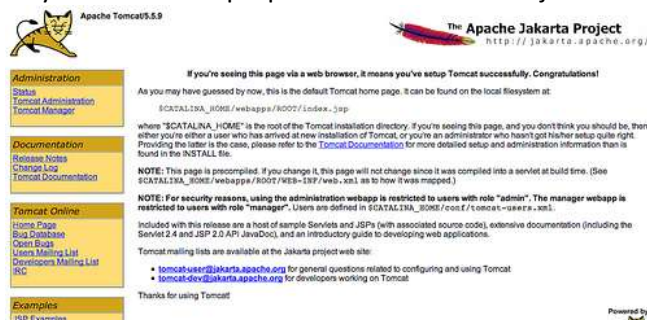
Para comprobar que nuestro servidor está ya escuchando, introducimos en un navegador la URL <http://127.0.0.1:8080/>, y éste debería mostrar la página de inicio de Tomcat.

En la siguiente tabla se resumen los pasos de instalación y puesta en funcionamiento de Tomcat.

INSTALACIÓN DE TOMCAT EN UBUNTU 1 PRERREQUISITOS. 2 DESCARGAR TOMCAT 3 DESCOMPRIMIR ARCHIVO EN CARPETA DESEADA. 4 ARRANCAR TOMCAT	1. PRERREQUISITOS(I): DEBE ESTAR INSTALADO JDK (Kit de desarrollo de Java). - Para buscar el paquete Java que más nos interese #aptitude search "?provides(java-runtime)" - Instalando este paquete ya sería suficiente para trabajar con Tomcat. #apt-get install default-jre
1. PRERREQUISITOS(II): - CREAR VARIABLE DE ENTORNO ASOCIADA A JAVA * Editar /etc/profile y añadir (puede variar la ruta) JAVA_HOME=/usr/lib/jvm/java-6-openjdk/jre/ PATH=\$PATH:\$JAVA_HOME/bin export PATH JAVA_HOME - ACTUALIZAR VARIABLES DE ENTORNO CREADAS #source /etc/profile	2. DESCARGAR TOMCAT: - ACCEDEMOS A LA URL: http://tomcat.apache.org/ Desde donde descargamos la última versión estable de Tomcat. - OTRO MÉTODO PODRÍA SER: wget http://apache.rediri.es/tomcat/tomcat-6/v6.0.32/bin/apache-tomcat-6.0.32.tar.gz
3. DESCOMPRIMIR ARCHIVO EN CARPETA DESEADA: - MOVEMOS EL ARCHIVO DESCARGADO A LA CARPETA DESEADA: * mv apache-tomcat-6.0.32.tar.gz /usr/local - DESCOMPRIMIMOS EL ARCHIVO .tar.gz: * tar xvfz apache-tomcat-6.0.32.tar.gz	4. ARRANCAR TOMCAT. - EL SCRIPT "catalina.sh" ES EL ENCARGADO DE LA GESTIÓN DE TOMCAT. sh /usr/local/apache-tomcat-6.0.32/bin/catalina.sh start - COMPROBAR DESDE NAVEGADOR SI TOMCAT ESTÁ ARRANCADO: http://localhost:8080/

3.1.2.- Iniciar Tomcat.

Tomcat va a estar escuchando en el puerto 8080 y va a tener su propio directorio de trabajo. La misión de apache2 va a ser interceptar todas las peticiones en el puerto 80 y derivar las que considere necesarias a Tomcat; de este modo observamos la ventaja de la escalabilidad, ya que apache, al funcionar como proxy, puede tener una batería de tomcats a los que balancear las conexiones, haciendo que, si nuestras necesidades crecen, nuestras máquinas puedan ampliarse en número siendo completamente transparente para los usuarios.



Apache por defecto busca los ficheros en `/var/www`, Tomcat trabaja sobre la carpeta `/usr/local/tomcat/webapps/ROOT`. La petición de una url se puede gestionar, parte por apache y parte por Tomcat, por lo que vamos a cambiar la carpeta por defecto de trabajo para unificarlo. Para ello editamos el fichero `/usr/local/tomcat/conf/server.xml`

```
#nano /usr/local/tomcat/conf/server.xml
```

en donde encontraremos una línea con "Host name=" y lo establecemos a:

```
<Host name="localhost" appBase="/var/www"
```

Cargaremos los módulos siguientes para poder conseguir que Apache funcione como proxy:

```
# a2enmod proxy
# a2enmod proxy_ajp
```

```
# a2enmod proxy_balancer
# /etc/init.d/apache2 restart
```

ajp es un protocolo de comunicación interno y muy rápido que usa conexiones TCP persistentes. Es este protocolo el que vamos a utilizar para comunicar apache2 con Tomcat, aunque podría ser utilizado http, indicando que pregunte en el 8080. El puerto de trabajo por defecto para Tomcat es el 8009, aunque este puede ser variado desde `/usr/local/tomcat/conf/server.xml`

Modificamos el fichero de configuración del virtualhost que se pretenda utilizar, empleando el establecido por defecto.

```
# nano /etc/apache2/sites-enabled/000-default
```

en donde añadimos lo siguiente:

```
<Proxy balancer://tomcat_cluster>
Order allow,deny
Allow from all
BalancerMember ajp://localhost:8009
</Proxy>
ProxyPreserveHost On
ProxyPass /phpmyadmin/ !
ProxyPass / balancer://tomcat_cluster/
ProxyPassReverse / balancer://tomcat_cluster/
```

Pasamos a definir cada uno de los parámetros anteriores:

- ✓ **Proxy balancer://tomcat_cluster**: Estamos definiendo un cluster con nombre "Tomcat_cluster"
- ✓ **BalancerMember ajp://localhost:8009**: Se define un miembro a Tomcat_cluster, protocolo, IP y puerto.
- ✓ **ProxyPass / balancer://tomcat_cluster/**: "/" y todo lo que cuelgue de ella, se ha pasado al cluster del tomcat para que lo procese él.
- ✓ **ProxyPreserveHost On**: Mantiene la cabecera http host original, en vez de reescribirla.<7p>

Lo último es cambiar de `/etc/apache2/sites-enabled/000-default` el "**DocumentRoot**" y "**<Directory /var/www/>**" para que apunten a `/var/www/ROOT`, de esta manera podemos decidir qué parte gestiona cada aplicación desde un solo directorio.

caciones web de Java completas, es decir que su contenido:

- ✓ Servlets y JSP.
- ✓ Contenido estático: HTML, imágenes, etc.
- ✓ Otros recursos web.

Aportan como ventaja, la simplificación del despliegue de aplicaciones web, debido a que su instalación es sencilla y solamente es necesario un fichero para cada servidor en un cluster, además de incrementar la seguridad ya que no permite el acceso entre aplicaciones web distintas.

Su estructura es la siguiente:

- ✓ `/`: En la carpeta raíz del proyecto se almacenan elementos empleados en los sitios web, tipo documentos html, CSS y los elementos JSP (*.html *.jsp *.css).
- ✓ `/WEB-INF/`: Aquí se encuentran los elementos de configuración del archivo .WAR como pueden ser: la página de inicio, la ubicación de los servlets, parámetros adicionales para otros componentes. El más importante de éstos es el archivo `web.xml`.
- ✓ `/WEB-INF/classes/`: Contiene las clases Java empleadas en el archivo .WAR y, normalmente, en esta carpeta se encuentran los servlets.
- ✓ `/WEB-INF/lib/`: Contiene los archivos JAR utilizados por la aplicación y que normalmente son las clases empleadas para conectarse con la base de datos o las empleadas por librerías de JSP.

Para generar archivos `.WAR` se pueden emplear diversas herramientas desde entorno IDE "Integrated Development Environment". Por ejemplo, encontramos: NetBeans y Eclipse, ambos Open-Source y también Jbuilder de Borland, Jdeveloper de Oracle; otro modo de construir archivos `war` es mediante Ant. Se trata de una herramienta Open-Source que facilita la construcción de aplicaciones en Java. No es considerado un IDE pero para los que conocen el entorno Linux, es considerado el **make** de Java.

Un archivo .WAR

- ☐ Es un archivo comprimido que se puede generar con cualquier tipo de compresor, por ejemplo winzip, winrar, tar, etc.
- ☐ Es una aplicación web formada únicamente por archivos .html
- ☒ **Es un archivo en el cual se empaqueta en una sola unidad, aplicaciones web completas.**
- ☐ Es un archivo que engloba el protocolo de comunicación de las aplicaciones web generadas con Java.

Su nombre procede de Web Application Archive (Archivo de Aplicación Web); permiten empaquetar en una sola unidad aplicaciones web de Java completas.

4.2.- Despliegue de aplicaciones con Tomcat

Caso práctico

Una vez que la empresa BK programación dispone de un servidor de aplicaciones, Ada considera necesario formar al personal acerca de cómo desplegar aplicaciones, especificar la estructura a seguir, etc.

El empleo del servidor de aplicaciones Tomcat es una actividad bastante común y útil para empezar a desplegar aplicaciones web. María ha decidido emplear dicho servidor en su empresa y aquí nos proporciona una serie de pasos.

Una aplicación web puede ser desplegada empleando uno de los siguientes métodos:

- ✓ Por medio de archivos WAR (Web Archive).
- ✓ Editando los archivos web.xml y server.xml. Este método es el que se pasa a tratar a continuación.

Los directorios que forman una aplicación compilada suelen ser : `www`, `bin`, `src`, `tomcat` y `gwt-cache`.



La carpeta `www` contiene, a su vez, una carpeta con el nombre y ruta del proyecto que contiene los ficheros que forman la interfaz (html, js, css, etc.). La carpeta `bin` contiene las clases de java de la aplicación.

Para desplegar la aplicación en Tomcat:

1. Copiar la carpeta contenida en `www` (con el nombre del proyecto) en el directorio `webapps` de Tomcat.
2. Renombrar la nueva carpeta así creada en Tomcat con un nombre más sencillo. Esa será la carpeta de la aplicación en Tomcat.
3. Crear, dentro de dicha carpeta, otra nueva, y darle el nombre `WEB-INF` (respetando las mayúsculas).
4. Crear, dentro de `WEB-INF`, otros dos subdirectorios, llamados `lib` y `classes`.
5. Copiar en `lib` todas las librerías (.jar) que necesite la aplicación para su funcionamiento.
6. Copiar el contenido de la carpeta `bin` de la aplicación en el subdirectorio `WEB-INF/classes` de Tomcat.
7. Crear en `WEB-INF` un fichero de texto llamado `web.xml`, con las rutas de los servlets utilizados en la aplicación.
8. A la aplicación ya puede accederse en el servidor, poniendo en el navegador la ruta del fichero html de entrada, que estará ubicado en la carpeta de la aplicación en Tomcat.

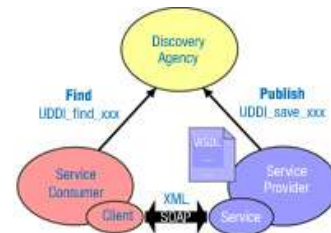
En la web que a continuación se detalla se muestran los pasos implicados en el despliegue de un servlet. Describe cómo tomar un servlet y crear una aplicación web, tanto en formato expandido como en un WAR. Ilustra cómo desplegar una aplicación web en Apache Tomcat y en WebLogic Server 6.0, un completo servidor de aplicaciones J2EE.

http://www.programacion.com/articulo/desplegar_servlets_y_aplicaciones_web_en_tomcat_y_weblogic_server_175

4.3.- Descriptor de despliegue.

Caso práctico

Una vez la empresa BK programación dispone de un servidor de aplicaciones, Ada considera necesario formar al personal acerca de cómo desplegar aplicaciones, especificar la estructura a seguir, etc. El empleo del servidor de aplicaciones Tomcat es una actividad bastante común y útil para empezar a desplegar aplicaciones web. Juan ha dedicado este punto de la wiki a explicar en qué consiste el descriptor del despliegue.



Un Descriptor de Despliegue es un documento XML que describe las características de despliegue de una aplicación, un módulo o un componente. Por esto, la información del descriptor de despliegue es declarativa, y esta puede ser cambiada sin la necesidad de modificar el código fuente.

Cualquier aplicación web tiene que aportar un descriptor de despliegue situado en `WEB-INF/web.xml`; en el caso concreto de Tomcat el descriptor `<TOMCAT_HOME>/conf/web.xml` es un descriptor por defecto que se ejecuta siempre antes del descriptor de la aplicación y, solamente, debería contener información general y no específica de la aplicación.

Un ejemplo de descriptor de despliegue puede ser el siguiente archivo web.xml:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app PUBLIC
    "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
    "http://java.sun.com/j2ee/dtds/web-app 2.2.dtd">
<web-app>
    <!-- Tus definiciones van aquí -->
</web-app>
```

Situadas entre las etiquetas `<web-app>` y `/<web-app>` estarían los descriptores de despliegue de servlets, los cuales deben contener las siguientes etiquetas en el siguiente orden:

```
<servlet>
  <servlet-name>nombre</servlet-name>
  <servlet-class>package.nombre.MiClass</servlet-class>
</servlet>
```

Para probar el servlet, una vez arrancado el servidor Tomcat, abrimos un navegador web, en el cual escribiríamos una URL con el siguiente formato:

```
http://{address}:{port}/{servletName}
```

por ejemplo:

```
http://localhost:8080/Servlet_de_prueba
```

Llegado hasta este punto, puedes realizar la siguiente **SOPA DE LETRAS** en la que podrás comprobar tus conocimientos sobre esta unidad de trabajo:

	Busca 8 conceptos relacionados con plataformas web											
Servidor web. >>	Z	G	J	M	Q	T	B	Z	A	M	I	V
Servidor web.>>	F	H	K	B	S	U	Y	Q	C	A	N	H
Plataforma web.>>	V	B	K	M	P	E	C	L	I	P	S	E
Plataforma web.>>	A	P	T	U	E	Y	I	Q	D	A	S	B
Gestor de base de datos.>>	P	E	R	L	P	W	M	X	F	C	C	A
Lenguaje de programación. >>	T	J	T	O	M	C	A	T	P	H	D	E
Lenguaje de programación. >>	A	E	R	T	Y	E	L	C	S	E	X	G
Lenguaje de programación. >>	N	C	V	B	N	Q	A	I	A	R	K	J
Lenguaje de programación. >>	P	Z	F	G	S	K	M	B	N	T	T	O
Lenguaje de programación. >>	M	T	Y	Y	I	J	P	H	P	M	R	W
Lenguaje de programación. >>	A	O	M	P	O	Z	E	R	J	A	H	B
Lenguaje de programación. >>	W	S	X	C	V	G	H	R	U	B	N	Y
Lenguaje de programación. >>	F	G	H	J	K	L	R	T	F	I	V	S

Escribe el nombre de tecnologías asociadas a aplicaciones siguiendo los enunciados				
1- La Interface Común de Entrada es uno de los estándares más antiguos en internet para trasladar la información desde una página web a un servidor web.	C	G	I	
2- Las Hojas de Estilo en Cascada se usan para formatear las páginas web.	C	S	S	
3- Las Páginas Activas se ejecutan del lado del servidor	A	S	P	
4- Este lenguaje es, como ASP, usado en el lado del servidor, es similar a ASP y puede ser usado en circunstancias similares	P	H	P	
5- Algo así como lenguaje práctico de extracción y de informes, nace con el objetivo principal de simplificar las tareas de administración de un sistema UNIX.	P	E	R	L

No hay secretos para el éxito. Éste se alcanza preparándose, trabajando arduamente y aprendiendo del fracaso.

Colin Powell (1937-..)

TEMA 2

Contenido

1.- Funcionamiento de un servidor Web.....	2
1.1.- Servicio de ficheros estáticos.....	3
1.2.- Contenido dinámico.....	4
1.3.- Protocolo HTTP y HTTPS.....	5
1.4.- Tipos MIME.....	6
1.4.1.- Configurar el servidor para enviar los tipos MIME correctos.....	8
2.- Hosts virtuales. Creación, configuración y utilización.....	10
2.1.- Virtualhosts basados en nombre.....	11
2.2.- Virtualhosts basados en IP.....	12
2.3.- Virtualhosts basados en varios servidores principales.....	13
3.- Módulos.....	14
3.1.- Operaciones sobre módulos.....	14
4.- Acceso a carpetas seguras.....	16
4.1.- Certificados digitales, AC y PKI.....	17
4.2.- Módulo ssl para apache.....	18
4.3.- Crear un servidor virtual seguro en Apache (I).....	18
4.3.1.- Crear un servidor virtual seguro en Apache (II).....	19
4.3.2.- Crear un servidor virtual seguro en Apache (III).....	20
4.4.- Comprobar el acceso seguro al servidor.....	21
5.- Autenticación y control de acceso.....	22
5.1.- Autenticar usuarios en apache mediante LDAP.....	23
6.- Monitorización del acceso: Archivos de registro (logs).....	25
6.1.- Directivas para archivos de registro.....	26
6.2.- Rotación de los archivos de registro (I).....	26
6.2.1.- Rotación de los archivos de registro (II).....	27
7.- Despliegue de aplicaciones sobre servidores Web.....	30
Anexo I - /etc/apache/sites-available/default.....	32
Anexo II - /etc/mime.types.....	33
Anexo III - /etc/apache2/sites-available/default-ssl.....	44
Anexo IV - openssl_autofirmado.txt.....	47
Anexo V - Instalación y configuración de OpenLDAP.....	48
Instalación de OpenLDAP.....	48
Configuración inicial de OpenLDAP.....	48
Asistente de configuración de slapd.....	48
Pregunta sobre la eliminación de la base de datos.....	48
Utilización LDAP versión 2.....	48
Arranque y parada manual del servidor LDAP.....	48
Anexo VI - Instalación y configuración del servidor OpenLDAP en Debian 6.....	49
Anexo VII.- Despliegue aplicación Opencart.....	51

Configuración y administración de servidores Web.

Caso práctico

A la empresa BK Programación le ha surgido un nuevo proyecto: una empresa con varias sucursales quiere montar una aplicación web por sucursal.

Ada, la directora, considera que para afrontar este proyecto y atender así la demanda ofrecida, deben configurar un nuevo equipo servidor. Para tal fin se reúne con María:

-Hola María -dijo Ada-, nos han ofrecido un nuevo proyecto relacionado con servicios web, pienso que podemos afrontarlo, pero quería saber tu opinión: ¿con la infraestructura que tenemos ahora ves necesario el montaje de otro equipo servidor dedicado a este proyecto o con lo que tenemos nos arreglamos?

-Pienso -dijo María- que tal como estamos ahora, sí o sí, independientemente de los recursos que consume este nuevo proyecto necesitamos la configuración de otro equipo servidor. Además debemos configurar dos entornos: el de pruebas y el de producción. ¿Para cuándo sería el proyecto?

-El proyecto debemos entregarlo con fecha final dentro de tres meses.

-Entonces, creo que si todo sigue su cauce normal no tendremos ningún tipo de problema para la ejecución del proyecto. ¿Qué recursos humanos habías pensado y dispones para destinar al proyecto?

-Ahora disponemos de todo el personal de la empresa y cuento contigo y con Juan para que os coordinéis las funciones de este proyecto.

-Pues por mí, no veo objeción al mismo.

-Bien -asintió Ada-, entonces no se hable más, tendremos que configurar otro equipo servidor y aceptamos el proyecto.

Así, la empresa BK Programación envió un presupuesto a la empresa del proyecto, ésta lo aprobó y comenzó el trabajo.

Para afrontar el nuevo proyecto al que se enfrenta BK Programación se acuerda en una reunión en la que asistieron: Ada, María y Juan, quien sería destinado al nuevo proyecto y las funciones a realizar en el mismo. Así, en dicha reunión se determinó que María sería la encargada del montaje, configuración y administración del nuevo equipo servidor y Juan el encargado de coordinar con el resto del personal la creación y funcionamiento de las aplicaciones web del proyecto.

María, entonces, se puso manos a la obra y determinó el siguiente escenario de trabajo para el equipo servidor de este proyecto:

- ✓ Sistema Operativo: Debian GNU/Linux 6.0
- ✓ Servidor Web: Apache (apache2)
 - Configuración de Red:
 - Servidor Web: 192.168.200.250
 - Cliente de pruebas (desde donde se lanza el navegador): 192.168.200.100

"Se debe hacer todo tan sencillo como sea posible, pero no más sencillo."

Albert Einstein

Hay que tener en cuenta que en el escenario las IP empleadas son **IP privadas**, sin existencia en Internet, por lo cual siempre que se haga referencia a las mismas a través de nombre de dominios, deberá existir un **servidor DNS** que las resuelva en local o bien en su defecto deberán existir las entradas correspondientes en el fichero del sistema local `/etc/hosts`.

1.- Funcionamiento de un servidor Web.

Caso práctico

Para poder llevar a buen fin el proyecto, María, reúne al equipo destinado al mismo, ya que quiere que todo el personal tenga claro los requisitos, entregables y fechas de ejecución del proyecto. Así, en esta reunión informativa para todo el equipo destinado al proyecto, trató los siguientes temas:

1. Recursos del equipo servidor.
2. Conectividad del equipo servidor.
3. Servidor web empleado: El porqué de su elección y funcionamiento.
4. Posibilidades del servidor web empleado.
5. Requisitos de las aplicaciones web del proyecto.
6. Entregables y fechas.

¿Alguna vez te has parado a pensar qué existe detrás de una página web? ¿Por qué al escribir un nombre en un navegador puedes visionar una página web? ¿Por qué no tienes acceso a determinadas páginas? ¿De qué modo puedes impedir el acceso a determinados sitios de una página: por directorio, por usuario? ¿Cómo se puede establecer una comunicación segura en una transición bancaria? ...

Hoy en día utilizamos Internet como una herramienta común: para el trabajo, para el ocio... Pero sin duda el elemento fundamental que usamos no es otro que el navegador, gracias al cual podemos sacar partido a todo lo que se encuentra en Internet: comprar entradas para el cine, acceder a nuestra cuenta bancaria, averiguar el tiempo que hará el fin de semana... pero nada de esto tendría sentido si detrás de cada página web a la que accedemos no existiera un servidor web, el cual permite que la página esté accesible 24x7 (24 horas al día y 7 días a la semana, es decir, siempre).

Detrás de cada página web debe existir un servidor web que ofrezca esa página, bien a los internautas, a los trabajadores de una empresa -por tratarse de una página web interna, de la empresa, no accesible a Internet-, o a todo aquel que disponga de una conexión de red con la cual pueda acceder a la página.

La configuración del servidor web dependerá de las páginas web que ofrezca, así la configuración no será la misma si la página posee contenido estático o no, o si se necesita que modifique el contenido según interacción del usuario, o si se necesita de comunicación segura en la transición de información, o si se debe tener en cuenta el control de acceso a determinados sitios de la página. Por lo tanto según las páginas web que se ofrezcan el servidor web deberá estar configurado para tal fin: con soporte PHP, con soporte de cifrado, con soporte de control de acceso, etc.



Pero ¿un servidor web pueda alojar varias páginas web o solamente una? Es más, ¿puede alojar varios sitios (*Conjunto de páginas web*), dominios de Internet (*Nombre por el cual se reconoce a un grupo de dispositivos o equipos conectados a la red. Éstos pueden ser nombres locales, no existentes en Internet, pero son mayoritariamente utilizados para su uso en Internet, por ejemplo: debian.org*) o solamente uno, esto es, permite hosts virtuales (*Dominios independientes que se pueden alojar en un mismo servidor web*)? Pues, un servidor web puede alojar varias páginas, sitios, dominios de Internet, pero hay que tener en cuenta que la elección del servidor web será muy importante para la configuración y administración de uno o múltiples sitios, ya que: ¿puede el servidor web ser modular -fácilmente se le pueden añadir o quitar características-, o por la contra si queremos añadirle una funcionalidad que no posea en la instalación base debemos desinstalarlo e instalarlo de nuevo, por ejemplo: hasta ahora el servidor web solamente ofrecía páginas estáticas pero queremos ofrecer también páginas web dinámicas, qué hacemos: modular o nueva instalación.

También tenemos que pensar que todo puede crecer y lo que ahora era un servidor web que ofrecía x número de páginas necesitamos que ofrezca $x*y$, con lo cual tenemos que prever la escalabilidad del servidor web, y también la estabilidad: ¿cómo se comporta ante múltiples conexiones simultáneas?

De nada servirá tener instalado un servidor web sin saber cómo se va a comportar ofreciendo el servicio, con lo cual será muy importante previamente y durante el funcionamiento del servidor establecer unas pruebas de funcionamiento del mismo y registrar lo acontecido.

Por todo lo anteriormente comentado veremos cómo configurar y administrar el servidor Apache (apache2), ya que soporta: páginas web estáticas, dinámicas, hosts virtuales, seguridad mediante cifrado, autenticación y control de acceso, modularización y monitorización de archivos de registro.

1.1.- Servicio de ficheros estáticos.

¿Es necesario que todas las páginas web se modifiquen constantemente? ¿Un blog sería útil si el contenido no sufre cambios? ¿Y un manual? ¿Si actualizamos un manual la página deja de ser estática?

Todas aquellas páginas web que durante el tiempo no cambian su contenido no necesariamente son estáticas. Una página estática puede modificarse, actualizando su contenido y seguir siendo estática, ¿entonces? Entonces debemos diferenciar cuando accedemos a una página web entre código ejecutable en el lado del servidor y en el lado del cliente -equipo que solicita la página mediante el cliente web (navegador)-. Si al acceder a una página web no es necesaria la intervención de código en el lado del servidor -por ejemplo código PHP- o en el lado del cliente -por ejemplo javascript- entonces entenderemos que la página es estática, si por el contrario es necesaria la intervención en el lado del servidor y/o en el lado del cliente entenderemos que la página es dinámica.

Ofrecer páginas estáticas es simple, puesto que solamente se necesita que el servidor web disponga de soporte html/xhtml/css o incluso solamente html/xhtml. En cuanto a configuración y administración del servidor es el caso más simple: solamente se necesita un soporte mínimo base de instalación del servidor Apache, esto es, no se necesita por ejemplo soporte PHP. En cuanto a rendimiento del servidor, sigue siendo el caso más beneficioso: no necesita de ejecución de código en el lado del servidor para visionar la página y tampoco necesita ejecución de código en el lado del cliente, lo que significa menos coste de CPU y memoria en el servidor y en el cliente, y por lo tanto una mayor rapidez en el acceso a la información de la página.



Para poder ofrecer páginas estáticas mediante el servidor Apache simplemente copias la página en la ruta correspondiente donde quieres que se visiona la página. Así por ejemplo cuando se instala Apache en un GNU/Linux Debian 6 se crean una serie de rutas en el equipo servidor similar a la estructura siguiente.

Rutas de interés en la instalación de Apache (apache2)

Rutas de interés en la instalación de Apache (apache2) en un GNU/Linux Debian

/etc/apache2/	/etc/apache2/sites-available/
├── apache2.conf	├── default
├── conf.d	└── default-ssl
├── envvars	
├── httpd.conf	/var/www/
├── magic	└── index.html
├── mods-available	
├── mods-enabled	/etc/apache2/mods-available/mime.conf
├── ports.conf	
├── sites-available	/etc/apache2/apache2.conf
└── sites-enabled	

En la instalación de Apache se crea una página web en `/var/www/index.html` referenciada a través del [archivo default](#) (`/etc/apache/sites-available/default`), éste contiene la configuración por defecto, generada en la instalación de Apache, para esa página. Si solamente quieres servir una página web la forma más fácil de hacerlo sería sustituyendo la página `index.html`, referenciada en `default`, por la página que quieres servir, por ejemplo `empresa.html`. Puedes comprobarlo siguiendo el procedimiento:

1. Abres el navegador en la página por defecto creada en la instalación de Apache: `index.html`.
2. Sustituyes los archivos en el servidor. Ten en cuenta que la página a servir debe siempre poseer el nombre `index.html`.
3. Pulsas F5 en el navegador para actualizar la página y la página que verás será la tuya.

Si lo que quieres es servir otra página, por ejemplo `empresa.html`, simplemente no le cambies como antes el nombre, deja el que posee la página. Ahora podrás ver dos páginas en el servidor: la página `index.html` y la página `empresa.html`. Si lo que quieres es servir más páginas pues, como antes, simplemente vas subiendo al servidor las páginas e incluso podrías organizarlas en carpetas.

Te proponemos que hagas un viaje por la página web de documentación de Apache.
<http://httpd.apache.org/docs/2.2/es/>

1.2.- Contenido dinámico.

"El progreso consiste en el cambio."

Miguel de Unamuno

Muchas veces seguro que te encuentras visitando una página web y la información te parece tan interesante que procedes y guardas en **Favoritos** la dirección URL (*dirección de Internet de un recurso válida para su posible utilización a través de Internet, la cual permite que el navegador la encuentre y la muestre de forma adecuada, por ejemplo: <http://www.debian.org>*) para una posterior visión, pero cuando de nuevo deseas ver la página resulta que lo que estás viendo no tiene nada que ver o es distinto de lo que esperabas, ¿qué ha ocurrido? Pues puede que la página haya cambiado su contenido o que la página que visitas posee contenido no estático, dinámico, dependiente del código ejecutado en el servidor o en el cliente al acceder a la página.



Imagínate que accedes a una página web y dependiendo si posees una cuenta de usuario u otra el contenido es distinto, o que presionas en una imagen de la página y se produce un efecto en la misma, o que el contenido cambia dependiendo del navegador. De cualquier forma la página ha sido modificada mediante una interacción con el usuario y/o el navegador, por lo tanto nos encontramos con una página dinámica.

Como bien puedes pensar, una página dinámica, necesita más recursos del servidor web que una página estática, ya que consume más tiempo de CPU y más memoria que una página estática. Además la configuración y administración del servidor web será más compleja: cuántos más módulos tengamos que soportar, más tendremos que configurar y actualizar. Esto también tendrá una gran repercusión en la seguridad del servidor web: cuántos más módulos más posibilidades de problemas de seguridad, así si la página web dinámica necesita, para ser ofrecida, de ejecución en el servidor debemos controlar que es lo que se ejecuta.

Algunos módulos con los que trabaja el servidor web Apache para poder soportar páginas dinámicas son: `mod_actions`, `mod_cgi`, `mod_cgid`, `mod_ext_filter`, `mod_include`, `mod_ldap`, `mod_perl`, `mod_php5`, `mod_python`.

En el siguiente enlace a la página de Apache puedes ampliar la información que te proporcionamos sobre los módulos.

<http://httpd.apache.org/docs/2.2/es/mod>

Abres el navegador y solicitas una página a un servidor web: ¿cuál de las siguientes acciones indica que la página solicitada no es dinámica?

- ☐ La página tiene un panel de control, al cual accedes mediante tu usuario y tu contraseña, los cuales nunca cambias. La página entonces establece comunicación con una base de datos y te permite el acceso a tu perfil, distinto del perfil del administrador de la página.
- ☐ Al pasar el puntero por encima de una imagen, ésta se redimensiona y al salir vuelve al tamaño original.
- ☐ Cuando visitas la página con distintos navegadores aparece un comentario de alerta indicando el navegador con el cual estás accediendo a la página.
- ☒ La página solicitada es un manual sobre el Servidor Apache, y está totalmente escrita en código HTML y CSS.

Aquellas páginas cuyo contenido no depende de la interacción del usuario, del navegador o un sistema gestor de bases de datos son páginas estáticas.

1.3.- Protocolo HTTP y HTTPS.

¿Quieres conservar la información de forma confidencial? ¿Quieres transferir información de forma segura? Si estás pensando en este tipo de preguntas necesariamente estás pensando en el protocolo HTTPS (*protocolo basado en el protocolo HTTP, destinado a la transferencia segura de datos mediante cifrado, es decir, es la versión segura de HTTP*) y no en el protocolo HTTP (*protocolo usado en cada transacción de la World Wide Web*).



El protocolo HTTPS permite que la información viaje de forma segura entre el cliente y el servidor, por la contra el protocolo HTTP envía la información en texto claro, esto es, cualquiera que accediese a la información transferida entre el cliente y el servidor puede ver el contenido exacto y textual de la información.

Para asegurar la información, el protocolo HTTPS requiere de certificados y siempre y cuando sean validados, la información será transferida cifrada. Pero cifrar la información requiere un tiempo de computación, por lo que será perjudicado el rendimiento del servidor web. Así, ¿es necesario que toda, absolutamente toda, la información sea transferida entre el cliente y servidor de forma cifrada? A lo mejor solamente es necesario que sea cifrada la autenticación a dicha información, por eso en algunas páginas web puede que el servidor esté configurado para que en todo el dominio esté cifrada su información o simplemente el intento de acceso a la misma.

Un servidor web, como Apache, puede emitir certificados, pero puede que en algún navegador sea interpretado como peligroso, esto suele ser debido a que los navegadores poseen en su configuración una lista de Entidades Certificadoras que verifican, autentican y dan validez a los certificados. ¿Tú, confiarías en un DNI que no fuese certificado por una entidad de confianza como el Ministerio del Interior? Pues, lo mismo le pasa a los navegadores, solamente confían en quien confían. Eso no quiere decir que no puedes crear tus certificados en un servidor web, de hecho muchas empresas lo hacen, sobre todo para sitios internos o externos en los que solamente puede acceder personal autorizado por la propia empresa. Ahora si, si utilizas certificados mediante Apache en un sitio visible a través de Internet y accesible por cualquier usuario, o bien eres una empresa o entidad en la que de por si confía el usuario o la imagen de la empresa o entidad quedará muy mal parada, ya que lo más probable es que el usuario no aceptará la comunicación, por visionar en el navegador un aviso de problema de seguridad.

El protocolo HTTPS utiliza cifrado sobre SSL/TLS (*protocolos criptográficos que proporcionan comunicaciones seguras por una red, comúnmente Internet*) que proporcionan autenticación y privacidad. Entonces, si necesitas que la información viaje cifrada debes emplear el protocolo HTTPS, en caso contrario el protocolo HTTP. Hay que dejar claro que la utilización del protocolo HTTPS no excluye ni impide el protocolo HTTP, los dos pueden convivir en un mismo dominio.

Bien, pero, ¿cómo funcionan? En el protocolo HTTP cuando escribes una dirección URL en el navegador, por ejemplo `http://www.debian.org/index.es.html`, antes de ver la página en el navegador existe todo un juego de protocolos, sin profundizar en todos ellos básicamente lo que ocurre es lo siguiente: se traduce el dominio DNS (*sistema de nomenclatura jerárquica para computadoras, servicios o cualquier recurso conectado a Internet o a una red privada, por ejemplo: apache.org determina un dominio org (organización) y un subdominio que identifica en este caso la máquina o conjunto de máquinas de nombre apache*) por una IP, una vez obtenida la IP se busca en ella si un servidor web aloja la página solicitada en el puerto 80 (*número utilizado en las comunicaciones cliente/servidor, en transmisiones TCP o UDP comprendido entre 1 y 65535, que indica por donde tiene lugar la conexión con un servidor. Están estandarizados, esto es, un servidor suele estar activo siempre por definición en un puerto determinado, pero éste puede que sea modificado en la configuración del servidor. Por ejemplo un servidor web espera en el puerto TCP 80*), puerto TCP (*es uno de los protocolos fundamentales en Internet. Garantiza que los datos serán entregados en su destino sin errores y una vez recogidos ponerlos en el mismo orden en que se transmitieron*) asignado por defecto al protocolo HTTP. Si el servidor web aloja la página ésta será transferida a tu navegador. Sin embargo cuando escribes en el navegador una dirección URL con llamada al protocolo HTTPS, el procedimiento es similar al anterior pero un poco más complejo, así se traduce el dominio DNS por una IP, con la IP se busca el servidor web que aloja la página solicitada en el puerto 443, puerto TCP asignado por defecto al protocolo HTTPS, pero ahora antes de transferir la página a tu navegador se inicia una negociación SSL, en la que entre otras cosas el servidor envía su certificado -el navegador aunque es poco habitual también puede enviar el suyo-. Si el certificado es firmado por un Entidad Certificadora de confianza se acepta el certificado y se cifra la comunicación con él, transfiriendo así la página web de forma cifrada.

Puedes hacer que un servidor web para una determinada página espere los protocolos HTTP y HTTPS en puertos TCP distintos del 80 y 443 respectivamente. Eso sí, cuando visites la página web a mayores en la dirección URL debes especificar el puerto TCP, por ejemplo: `http://www.tupagina.local:8080`, de esta forma el servidor web espera la petición de la página `www.tupagina.local` en el puerto 8080; del mismo modo en la dirección URL: `https://www.tupagina.local:4333` espera la petición de la página `www.tupagina.local` en el puerto 4333. Como ves, puedes configurar los puertos, pero ten en cuenta que cualquiera que quisiera acceder a esas páginas debería saber el puerto TCP de la solicitud. Entonces, quiere decir que ¿aunque no escribas el puerto TCP en las direcciones URL estas se interpretan en el puerto 80 y 443 para el protocolo HTTP y HTTPS respectivamente? Pues sí, así es. Es lo mismo escribir `http://www.tupagina.local:80` que `http://www.tupagina.local` y es lo mismo escribir `https://www.tupagina.local:443` que `https://www.tupagina.local`.

En la página <http://www.warriorsofthe.net/index.html> puedes encontrar un vídeo muy ameno sobre el funcionamiento de Internet.

1.4.- Tipos MIME.

¿Cómo se transmite un vídeo por Internet, con qué codificación? ¿Cómo sabe un navegador que al seguir un enlace de vídeo el programa que debe utilizar para reproducirlo?

El estándar Extensiones Multipropósito de Correo de Internet o MIME (Multipurpose Internet Mail Extensions), especifica como un programa debe transferir archivos de texto, imagen, audio, vídeo o cualquier archivo que no esté codificado en US-ASCII. **MIME** está especificado en seis RFC (*Request for Comments. Serie de documentos en los que se detalla prácticamente todo lo relacionado con la tecnología de la que se sirve Internet: protocolos, recomendaciones, comunicaciones...*):

RFC2045	http://tools.ietf.org/html/rfc2045
RFC 2046	http://tools.ietf.org/html/rfc2046
RFC 2047	http://tools.ietf.org/html/rfc2047
RFC 4288	http://tools.ietf.org/html/rfc4288
RFC4289	http://tools.ietf.org/html/rfc4289
RFC2077	http://tools.ietf.org/html/rfc2077

¿Cómo funciona? Imagínate el siguiente ejemplo: Transferencia de una página web.

Cuando un navegador intenta abrir un archivo el estándar MIME le permite saber con qué tipo de archivo está trabajando para que el programa asociado pueda abrirlo correctamente. Si el archivo no tiene un tipo MIME especificado el programa asociado puede suponer el tipo de archivo mediante la extensión del mismo, por ejemplo: un archivo con extensión **.txt** supone contener un archivo de texto.

Bien, pero ¿cómo lo hace?

El navegador solicita la página web y el servidor antes de transferirla confirma que la petición requerida existe y el tipo de datos que contiene. Esto último, mediante referencia al tipo MIME al que corresponde. Este diálogo, oculto al usuario, es parte de las cabeceras HTTP (*Son el lenguaje que utilizan el cliente(navegador web) y el servidor web para comunicarse entre sí. Se puede considerar cada cabecera como un mensaje aparte en el sentido de la comunicación entre el cliente y el servidor. Primero hay unas cuantas preguntas (cabeceras de solicitud), las cuales son respondidas (cabeceras de respuesta)*), protocolo que se sigue en la web.

En ese diálogo, en las cabeceras respuestas del servidor existe el campo **Content-Type**, donde el servidor avisa del tipo MIME de la página. Con esta información, el navegador sabe cómo debe presentar los datos que recibe. Por ejemplo cuando visitas <http://www.debian.org/index.es.html> puedes ver como respuesta en la cabecera del servidor el campo **Content-Type: text/html**, indicando que el contenido de la página web es tipo texto/html:

```
HTTP/1.1 200 OK
Date: Fri, 13 May 2011 18:11:36 GMT
Server: Apache
Last-Modified: Fri, 13 May 2011 16:22:52 GMT
Etag: "3a9b-4a32ab7a76f00"
Accept-Ranges: bytes
Cache-Control: max-age=86400
Expires: Sat, 14 May 2011 18:11:36 GMT
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 4864
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html
Content-Language: es
```

Cada identificador de tipo MIME consta de dos partes. La primera parte indica la categoría general a la que pertenece el archivo como, por ejemplo, **"text"**. La segunda parte del identificador detalla el tipo de archivo específico como, por ejemplo, **"html"**. Un identificador de tipo MIME **"text/html"**, por ejemplo, indica que el archivo es una página web estándar.

Los tipos MIME pueden indicarse en tres lugares distintos: el servidor web, la propia página web y el navegador.

- ✓ El servidor debe estar capacitado y habilitado para manejar diversos tipos MIME.
- ✓ En el código de la página web se referencia tipos MIME constantemente en etiquetas link, script, object, form, meta, así por ejemplo:

➔ El enlace a un archivo hoja de estilo CSS:

```
<link href="/miarchivo.css" rel="stylesheet" type="text/css">
```

➔ El enlace a un archivo código javascript:

```
<script language="JavaScript" type="text/javascript" src="scripts/mijavascript.js">
```

- ✓ Con las etiquetas meta podemos hacer que la página participe en el diálogo servidor-cliente, especificando datos MIME:

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
```

- ✓ El navegador del cliente también participa, además de estar capacitado para interpretar el concreto tipo MIME que el servidor le envía, también puede, en el diálogo previo al envío de datos, informar que tipos MIME puede aceptar la cabecera http_accept, así por ejemplo una cabecera http_accept tipo de un navegador sería:

```
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

El valor `*/*` significa que el navegador aceptará cualquier tipo MIME

Complementos del navegador Firefox para ver cabeceras HTTP/HTTPS:

Tamper Data <https://addons.mozilla.org/es-ES/firefox/addon/tamper-data/>

Live HTTP Headers <https://addons.mozilla.org/es-ES/firefox/addon/live-http-headers/>

1.4.1.- Configurar el servidor para enviar los tipos MIME correctos.

En un servidor web podemos especificar el tipo MIME por defecto para aquellos archivos que el servidor no pueda identificar automáticamente como pertenecientes a un tipo concreto, esto es, para aquellos los cuales no se resuelven según su extensión.

Para el servidor web Apache se utilizan dos directivas: `DefaultType` y `ForceType`.

- ✓ `DefaultType` asigna la cabecera Content-Type a cualquier archivo cuya MIME no pueda determinarse desde la extensión del archivo.
- ✓ `ForceType` hace que todos los ficheros cuyos nombres tengan una equivalencia con lo que se especifique sean servidos como contenido del tipo MIME que se establezca.

Ejemplos:

- ✓ `DefaultType text/plain` : Esto significa que cuando el navegador web solicita y recibe ese archivo como respuesta, desplegará el contenido como un archivo de texto.
- ✓ `DefaultType text/html` : Desplegará el contenido como un archivo HTML.
- ✓ `ForceType image/gif` : Desplegará el contenido como un archivo de imagen gif.
- ✓ `ForceType video/mp4` : Desplegará el contenido como un archivo de vídeo mp4.

En el siguiente enlace puedes encontrar más información sobre la directiva `DefaultType`.

<http://httpd.apache.org/docs/2.0/mod/core.html#defaulttype>

Puedes consultar más información en la documentación de Apache sobre directivas.

<http://httpd.apache.org/docs/2.0/mod/directives.html>

<http://httpd.apache.org/docs/2.0/mod/quickreference.html>

En el servidor web Apache existe el archivo `/etc/apache2/mods-available/mime.conf` donde encontrarás una referencia al archivo [/etc/mime.types](#), el cual contiene la lista de tipos MIME reconocidos por el servidor.

En el siguiente enlace encontrarás la lista oficial de los tipos MIME.

<http://www.iana.org/assignments/media-types/>

Abres el navegador y solicitas una página web que contiene un vídeo con la extensión .flv a un servidor web Apache: ¿cuáles de las siguientes afirmaciones son correctas teniendo en cuenta que el vídeo puede reproducirse y visualizarse sin problemas?



El servidor web no identifica el tipo MIME pero la extensión .flv es reconocida por el navegador, es por esto que el navegador asocia el programa correspondiente al vídeo y se reproduce sin problemas.



El archivo no es reconocido por el servidor web, por lo que el servidor web envía al navegador otro tipo MIME, compatible con el esperado y el vídeo se reproduce sin problemas.



Si la extensión .flv no es reconocida por el navegador ni por el servidor web es debido a que el tipo MIME es reconocido por cómo está programada la página web.



El servidor web no identifica el tipo MIME pero como el servidor web reconoce la extensión .flv modifica la programación de la página web incorporando el código necesario para la reproducción del vídeo.

El archivo se reproduce porque el tipo MIME viene especificado por el código programado en la página web o porque programa asociado supone el tipo de archivo mediante su extensión.

2.- Hosts virtuales. Creación, configuración y utilización.

Caso práctico

A la empresa BK Programación le ha surgido el siguiente proyecto: una empresa con varias sucursales quiere montar una aplicación web por sucursal. La empresa en cuestión consta de 7 sucursales. Todas ellas dedicadas a la misma línea de negocio. Así, las aplicaciones tendrán un frontal similar, pero estarán personalizadas dependiendo de la situación de la sucursal, de tal forma que los banners, logos e imágenes de cada aplicación serán monumentos locales a la zona de la sucursal.

El equipo de trabajo del proyecto está coordinado por María, ella es la encargada del montaje, creación y configuración del servidor web donde irán alojadas las aplicaciones web.

La empresa quiere que las sucursales puedan ser localizadas en Internet mediante URLs tipo:

`www.sucursal-zonaX.empresa-proyecto.com`, donde X puede variar de 1 a 7. Además quiere que si las páginas se buscan sin `www` éstas sigan viéndose, es decir, que `sucursal-zonaX.empresa-proyecto.com` se dirija a la misma página que `www.sucursal-zonaX.empresa-proyecto.com`

La empresa también desea que exista un único panel de control de usuarios, en la URL `www.empresa-proyecto.panel-de-control.com`, de tal forma que según el perfil que posea el usuario podrá ver un contenido u otro. Así, desea que los comerciales tengan la posibilidad de saber que productos y cantidades de los mismos existen en stock. Al panel de control se accede a través de un enlace configurado en cada aplicación.

María se reúne con Juan, el encargado del desarrollo de las aplicaciones web, y con Antonio, que ejerce el rol del usuario destinado a comprobar el buen funcionamiento de las aplicaciones haciendo pruebas con distintos navegadores:

—Pienso —dijo María— que la mejor forma de llevar a buen puerto el proyecto se realiza configurando hosts virtuales en el servidor web Apache y no solamente colgando las aplicaciones web en un directorio raíz común para luego, cada una, disponer de su espacio en una carpeta independiente.

—Sí, —dijo Juan—, además tenemos que tener en cuenta la seguridad del panel de control, deberíamos pensar en el protocolo HTTPS, para asegurarnos que la información vaya cifrada.

—Estoy de acuerdo —afirmó María—. Entonces, Antonio, deberás hacer las pruebas mediante HTTP y HTTPS.

—Vale, de acuerdo —dijo Antonio—.



Anteriormente hemos visto como poder alojar múltiples páginas web en el servidor web Apache, pero todas pertenecientes al mismo sitio/dominio, es decir, todas pertenecientes a **empresa.com**, entonces, ¿no se puede alojar páginas de distintos dominios en el mismo servidor web? La respuesta es que sí, si se puede, ¿cómo?, mediante la configuración de hosts virtuales o virtualhosts. Éstos básicamente lo que hacen es permitir que un mismo servidor web pueda alojar múltiples dominios, así configurando hosts virtuales podemos alojar: **empresa1.com**, **empresa2.com**, ..., **empresaN.com** en el mismo servidor web. Cada empresa tendrá su virtualhost único e independiente de las demás.

Aunque como se ha comentado anteriormente cada virtualhost es único e independiente de los demás, todo aquello que no esté incluido en la definición de cada virtualhost se heredará de la configuración principal: **apache2.conf** (`/etc/apache2/apache2.conf`), así, si quieres definir una directiva común en todos los virtualhost no debes modificar cada uno de los virtualhost introduciendo esa directiva sino que debes definir esa directiva en la configuración principal del servidor web Apache, de tal forma que todos los virtualhost heredarán esa directiva, por ejemplo en `apache2.conf` puedes encontrar la directiva `Timeout 300`, que establece la directiva `Timeout` igual a 300 segundos, esto es, indica el número de segundos antes de que se cancele un conexión por falta de respuesta.

Existen tres tipos de virtualhost: basados en nombre, basados en IP y basados en varios servidores principales.

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
```

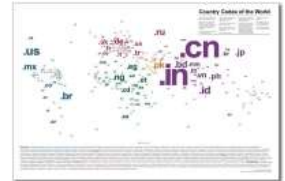
Cada campo de cada entrada puede ir separado por espacios o por tabulados.

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 resuelven esas entradas.

2.1.- Virtualhosts basados en nombre

La IP que debemos poner siempre en la definición de la directiva Virtualhost es la IP del servidor web, en nuestro escenario:

```
IP Servidor Web=192.168.200.250
```



¿Cómo lo haces? Sigues el procedimiento:

1. En la configuración de Apache2 existe un directorio `/etc/apache2/sites-available` donde se definen los virtualhosts, cada virtualhost en un fichero de texto de configuración distinto, así crea los dos ficheros siguientes en la ruta `/etc/apache2/sites-available`.
2. Fichero configuración virtualhost: `empresa1`

```
<VirtualHost IP_Servidor_Web:80>
DocumentRoot /var/www/empresa1/
ServerName www.empresa1.com.
ServerAlias empresa1.com empresa1.es www.empresa1.es
</VirtualHost>
```

3. Fichero configuración virtualhost: `empresa2`

```
<VirtualHost IP_Servidor_Web:80>
DocumentRoot /var/www/empresa2/
ServerName www.empresa2.com.
ServerAlias empresa2.com empresa2.es www.empresa2.es
</VirtualHost>
```

Explicación fichero virtualhost:

`<VirtualHost IP_Servidor_Web:80>` : Inicio etiqueta virtualhost, define la IP del servidor web donde se aloja la página de la empresa, en este caso `empresa1`. El puerto TCP para el protocolo HTTP por defecto es el 80, definido en la configuración principal del servidor, mediante la directiva **Listen**, por lo cual no es necesario ponerlo. Se pueden usar varias directivas **Listen** para especificar varias direcciones y puertos de escucha. El servidor responderá a peticiones de cualquiera de esas direcciones y puertos. Por ejemplo, para hacer que el servidor acepte conexiones en los puertos 80 y 8080, usa:

```
Listen 80
Listen 8080
```

Para hacer que el servidor acepte conexiones en dos direcciones IP y puertos diferentes, usa:

```
Listen 192.168.200.250:80
Listen 192.168.200.251:8080
```

- ✓ `DocumentRoot /var/www/empresa1/` : Definición de la ruta donde está alojada la página web en el servidor, en este caso: `/var/www/empresa1/` mediante la directiva `DocumentRoot`.
- ✓ `ServerName www.empresa1.com` : Definición del nombre DNS que buscará la página alojada en la ruta anterior del servidor mediante la directiva `ServerName`. Es el nombre que escribes en el navegador para visitar la página.
- ✓ `ServerAlias empresa1.com` : La directiva `ServerAlias` permite definir otros nombres DNS para la misma página.

- ✓ `</VirtualHost>` : Fin de la etiqueta `VirtualHost`: fin de la definición de este virtualhost para la empresa1.

Si deseas que tu servidor web ofrezca en la misma IP las URL:

`www.sucursal-zona2.empresa-proyecto.com`, `sucursal-zona2.empresa-proyecto.com`
`www.empresa-proyecto.panel-de-control.com`.

donde las 2 primeras identifican el mismo sitio web y la última otro totalmente distinto. Entonces, ¿podrías utilizar para definir los virtualhosts?

- ☐ Un solo fichero.
- ☒ Dos ficheros.
- ☐ No se pueden utilizar virtualhosts, debido a que los dominios son distintos.
- ☐ Incorrecta la pregunta ya que las URL están mal definidas, no pueden contener el carácter guión.

Si, ya que las 3 URL definen 2 sitios totalmente distintos.

2.2.- Virtualhosts basados en IP

La IP que debemos poner ahora en la definición de la directiva Virtualhost cambia, cada IP corresponde a una interfaz de red del servidor web, en nuestro escenario:

```
IP1_Servidor_Web=192.168.200.250
IP2_Servidor_Web=192.168.200.251
```



Este método no aporta ventajas sobre el anterior, es más, aún puede ser más difícil de mantener si las IP del servidor web se modifican con cierta frecuencia.

¿Cómo lo haces? Sigues el mismo procedimiento usado para los virtualhost basado en nombre, únicamente se diferencia en los ficheros a crear para los virtualhost, así:

1. En la configuración de Apache2 existe un directorio `/etc/apache2/sites-available` donde se definen los virtualhost, cada virtualhost en un fichero de texto de configuración distinto, así crea los dos ficheros siguientes en la ruta `/etc/apache2/sites-available`.

2. Fichero configuración virtualhost: `empresa3`

```
<VirtualHost IP1_Servidor_Web:80>
DocumentRoot /var/www/empresa3/
ServerName 192.168.200.250
</VirtualHost>
```

3. Fichero configuración virtualhost: `empresa4`

```
<VirtualHost IP2_Servidor_Web:80>
DocumentRoot /var/www/empresa4/
ServerName 192.168.200.251
</VirtualHost>
```

Explicación fichero virtualhost:

`<VirtualHost>`: Inicio etiqueta virtualhost, define la IP1 del servidor web donde se aloja la página de la empresa, en este caso `empresa3`. El puerto TCP por defecto es el 80, definido en la configuración principal del servidor, mediante la directiva Listen, por lo cual no es necesario ponerlo. Se pueden usar varias directivas `Listen` para especificar varias direcciones y puertos de escucha. El servidor responderá a peticiones de cualquiera de esas direcciones y puertos. Por ejemplo, para hacer que el servidor acepte conexiones en los puertos 80 y 8080, usa:

```
<VirtualHost IP1_Servidor_Web:80>
DocumentRoot /var/www/empresa3/
ServerName www.empresa3.com.
ServerAlias empresa3.com empresa3.es www.empresa3.es
</VirtualHost>
```

Para hacer que el servidor acepte conexiones en dos direcciones IP y puertos diferentes, usa:

```
<VirtualHost IP2_Servidor_Web:80>
DocumentRoot /var/www/empresa4/
ServerName www.empresa4.com.
ServerAlias empresa4.com empresa4.es www.empresa4.es
</VirtualHost>
```

- ✓ `DocumentRoot /var/www/empresa3/`: Definición de la ruta donde está alojada la página web en el servidor, en este caso: `/var/www/empresa3/` mediante la directiva `DocumentRoot`.
- ✓ `ServerName www.empresa3.com`: Definición del nombre DNS que buscará la página alojada en la ruta anterior del servidor mediante la directiva `ServerName`. Es el nombre que escribes en el navegador para visitar la página.
- ✓ `ServerAlias empresa3.com`: La directiva `ServerAlias` permite definir otros nombres DNS para la misma página.
- ✓ `</VirtualHost>`: Fin de la etiqueta VirtualHost: fin de la definición de este virtualhost para la empresa3.

En el siguiente enlace encontrarás información sobre la directiva `RewriteRule`, la cual te puede evitar tener que utilizar la directiva `ServerAlias`, pues te permite reescribir las direcciones URL.

http://httpd.apache.org/docs/2.2/mod/mod_rewrite.html#rewriterule

2.3.- Virtualhosts basados en varios servidores principales

"Cuando me preguntan cuándo estará listo un programa, contesto: depende de cuánto trabaje usted en ello."

Richard Stallman

Este método es el más complejo de todos, solo tiene sentido cuando quieras tener varios archivos de configuración **apache2.conf** independientes organizando cada uno sus propios hosts virtuales, en otro caso, mejor emplear alguno de los dos métodos anteriores.

XAMPP es una forma fácil de instalar y usar el servidor web Apache con MySQL, PHP y Perl. XAMPP, basta con descargarlo, extraerlo y comenzar.



En este momento hay cuatro versiones de XAMPP, para: Linux, Windows, Mac OS X y Solaris.

<http://www.apachefriends.org/es/xampp.html>

Te proponemos los siguientes enlaces con interesantísimos vídeos sobre la instalación y uso de XAMPP en Windows.

<http://informatica.iesanclemente.net/screencast/xampp/>

3.- Módulos.

Caso práctico

Está bien -pensó María-. Manos a la obra. Debo montar un nuevo servidor web Apache y necesito... veamos:

✓ Que varias aplicaciones web atiendan en el mismo dominio, tal que:

`sucursal-zonaX.empresa-proyecto.com`, `www.sucursal-zonaX.empresa-proyecto.com`

✓ Un único panel de control de usuarios, en la URL `www.empresa-proyecto.panel-de-control.com`,

✓ También soporte SSL para cifrado.

✓ Soporte para páginas dinámicas mediante PHP.

✓ Y soporte para control de usuarios LDAP.

Uhm..., ya lo tengo claro. Tengo que montar Apache con varios módulos, así primero instalaré Apache, luego verificaré que módulos vienen instalados por defecto, si me conviene dejarlos instalados o no, igual tengo que desinstalar alguno y tendré que investigar cuales son los módulos nuevos a instalar.

Muy bien, pues lo dicho: ¡Manos a la obra!



La importancia de un servidor web radica en su: estabilidad, disponibilidad y escalabilidad. Es muy importante poder dotar al servidor web de nuevas funcionalidades de forma sencilla, así como del mismo modo quitárselas. Es por esto que la posibilidad que nos otorga el servidor web Apache mediante sus módulos sea uno de los servidores web más manejables y potentes que existen: que necesito soporte SSL pues módulo SSL, que necesito soporte PHP pues módulo PHP, que necesito soporte LDAP pues módulo LDAP, que necesito...

En Debian, y derivados, existen dos comandos fundamentales para el funcionamiento de los módulos en el servidor web Apache: `a2enmod` y `a2dismod`.

- ✓ `a2enmod`: Utilizado para habilitar un módulo de apache. Sin ningún parámetro preguntará que módulo se desea habilitar. Los ficheros de configuración de los módulos disponibles están en `/etc/apache2/mods-available/` y al habilitarlos se crea un enlace simbólico desde `/etc/apache2/mods-enabled/`.
- ✓ `a2dismod`: Utilizado para deshabilitar un módulo de Apache. Sin ningún parámetro preguntará que módulo se desea deshabilitar. Los ficheros de configuración de los módulos disponibles están en `/etc/apache2/mods-available/` y al deshabilitarlos se elimina el enlace simbólico desde `/etc/apache2/mods-enabled/`.
- ✓ Si no dispones de esos comandos para poder habilitar y deshabilitar módulos Apache simplemente haces lo que ellos: crear los enlaces simbólicos correspondientes desde `/etc/apache2/mods-enabled/` hasta `/etc/apache2/mods-available/`.

`a2ensite` es un comando (en Debian y derivados) para habilitar configuraciones de "sitios web" en Apache2. Los ficheros de configuración de los "sitios web" disponibles (normalmente son configuraciones de hosts virtuales) están en `/etc/apache2/sites-available/` y al habilitarlos se crea un enlace simbólico desde `/etc/apache2/sites-enabled/`

Puedes consultar más información en la documentación de Apache sobre módulos.

<http://httpd.apache.org/docs/2.2/es/mod/>

La instalación o desinstalación de un módulo no implica la desinstalación de Apache o la nueva instalación de Apache perdiendo la configuración del servidor en el proceso, simplemente implica la posibilidad de poder trabajar en Apache con un nuevo módulo o no.

3.1.- Operaciones sobre módulos.

"Me lo contaron y lo olvidé. Lo vi y lo entendí. Lo hice y lo aprendí."

Confucio

Los módulos de Apache puedes instalarlos, desinstalarlos, habilitarlos o deshabilitarlos, así, puedes tener un módulo instalado pero no habilitado. Esto quiere decir que aunque instales módulos hasta que los habilites no funcionarán.



En la tabla siguiente encontrarás un resumen de operaciones, ejemplos y comandos necesarios que se le pueden realizar a los módulos:

Operaciones sobre módulos Apache en un GNU/Linux Debian

Operaciones sobre módulos Apache en un en un GNU/Linux Debian	
Instalar un módulo	Ejemplo: Instalar el módulo ssl
<code>apt-get install nombre-modulo</code>	<code>apt-get install libapache2-mod-gnutls</code>
Desinstalar un módulo	Ejemplo: Desinstalar el módulo ssl
<code>apt-get remove nombre-modulo</code>	<code>apt-get remove libapache2-mod-gnutls</code>
Habilitar un módulo	Ejemplo: Habilitar el módulo ssl
<code>a2enmod nombre-modulo-apache</code>	<code>a2enmod ssl</code>
Deshabilitar un módulo	Ejemplo: Deshabilitar el módulo ssl
<code>a2dismod nombre-modulo-apache</code>	<code>a2dismod ssl</code>

Para habilitar un módulo Apache, en Debian, también puedes ejecutar el comando `a2enmod` sin parámetros. La ejecución de este comando ofrecerá una lista de módulos a habilitar, escribes el módulo en cuestión y el módulo se habilitará. Del mismo modo para deshabilitar un módulo Apache, en Debian, puedes ejecutar el comando `a2dismod` sin parámetros. La ejecución de este comando ofrecerá una lista de módulos a deshabilitar, escribes el módulo en cuestión y el módulo se deshabilitará.

Una vez habilitado o deshabilitado los módulos Apache sólo reconocerá estos cambios cuando recargues su configuración, con lo cual debes ejecutar el comando: `/etc/init.d/apache2 restart`

Si la configuración es correcta y no quieres reiniciar Apache puedes recargar la configuración mediante el comando: `/etc/init.d/apache2 reload`.

Si no dispones de los comandos `a2enmod` y `a2dismod` puedes habilitar y deshabilitar módulos Apache creando los enlaces simbólicos correspondientes desde `/etc/apache2/mods-enabled/` hasta `/etc/apache2/mods-available/`, por ejemplo si quisieras habilitar el módulo ssl:

1. Te sitúas en el directorio `/etc/apache2/mods-available`.
`cd /etc/apache2/mods-available`
2. Verificas que el módulo aparece en esta ruta y por lo tanto está instalado
`ls ssl.*`

Este comando debe listar dos ficheros: `ssl.conf` (la configuración genérica del módulo) y `ssl.load` (la librería que contiene el módulo a cargar)

3. Creas el enlace simbólico para habilitar el módulo:

```
ln -s /etc/apache2/mods-available/ssl.load /etc/apache2/mods-enabled/ssl.load
ln -s /etc/apache2/mods-available/ssl.conf /etc/apache2/mods-enabled/ssl.conf
```

Estos comandos crean los enlaces `/etc/apache2/mods-enabled/ssl.conf` y `/etc/apache2/mods-enabled/ssl.load` que apuntan a los ficheros `/etc/apache2/mods-available/ssl.conf` y `/etc/apache2/mods-available/ssl.load` respectivamente.

4. Recargas la configuración de Apache:

```
/etc/init.d/apache2 restart
```

5. El módulo ssl ya está habilitado.

Y si quisieras deshabilitarlo, simplemente eliminas en `/etc/apache2/mods-enabled` los enlaces simbólicos creados, así si quisieras deshabilitar el módulo `ssl` ejecutarías el siguiente comando:

```
rm -f /etc/apache2/mods-enabled/ssl.*
```

Por último, no te olvides recargar la configuración de Apache: `/etc/init.d/apache2 restart`

4.- Acceso a carpetas seguras.

Caso práctico

En el transcurso del proyecto sobre aplicaciones web de varias sucursales para una empresa en las oficinas de la empresa BK Programación tuvo lugar la siguiente charla:

Bien, -le dijo Ana a Ada-, ya tenemos casi configurado el servidor web Apache.

- ¿Entonces?- preguntó Ada-

-Nos falta la configuración de la navegación de forma segura, para que la comunicación viaje cifrada.

-¿Os llevará mucho tiempo?

-Bueno...



"Depende qué tan hombre eres."

Miguel de Icaza

¿Todas las páginas web que están alojadas en un sitio deben ser accesibles por cualquier usuario?
¿Todas las accesibles deben enviar la información sin cifrar, en texto claro? ¿Es necesario que todo el trasiego de información navegador-servidor viaje cifrado?

Existe la posibilidad de asegurar la información sensible que viaja entre el navegador y el servidor, pero esto repercutirá en un mayor consumo de recursos del servidor, puesto que asegurar la información implica en que ésta debe ser cifrada, lo que significa computación algorítmica.

El cifrado al que nos referimos es el cifrado de clave pública o asimétrico: **clave pública (kpub)** y **clave privada (kpriv)**. La **kpub** interesa publicarla para que llegue a ser conocida por cualquiera, la **kpriv** no interesa que nadie la posea, solo el propietario de la misma. Ambas son necesarias para que la comunicación sea posible, una sin la otra no tiene sentido, así una información cifrada mediante la **kpub** solamente puede ser descifrada mediante la **kpriv** y una información cifrada mediante la **kpriv** solo puede ser descifrada mediante la **kpub**.

<http://www.criptored.upm.es/intypedia/video.php?id=criptografia-asimetrica&lang=es>

En el cifrado asimétrico podemos estar hablando de individuos o de máquinas, en nuestro caso hablamos de máquinas y de flujo de información entre el **navegador (A)** y el **servidor web (B)**. Ver la siguiente tabla como ejemplo de funcionamiento del cifrado asimétrico:

Funcionamiento del cifrado asimétrico.



$A(\text{inf}) \rightarrow \text{inf cifrada} \rightarrow B [\text{descifrar inf}] \rightarrow B(\text{inf}) = A(\text{inf})$

$A(\text{inf}) \rightarrow \text{inf cifrada} = [(\text{inf})]k_{\text{pub}B} \rightarrow B [\text{inf. cifrada}]k_{\text{priv}B} \rightarrow B(\text{inf}) = A(\text{inf})$

Identificación

A	Navegador web.
inf cifrada = $[(\text{inf})]k_{\text{pub}B}$	Información cifrada mediante la clave pública de B obtenida a través de un certificado digital.
$[\text{inf. cifrada}]k_{\text{priv}B}$	Información descifrada mediante la clave privada de B.
B	Servidor web.

Como ves, **A** envía la información cifrada mediante la **kpubB** y **B** la descifra mediante su clave privada(**kprivB**), por lo que se garantiza la confidencialidad de la información. Pero, ¿estás seguro

que B es quién dice que es? ¿Es quién debe ser? ¿Cómo garantizas la autenticidad de B? Pues ya que supones que B es quien dice ser mediante un certificado digital, debes confiar en ese certificado, así ¿quién emite certificados digitales de confianza? Igual que el DNI es emitido por una entidad certificadora de confianza, el Ministerio del Interior, en Internet existen autoridades de certificación (CA ó AC) que aseguran la autenticidad del certificado digital, y así la autenticidad de B, como: [VeriSign](#) y [Thawte](#). Pero, como ya hemos comentado el Servidor Web Apache permite ser CA, por lo que tienes la posibilidad de crear tus propios certificados digitales, ahora bien, ¿el navegador web(A) confiará en estos certificados? Pues, en principio no, por lo que los navegadores avisarán que la página a la cuál intentas acceder en el servidor web representa un peligro de seguridad, ya que no existe en su lista de autoridades certificadoras de confianza. En determinados casos, por imagen, puede ser un problema, pero si la empresa posee una entidad de importancia reconocida o el sitio es privado y no público en Internet o sabes el riesgo que corres puedes aceptar la comunicación y el flujo de información viajará cifrado.

4.1.- Certificados digitales, AC y PKI.

Un certificado digital es un documento electrónico que asocia una clave pública con la identidad de su propietario, individuo o máquina, por ejemplo un servidor web, y es emitido por autoridades en las que pueden confiar los usuarios. Éstas certifican el documento de asociación entre clave pública e identidad de un individuo o máquina (servidor web) firmando dicho documento con su clave privada, esto es, mediante firma digital.

La idea consiste en que los **dos extremos de una comunicación, por ejemplo cliente (navegador web) y servidor (servidor web Apache)** puedan confiar directamente entre sí, si ambos tienen relación con una tercera parte, que da fe de la fiabilidad de los dos, aunque en la práctica te suele interesar solamente la fiabilidad del servidor, para saber que te conectas con el servidor que quieres y no con otro servidor *-supuestamente cuando tú te conectas con el navegador al servidor eres tú y no otra persona la que establece la conexión-*. Así la necesidad de una **Tercera Parte Confiable (TPC ó TTP, Trusted Third Party)** es fundamental en cualquier entorno de clave pública. La forma en que esa tercera parte avalará que el certificado es de fiar es mediante su firma digital sobre el certificado. Por tanto, podremos confiar en cualquier certificado digital firmado por una tercera parte en la que confiamos. La **TPC** que se encarga de la firma digital de los certificados de los usuarios de un entorno de clave pública se conoce con el nombre de **Autoridad de Certificación (AC)**.

El modelo de confianza basado en **Terceras Partes Confiables** es la base de la definición de las **Infraestructuras de Clave Pública (ICP o PKIs, Public Key Infrastructures)**. Una Infraestructura de Clave Pública es un conjunto de protocolos, servicios y estándares que soportan aplicaciones basadas en criptografía de clave pública.

Algunos de los servicios ofrecidos por una **ICP (PKI)** son los siguientes:

- ✓ Registro de claves: emisión de un nuevo certificado para una clave pública.
- ✓ Revocación de certificados: cancelación de un certificado previamente emitido.
- ✓ Selección de claves: publicación de la clave pública de los usuarios.
- ✓ Evaluación de la confianza: determinación sobre si un certificado es válido y qué operaciones están permitidas para dicho certificado.
- ✓ Recuperación de claves: posibilidad de recuperar las claves de un usuario.

Las **ICP (PKI)** están compuestas por:

- ✓ **Autoridad de Certificación (AC)**: realiza la firma de los certificados con su clave privada y gestiona la lista de certificados revocados.
- ✓ **Autoridad de Registro (AR)**: es la interfaz hacia el mundo exterior. Recibe las solicitudes de los certificados y revocaciones, comprueba los datos de los sujetos que hacen las peticiones y traslada los certificados y revocaciones a la **AC** para que los firme.

Existen varios formatos para certificados digitales, pero los más comúnmente empleados se rigen por el estándar UIT-T (es la organización de las Naciones Unidas para las tecnologías de la información y la comunicación. En su calidad de coordinador mundial de gobiernos y sector privado, la función de la UIT abarca tres sectores fundamentales, a saber: radiocomunicaciones, normalización y desarrollo) [X.509](#). El certificado X.509 contiene los siguientes campos: versión, nº de serie del certificado, identificador del algoritmo de firmado, nombre del emisor, periodo de validez, nombre del sujeto, información de clave pública del sujeto, identificador único del emisor, identificador único del sujeto y extensiones.



4.2.- Módulo ssl para apache.

Todos los días los bancos efectúan transferencias bancarias, así como también aceptan conexiones a sus páginas web para ofrecer su servicio online. ¿Qué pasaría si cualquiera pudiese interceptar una comunicación bancaria de ese tipo? ¿Sería interesante cifrar la información efectuada antes y durante la conexión bancaria?



El método de cifrado SSL/TLS utiliza un método de cifrado de clave pública (cifrado asimétrico) para la autenticación del servidor.

El **módulo ssl** es quien permite cifrar la información entre navegador y servidor web. En la instalación por defecto éste módulo no viene activado, así que debes ejecutar el siguiente comando para poder activarlo: `a2enmod ssl`

Este módulo proporciona SSL v2/v3 y TLS v1 para el Servidor Apache HTTP; y se basa en Open SSL (Paquete de herramientas de administración y bibliotecas relacionadas con la criptografía, que suministran funciones criptográficas, entre otros, a navegadores web para acceso seguro a sitios mediante el protocolo HTTPS) para proporcionar el motor de la criptografía.

En el siguiente enlace puedes encontrar más información sobre el módulo ssl
http://httpd.apache.org/docs/2.2/es/mod/mod_ssl.html

¿Cómo harías, en Debian 6, para deshabilitar el módulo ssl de Apache?

Mediante el comando `a2dismod ssl` y recargando el servicio Apache: `/etc/init.d/apache2 reload` ó `/etc/init.d/apache2 restart`.

Y ¿cómo lo harías si no dispones del comando para Debian?

Quitando los enlaces existentes en `mods-enabled` que apunten a los archivos ssl correspondientes de la carpeta `mods-available`, por ejemplo en un Debian 6 eliminarías los archivos: `ssl.conf` y `ssl.load` situados en `/etc/apache2/mods-enabled/` y recargando el servicio Apache: `/etc/init.d/apache2 reload` ó `/etc/init.d/apache2 restart`

4.3.- Crear un servidor virtual seguro en Apache (I).

En Debian, Apache posee por defecto en su instalación el fichero `/etc/apache2/sites-available/default-ssl`, que contiene la configuración por defecto de SSL. En su contenido podemos ver las siguientes líneas:

```
SSLEngine on
SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
```

donde,

SSLEngine on : Activa o desactiva SSL

SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem : Certificado digital del propio servidor Apache

SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key : Clave privada del servidor Apache.

Esas líneas lo que quieren decir es que Apache permite conexiones SSL y posee un certificado digital autofirmado por sí mismo -ya que Apache actúa como entidad certificadora-.

Cuando activaste el módulo ssl, mediante el comando `a2enmod ssl` permitiste que Apache atienda el protocolo SSL. Así, si ahora lanzas el navegador **Firefox** con la dirección de tu servidor web Apache mediante el protocolo HTTPS, verás una imagen similar a la siguiente:



Lo que indica que el certificado digital del servidor no viene firmado por una **AC** contenida en la lista que posee el navegador, sino por el mismo Apache. Si lo compruebas haciendo clic en **Detalles Técnicos** verás algo similar a:

192.168.200.250 usa un certificado de seguridad no válido.

No se confía en el certificado porque está autofirmado.
El certificado sólo es válido para debian-servidor-fp.

(Código de error: `sec_error_untrusted_issuer`)

Ahora tienes dos opciones: Confiar en el certificado o no.

- ✓ Si confías haces clic en **Entiendo los riesgos y Añadir excepción...**
Una vez que confías puedes, antes de **Confirmar excepción de seguridad**, ver el contenido del certificado. Si estás de acuerdo la comunicación se establece y la información viaja cifrada.
- ✓ Si no confías haces clic en **¡Sácame de aquí!**

¿Pero...? Como eres AC puedes firmar certificados e incluso puedes generar también tu propio certificado autofirmado similar al que viene por defecto en Apache.

Hay que tener en cuenta que la negociación SSL es dependiente totalmente de la IP, no del nombre del sitio web, así no puedes servir distintos certificados en una misma IP.

4.3.1.- Crear un servidor virtual seguro en Apache (II).

En una distribución Debian 6 el procedimiento para generar un certificado digital sería el siguiente:

1. Instalación del paquete `openssl`

```
apt-get install openssl
```

2. Crear un certificado autofirmado para el servidor web.

```
mkdir /etc/apache2/tus-ssl/  
make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/apache2/tus-ssl/apache.pem
```

Cuando se solicite el nombre del servidor HTTP indicar el nombre DNS que corresponda a la IP del certificado, por ejemplo: `autofirmado.ssl.empresa-proyecto.com`

El nombre de dominio `autofirmado.ssl.empresa-proyecto.com` debe resolverse a una IP mediante un servidor DNS o en su defecto mediante el fichero `/etc/hosts`.

El fichero generado `/etc/apache2/tus-ssl/apache.pem` contiene tanto el certificado del servidor como la clave privada asociada al mismo.

El comando, de Debian, `make-ssl-cert` permite generar certificados autofirmados para pruebas. Los datos de configuración del certificado a generar se indican en `/usr/share/ssl-cert/ssleay.cnf`. Internamente hace uso de las utilidades de la librería `openssl`.

3. Editar la configuración SSL por defecto en el archivo `/etc/apache2/sites-available/default-ssl` para indicar el certificado del servidor y su respectiva clave privada asignando los siguientes valores a los parámetros :

```
SSLEngine on
SSLCertificateFile /etc/apache2/tus-ssl/apache.pem
SSLCertificateKeyFile /etc/apache2/tus-ssl/apache.pem
```

4. Asegúrate que el fichero `/etc/apache2/ports.conf` incluya el valor `Listen 443`
5. Habilita el soporte SSL en Apache y habilita la configuración SSL por defecto:

```
a2enmod ssl
a2ensite default-ssl
/etc/init.d/apache2 restart
```

En el equipo cliente, **192.168.200.100**, lanzas el navegador:

1. Indicar `https://autofirmado.ssl.empresa-proyecto.com` en la barra de direcciones.
2. Dará un aviso de que la AC que firma el certificado del servidor no está reconocida. Añadir la correspondiente excepción de seguridad y permitir la descarga y aceptación del certificado. Antes de aceptarlo puedes ver el contenido del certificado:



4.3.2.- Crear un servidor virtual seguro en Apache (III).

De forma genérica, por si no posees el comando `make-ssl-cert`, puedes emplear el comando `openssl` para generar los certificados.

Por ejemplo:

1. Instalación del paquete `openssl`:
`apt-get install openssl`
2. Genera el certificado y la clave privada de tu autoridad de certificación (AC)
`mkdir /etc/apache2/tus-ssl/`
`cd /etc/apache2/tus-ssl/`



Puedes ver la ejecución del los dos comandos que se utilizan en el anexo `openssl_autofirmado.txt`

```
openssl req -new -nodes -keyout tupaginaweb.key -out tupaginaweb.csr
```

Este comando genera dos archivos:

- ✓ La clave privada con el que firmarás tus futuros certificados: `tupaginaweb.key`
- ✓ El certificado con la clave pública de la AC: `tupaginaweb.csr`

Este comando pedirá algunos datos: nombre de empresa, país, contraseña... La contraseña, puedes omitirla, pero por seguridad es conveniente crearla para utilizarla cuando firmes un certificado SSL .

3. Autofirma el certificado. Puedes hacerlo porque eres una AC, de tal forma que el primer certificado que firmas es el de tu propia AC.

```
openssl x509 -in tupaginaweb.csr -out tupaginaweb.crt -req -signkey tupaginaweb.key -days 3650
```

El campo `days 3650` significa que el certificado de tu AC tardará 10 años en caducar.

4. Editar la configuración SSL por defecto en el archivo `/etc/apache2/sites-available/default-ssl` para indicar el certificado del servidor y su respectiva clave privada asignando los siguientes valores a los parámetros :

```
SSLEngine on
SSLCertificateFile /etc/apache2/tus-ssl/tupaginaweb.crt
SSLCertificateKeyFile /etc/apache2/tus-ssl/tupaginaweb.key
```

5. Asegúrate que el fichero `/etc/apache2/ports.conf` incluya el valor `Listen 443`

6. Habilita el módulo ssl y la configuración SSL por defecto.

```
a2enmod ssl
a2ensite default-ssl
/etc/init.d/apache2 restart
```

En el archivo `/usr/share/doc/apache2.2-common/README.Debian.gz` encontrarás información sobre como configurar SSL y crear certificados autofirmados.

4.4.- Comprobar el acceso seguro al servidor.

"La manera de estar seguro es no sentirse nunca seguro."

Proverbio español

A continuación una serie de actuaciones que te servirán para comprobar que el acceso seguro que estableces con el servidor seguro es el esperado:

- ✓ Siempre que te conectes mediante SSL a una página web y el certificado no sea admitido, debes ver los campos descriptivos del certificado antes de generar la excepción que te permita visitar la página.
 - ✓ Debes comprobar en el certificado si la página a la que intentas acceder es la misma que dice el certificado.
 - ✓ Típicamente en los navegadores, si no está configurado lo contrario, cuando accedes mediante cifrado SSL a una página web puedes ver en algún lugar del mismo un icono: un candado, por lo cual debes verificar su existencia para asegurarte que estás accediendo por https.
- Incluso si el certificado pertenece a alguna AC que el navegador posee en su lista de AC puedes ver en la barra de direcciones indicaciones del tipo de certificado con el que se cifra la comunicación.
- ✓ Revisar la lista de certificados admitidos que posee tu navegador. En **Firefox**, versión > 3.x , donde x es el número de revisión de la versión 3, puedes verlas dirigiéndote por las pestañas a:
`Editar → Preferencias → Avanzado → Cifrado → Ver certificados`
- Revisar la lista de revocaciones que posee tu navegador. En **Firefox**, versión > 3.x , donde x es el número de revisión de la versión 3, puedes verlas dirigiéndote por las pestañas a:

`Editar → Preferencias → Avanzado → Cifrado → Listas de revocación`

Puedes **Importar/Exportar** certificados en los navegadores, con lo cual los puedes llevar a cualquier máquina. Esto es muy útil cuando necesitas un certificado personal en máquinas distintas.



En el siguiente enlace encontrarás información muy interesante, amena y explicativa sobre la seguridad de la información y el cifrado.

<http://www.criptored.upm.es/intypedia/index.php?lang=es>

5.- Autenticación y control de acceso.

Caso práctico

La reunión tuvo lugar.

El equipo de BK Programación destinado al proyecto de aplicaciones web para varias sucursales de una empresa llegó a un acuerdo para la autenticación y el control de acceso sobre la aplicación de panel de control. Se barajaron varias alternativas: usuarios del sistema, ficheros de usuarios, base de datos SQL y LDAP. Al final se decantaron por dos opciones: ficheros de usuarios para el estado de pruebas y LDAP para la aplicación definitiva, con lo cual establecieron el siguiente protocolo de actuación:

- ✓ En la aplicación de desarrollo montada por María se realizarán las pruebas, siendo los encargados de las mismas Antonio y Carlos.
- ✓ El diseño web de la aplicación recaerá en Ana: banners, logos ...
- ✓ Juan se dedicará a la programación del panel de control: autenticación por medio de LDAP
- ✓ La encargada de montar el servicio LDAP, integrarlo en Apache y conseguir el control de acceso fue María.

Ante la espera que María instale y configure Apache con LDAP, y con ello imposibilidad de probar la autenticación por LDAP, María crea un fichero de usuarios para autenticarse en la aplicación y todos empiezan a trabajar en el resto de las cosas.

Puede que interese impedir el acceso a determinadas páginas ofrecidas por el servidor web, así: ¿crees que a una empresa le interesaría que cualquiera tuviera acceso a determinada información confidencial?, o puede que interese controlar el acceso hacia un servicio a través de la web, como el correo electrónico. Para este tipo de casos tenemos que pensar en la autenticación y el control de acceso.



Cuando nos autenticamos en una web suele transferirse la información de autenticación a una base de datos, que puede existir en la misma máquina que el servidor web o en otra totalmente diferente. Suelen emplearse bases de datos SQL o LDAP para la autenticación de usuarios, siendo OpenLDAP (<http://www.openldap.org/>) una de las alternativas más empleadas.

Puedes visitar el enlace de wikipedia [AAA](http://es.wikipedia.org/wiki/Protocolo_AAA) (Autenticación, Autorización y Registro. Conjunto de herramientas, procedimientos y protocolos que garantizan un tratamiento coherente de las tareas de autenticación, autorización y registro de actividad de las entidades que tienen acceso a un sistema de información) donde encontrarás más información referente a la autenticación:

http://es.wikipedia.org/wiki/Protocolo_AAA

HTTP proporciona un método de autenticación básico de usuarios: **basic**. Este método ante una petición del cliente (navegador web) al servidor cuando se solicita una URL mostrará un diálogo pidiendo usuario y contraseña. Una vez autenticado el usuario, el cliente volverá a hacer la petición al servidor pero ahora enviando el usuario y contraseña, en texto claro (sin cifrar) proporcionados en el diálogo. Es recomendable entonces si empleas este método que lo hagas combinado con conexión SSL (HTTPS).

En la autenticación HTTP Basic es muy típico utilizar archivos .htaccess en los directorios que queremos controlar el acceso. Puedes encontrar un ejemplo sobre basic con https en el archivo

virtualhost-ssl-basic:

```
<IfModule mod_ssl.c>
#<VirtualHost default :443>
<VirtualHost 192.168.200.250:443>
    ServerAdmin web-autenticacion@empresa-proyecto.com
    ServerName web-con-autenticacion-basic.empresa-proyecto.com
    DocumentRoot /var/www/web-con-autenticacion-basic
    <Directory /var/www/web-con-autenticacion-basic/>
        AllowOverride AuthConfig
```

```
Options Indexes FollowSymLinks MultiViews
Order allow,deny
allow from all
</Directory>
ErrorLog ${APACHE_LOG_DIR}/error-web-autenticacion-basic.log
LogLevel warn
CustomLog ${APACHE_LOG_DIR}/ssl_access-web-autenticacion-basic.log combined
SSLEngine on
SSLCertificateFile /etc/apache2/tus-ssl/tupaginaweb.crt
SSLCertificateKeyFile /etc/apache2/tus-ssl/tupaginaweb.key
</VirtualHost>
</IfModule>
```

y un ejemplo sobre `.htaccess` en el archivo `htaccess` :

```
AuthType Basic
AuthName "Web con Autenticacion Basic"
AuthUserFile /etc/apache2/web.htpasswd
##Require valid-user
Require user user1
```

Para usar archivos `.htaccess`, necesitas tener una configuración en el servidor que permita poner directivas de autenticación en estos archivos, mediante la directiva `AllowOverride`, así:

```
AllowOverride AuthConfig
```

Puedes visitar el siguiente enlace donde encontrarás más información referente a la autenticación http basic:

<http://httpd.apache.org/docs/2.0/es/howto/auth.html>

También se puede controlar el acceso mediante IP. Puedes encontrar un ejemplo en el archivo `virtualhost-control-por-IP`:

```
<VirtualHost IP_Servidor_Web:80>
Alias /carpeta-controlada "/usr/srv/control/carpeta-controlada/"
<Directory "/usr/srv/control/carpeta-controlada/">
    Order deny,allow
    Deny from all
    Allow from IP permiso concedido
</Directory>
DocumentRoot /usr/srv/control/carpeta-controlada
ServerName www.empresa.com.
ServerAlias empresa.com
</VirtualHost>
```

5.1.- Autenticar usuarios en apache mediante LDAP.

Se ha comentado en el apartado anterior que el servidor web Apache permite la autenticación de usuarios mediante LDAP. Esto es posible mediante los módulos `ldap` y `authnz_ldap`.

En este anexo se encuentra cómo instalar y configurar un servidor OpenLDAP pero también deberías visitar la siguiente web

http://www.ite.educacion.es/formacion/materiales/85/cd/linux/m6/instalacin_y_configuracion_de_openldap.html

Para una instalación de OpenLDAP en Debian 6 visita el enlace al siguiente documento:

Instalación y configuración del servidor OpenLDAP en Debian 6

En el siguiente enlace encontrarás más información sobre la autenticación LDAP para el servidor web Apache mediante el módulo `authnz_ldap`.

http://httpd.apache.org/docs/2.2/mod/mod_authnz_ldap.html

Para el buen funcionamiento de lo expuesto a continuación se asume que tanto Apache2 como OpenLDAP están instalados y configurados:

1. Habilita el soporte LDAP para Apache2:

```
a2enmod authnz_ldap
/etc/init.d/apache2 restart
```

2. Configura el virtualhost `autenticacion-ldap-apache` como sigue:

```
<VirtualHost *:80>
  DocumentRoot /var/www/autenticacion-ldap
  ServerName www.empresa-proyecto.panel-de-control.com
  ServerAlias www.autenticacion-ldap.empresa-proyecto.com
  <Directory /var/www/autenticacion-ldap>
    AllowOverride All
  </Directory>
  ErrorLog /var/log/apache2/error-autenticacion-ldap.log
  LogLevel warn
  CustomLog /var/log/apache2/access-autenticacion-ldap.log combined
</VirtualHost>
```

La directiva `AllowOverride All` es necesaria para habilitar ficheros `.htaccess`

3. Crea el fichero `/var/www/autenticacion-ldap/.htaccess` que permite configurar la autenticación ldap para el virtualhost anterior:

```
AuthName "Autenticacion por LDAP"
AuthType Basic
AuthBasicProvider ldap
AuthzLDAPAuthoritative on
AuthLDAPUrl ldap://127.0.0.1/ou=usuarios,dc=proyecto,dc=com?uid
Require ldap-user user1LDAP
```

La directiva `Require ldap-user admin` permite la autenticación al usuario `user1LDAP`, todos los demás usuarios tienen el acceso denegado.

4. Accede a la URL: `www.empresa-proyecto.panel-de-control.com` ó `www.autenticacion-ldap.empresa-proyecto.com`



6.- Monitorización del acceso: Archivos de registro (logs).

Caso práctico

¿Qué, quién, dónde, cuándo, por qué ha pasado? Eso es lo que queremos saber en todo momento - comentó María-. Recordad que es necesario guardar los archivos de registro al menos durante 1 año según la LSSI/CE. Tenemos que estar preparados ante cualquier petición de los logs (requerimiento judicial) por parte de las administraciones. Es por esto que tú, Antonio, vas a realizar una batería de pruebas: accesos a páginas existentes y no existentes, búsqueda de listado de ficheros y no solamente el index.html, accesos no permitidos a bases de datos, accesos controlados por IP, por usuario, etc.

Muy bien, eso está hecho -dijo Antonio-.

Tan importante como es configurar un servidor web lo es mantener y comprobar su correcto funcionamiento, y para ello debes ayudarte de los logs o archivos de registro que te permiten revisar y estudiar su funcionamiento

Apache permite mediante diversas directivas crear archivos de registro que guardarán la información correspondiente a las conexiones con el servidor. Esta información es guardada en formato CLF (**Common Logon Format**) por defecto. Ésta es una especificación utilizada por los servidores web para hacer que el análisis de registro entre servidores sea mucho más sencillo, de tal forma que independientemente del servidor web utilizado podamos emplear el mismo método de análisis de registro, ya sea mediante lectura, mediante programas ejecutables (scripts) o mediante programas propios de análisis de registro.

En un archivo de registro en formato CLF cada línea identifica una solicitud al servidor web. Esta línea contiene varios campos separados con espacios. Cada campo sin valor es identificado con un guión (-). Los campos empleados en una configuración por defecto de Apache2 son los definidos en la siguiente tabla:

Ejemplo log Apache en formato CLF		
192.168.200.100 - - [05/May/2011:17:19:18 +0200] "GET /index.html HTTP/1.1" 200 20		
Campos (especificadores)	Definición	Ejemplo
host (%h)	Identifica el equipo cliente que solicita la información en el navegador.	192.168.200.100
ident (%l)	Información del cliente cuando la máquina de éste ejecuta <code>identd</code> y la directiva <code>IdentityCheck</code> está activada.	
authuser (%u)	Nombre de usuario en caso que la URL solicitada requiera autenticación HTTP.	
date (%t)	Fecha y hora en el que se produce la solicitud al servidor. Va encerrado entre corchetes. Este campo tiene su propio formato: <code>[día/mes/año:hora:minuto:segundo zona]</code>	[05/May/2011:17:19:18 +0200]
request (%r)	Petición del cliente, esto es, la página web que está solicitando. En el ejemplo: <code>/index.html</code> , esto es, dentro de la raíz del dominio que se visite la página	/index.html
status (%s ó %>s)	Identifica el código de estado HTTP de tres dígitos que se devuelve al cliente.	200
Bytes (%b)	Sin tener en cuenta las cabeceras HTTP el número de bytes devueltos al cliente.	20

Cada campo tiene su especificador, el cual se emplea en las directivas de Apache para indicar que campo queremos registrar.

6.1.- Directivas para archivos de registro.

El contexto de aplicación de todas las directivas que se indican a continuación en la siguiente tabla puede ser el de la configuración principal del servidor así como el de la configuración de los host virtuales.

Directivas para archivos de registro.	
Directivas	Definición
<code>TransferLog</code>	Directiva que define el nombre del archivo de registro o al programa al que se envía la información de registro. Emplea los especificadores asignados por la directiva <code>LogFormat</code> .
<code>LogFormat</code>	Directiva que define el formato del archivo de registro asignado con la directiva <code>TransferLog</code>
<code>ErrorLog</code>	Directiva que permite registrar todos los errores que encuentre Apache. Permite guardar la información en un archivo de registro o bien en <code>syslog</code>
<code>CustomLog</code>	Directiva similar a la directiva <code>TransferLog</code> , pero con la particularidad que permite personalizar el formato de registro empleando los especificadores anteriormente vistos.
<code>CookieLog</code>	Directiva que define el nombre del archivo de registro donde registrar información sobre cookies

La tabla siguiente muestra la sintaxis y el uso de las anteriores directivas:

Sintaxis y uso de directivas para archivos de registro	
Directiva TransferLog	
Sintaxis	<code>TransferLog nombre_fichero_archivo_registro tubería_para_enviar_al_programa_la_información_de_registro</code>
Uso	<code>TransferLog logs/acceso_a_empresal.log</code>
Directiva LogFormat	
Sintaxis	<code>LogFormat nombre_fichero_archivo_registro [opcional_alias] [opcional_alias] permite definir un logformat con un nombre de tal forma que cuando hacemos referencia al nombre lo hacemos al logformat vinculado.</code>
Uso	<code>LogFormat logs/acceso_a_empresal.log</code>
Directiva ErrorLog	
Sintaxis	<code>ErrorLog nombre_fichero_archivo_registro</code>
Uso	<code>ErrorLog logs/acceso_a_empresal.log</code>
Directiva CustomLog	
Sintaxis	<code>CustomLog nombre_fichero_archivo_registro tubería_para_enviar_al_programa_la_información_de_registro [variable_de_entorno_opcional]</code>
Uso	<code>CustomLog logs/acceso_a_empresal.log</code>
Directiva CookieLog	
Sintaxis	<code>CookieLog nombre_fichero_archivo_registro</code>
Uso	<code>CookieLog logs/acceso_a_empresal.log</code>

En GNU/Linux puedes comprobar en tiempo real desde un terminal en el equipo que guarda los logs -que puede ser el propio equipo servidor web- que es lo que ocurre cuando accedes a una página web observando el contenido de los archivos de registro mediante el comando: `tail -f nombre_archivo_de_registro.log`

6.2.- Rotación de los archivos de registro (I).

Como los archivos de registro a medida que pasa el tiempo van incrementando su tamaño, debe existir una política de mantenimiento de registros para que éstos no consuman demasiados recursos

en el servidor, así es conveniente rotar los archivos de registro, esto es, hay que depurarlos, comprimirlos y guardarlos. Básicamente tienes dos opciones para rotar tus registros: **rotatelogs** un programa proporcionado por Apache, o **logrotate**, una utilidad presente en la mayoría de los sistemas GNU/Linux.

No debes olvidar que la información recopilada en los **ficheros log** se debe conservar al menos durante 1 año por eventuales necesidades legales, de este modo, además de rotarlos se opta habitualmente por **comprimir logs**.



Uso de rotatelogs

```
CustomLog "|ruta_rotatelogs ruta_log_a_rotar numero_segundos|tamaño_máximoMB" alias_logformat
```

Ejemplos

Rotar el archivo de registro access.log cada 24 horas

```
CustomLog "|/usr/sbin/rotatelogs /var/log/apache2/access.log 86400" common
```

Rotar el archivo de registro access.log cada vez que alcanza un tamaño de 5 megabytes

```
CustomLog "|/usr/sbin/rotatelogs /var/logs/apache2/access.log 5M" common
```

Rotar el archivo de registro error.log cada vez que alcanza un tamaño de 5 megabytes y el archivo se guardará con el sufijo de formato : YYYY-mm-dd-HH_MM_SS (Año-Mes-Día-Hora_Minutos_Segundos)

```
ErrorLog "|/usr/sbin/rotatelogs /var/logs/errorlog.%Y-%m-%d-%H %M %S 5M" common
```

Los ficheros rotados por intervalo de tiempo, lo harán siempre y cuando en el intervalo de tiempo definido existan nuevos datos.

Por defecto, si no se define formato mediante ningún modificador % para guardar los archivos de registro, el sufijo nnnnnnnnnn (10 cifras) se agrega automáticamente y es el tiempo en segundos tras pasados desde las 24 horas (medianoche).

El **alias logformat** es muy interesante, porque permite definir un grupo de modificadores en una palabra, de tal forma que incorporando esa palabra en la directiva log correspondiente estás activando todo un grupo de modificadores. En Apache existen predefinidos en el archivo **/etc/apache2/apache2.conf** los alias **logformat**: **vhost_combined**, **combined**, **common**, **referer** y **agent**, que puedes ver a continuación

Alias predefinidos

Alias logformat predefinidos en /etc/apache2/apache2.conf

```
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" vhost combined
LogFormat "%h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %O" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
```

Es conveniente que le des una visita al manual de **rotatelogs**: `man rotatelogs`.

6.2.1.- Rotación de los archivos de registro (II).

El programa **logrotate** rota, comprime y envía archivos de registro a diario, semanalmente, mensualmente o según el tamaño del archivo. Suele emplearse en una tarea diaria del cron (*Programa empleado en sistemas GNU/Linux para la automatización de tareas a intervalos regulares: minutos, horas, días ...*).



En **Debian** puedes encontrar los siguientes archivos de configuración para **logrotate**:

- ✓ **/etc/logrotate.conf** : Define los parámetros globales, esto es, los parámetros por defecto de logrotate. Te mostramos un fichero de este tipo:

```
# ejecutar "man logrotate" para más información

# rotar log semanalmente
weekly

# mantener logs durante 4 semanas
rotate 4

# rotar y crear nuevo log aunque esté vacío el anterior
create
# descomentar si quieres comprimir logs
#compress

# ubicación de paquetes para el rotado de logs
include /etc/logrotate.d

# los logs wtmp o btmp los haremos rotar aquí
/var/log/wtmp {
    missingok
    monthly
    create 0664 root utmp
    rotate 1
}

/var/log/btmp {
    missingok
    monthly
    create 0664 root utmp
    rotate 1
}

# los logs del sistema se pueden rotar aquí
```

- ✓ **/etc/logrotate.d/apache2** : Define para apache2 el rotado de logs, todos aquellos parámetros que no se encuentren aquí recogen su valor del fichero /etc/logrotate.conf. Puedes ver un archivo tipo a continuación:

```
/var/log/apache2/*.log {
    weekly → rotar log semanalmente
    missingok
    rotate 52 → mantener los logs durante 52 semanas
    compress → Archivos comprimidos mediante gzip por defecto
    delaycompress
    notifempty
    create 640 root adm → rotar y crear nuevo log, aunque esté vacío el anterior, con
    permisos 640, usuario root y grupo adm
    sharedscripts
    postrotate → Una vez rotado se recarga la configuración de apache2
        /etc/init.d/apache2 reload > /dev/null
    endscript
}
```

Uso de logrotate

Comprobar la correcta configuración de la rotación de un log

```
/usr/sbin/logrotate -d /etc/logrotate.d/apache2
```

Forzar la ejecución de logrotate

```
/usr/sbin/logrotate -f /etc/logrotate.conf
```

/etc/cron.daily/logrotate: Fichero tipo para ejecutar logrotate diariamente en el cron

```
#!/bin/sh

test -x /usr/sbin/logrotate || exit 0

/usr/sbin/logrotate /etc/logrotate.conf
```

Ejemplo para añadir al archivo crontab del sistema (crontab -e)

```
# Rotar logs de apache con logrotate a las 3 am

0 03 * * * root /usr/sbin/logrotate /etc/logrotate.conf > /dev/null 2>&1
```

El **rotado de logs** descrito anteriormente lo podemos aplicar a cualquier otra herramienta del sistema. Es conveniente que le des una visita al manual de **logrotate**: `man logrotate`.

Busca las palabras escondidas en la sopa de letras

	Z	V	J	M	Q	T	B	Z	A	M	I	V	
1. Formato archivo de registro	F	H	I	P	S	U	Y	L	H	S	N	H	CLF
2. Dominios independientes configurables	V	B	A	R	P	E	C	O	I	R	S	S	VIRTUALHOST
	A	D	T	U	T	Y	I	G	D	O	P	L	
3. Protocolo criptográfico	L	X	H	U	P	U	M	S	F	L	X	A	SSL
4. Sistema de autenticación de usuarios	T	J	T	I	N	O	A	E	T	B	D	E	LDAP
	Z	C	L	F	Y	E	U	L	S	Z	X	G	LOGS
5. Archivos de registro	N	H	V	B	N	L	T	S	H	R	K	J	
6. Verisign, Thawte, ...	A	C	F	G	B	K	P	B	N	O	T	O	AC
7. Asegurar http	F	T	Y	U	I	T	K	P	E	S	S	W	HTTPS
8. Políticas para los cambios de registro	K	O	R	O	T	A	C	I	O	N	H	T	ROTACIÓN
	W	S	X	H	V	G	H	R	U	B	N	Y	
	F	G	H	J	K	L	R	T	F	I	V	S	

Rellena los huecos con la palabra: `a2ensite` `available` `443` `enabled` `a2enmod` `80`

Servidores web

En apache2 utilizas el comando		para habilitar módulos	a2enmod
El puerto TCP		suele identificar a HTTPS	443
En apache2 utilizas el comando		para habilitar sitios	a2ensite
En apache2 el directorio mods-		contiene los módulos habilitados	enabled
En apache2 el directorio mods-		contiene los módulos posibles	available
El puerto TCP		suele identificar a HTTP	80

7.- Despliegue de aplicaciones sobre servidores Web.

Caso práctico

La empresa ha quedado muy contenta con el proyecto realizado por BK Programación, con lo cual ha considerado la posibilidad de contratarlos para un nuevo proyecto: la creación de una tienda virtual para la venta del material de la empresa a través de Internet. Para ello mantuvieron una reunión con los siguientes integrantes de BK Programación: Ada, la directora de la empresa y Juan el encargado de desarrollo de aplicaciones web.

-Juan -comentó-, pienso que se podría aprovechar para este proyecto varias aplicaciones de software libre, así el costo se abarataría y la comunidad de programadores es una garantía para la estabilidad del proyecto.

-Entonces -preguntó el representante de la empresa-, el desarrollo del proyecto mediante software libre y no la creación de una tienda virtual propia ¿reduciría el costo y el tiempo de desarrollo del proyecto?

-Sí, -dijo Juan-, existen varias aplicaciones de software libre en el mercado para tiendas virtuales, como: OpenCart, Magento, osCommerce.

-¿Cuál nos recomiendas?

-Pues, hoy en día, OpenCart, pero cualquiera de las tres son una buena elección.

Normalmente las aplicaciones sobre servidores web necesitan de los siguientes elementos para su correcto funcionamiento: `soporte php` y `soporte sql`.

El servidor web puede tener soporte php, pero el soporte sql debe ser ofrecido por otro servidor al que pueda acceder el servidor web. Este servidor con soporte sql puede estar configurado en el mismo equipo que el servidor web o en otro.

El procedimiento suele ser el siguiente:

1. Se descarga la aplicación.
2. Se configura para que sea visible a través del servidor web.
3. Suele traer una página de instalación que verifique si el servidor web cumple los requisitos para la instalación de la aplicación.
4. Es necesaria antes de finalizar el proceso de instalación autenticarse al servidor sql con un usuario con permisos para crear/modificar una base de datos. Puede que previamente se tenga que crear la base de datos para que el proceso de instalación genere las tablas necesarias en la misma.
5. Se pide un usuario y contraseña para poder acceder a la aplicación web.
6. Fin de la instalación.

A continuación, en el siguiente documento puedes ver un ejemplo basado en la aplicación **Opencart**.

En este documento se supone que tienes funcionando el siguiente entorno básico: [Apache](#), [MySQL](#) y [PHP](#). En Debian 6, instalado Apache, puedes lograrlo con el comando:

```
apt-get install libapache2-mod-auth-mysql mysql-server-5.5 php5-mysql curl php5-curl php5-gd libgd-tools
```

Otra buena opción sería instalar el paquete [XAMMP para GNU/Linux](#):

En los siguientes enlaces encontrarás demos de las aplicaciones para tienda virtual: OpenCart, Magento, osCommerce.

<http://www.opencart.com/index.php?route=demonstration/demonstration>

<http://demo.magentocommerce.com/>

<http://demo.oscommerce.com/>

Anexo I - /etc/apache/sites-available/default

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>

    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log

    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
    LogLevel warn

    CustomLog ${APACHE_LOG_DIR}/access.log combined

    Alias /doc/ "/usr/share/doc/"
    <Directory "/usr/share/doc/">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 127.0.0.0/255.0.0.0 ::1/128
    </Directory>
</VirtualHost>
```

Anexo II - /etc/mime.types

```
#####
#
# MIME-TYPES and the extensions that represent them
#
# This file is part of the "mime-support" package. Please send email (not a
# bug report) to mime-support@packages.debian.org if you would like new types
# and/or extensions to be added.
#
# The reason that all types are managed by the mime-support package instead
# allowing individual packages to install types in much the same way as they
# add entries in to the mailcap file is so these types can be referenced by
# other programs (such as a web server) even if the specific support package
# for that type is not installed.
#
# Users can add their own types if they wish by creating a ".mime.types"
# file in their home directory. Definitions included there will take
# precedence over those listed here.
#
# Note: Compression schemes like "gzip", "bzip", and "compress" are not
# actually "mime-types". They are "encodings" and hence must _not_ have
# entries in this file to map their extensions. The "mime-type" of an
# encoded file refers to the type of data that has been encoded, not the
# type of encoding.
#
#####

application/activemessage
application/andrew-inset          ez
application/annodex               anx
application/applefile
application/atom+xml              atom
application/atomcat+xml           atomcat
application/atomserv+xml          atomsrv
application/atomicmail
application/batch-SMTP
application/beep+xml
application/bbolin                lin
application/cals-1840
application/cap                   cap pcap
application/commonground
application/cu-seeme              cu
application/cybercash
application/davmount+xml          davmount
application/dca-rft
application/dec-dx
application/docbook+xml
application/dsptype               tsp
application/dvcs
application/ecmascript            es
application/edi-consent
application/edi-x12
application/edifact
application/eshop
application/font-tdpfr
application/futuresplash          spl
application/ghostview
application/hta                   hta
application/http
application/hyperstudio
application/iges
application/index
application/index.cmd
application/index.obj
application/index.response
application/index.vnd
application/iotp
application/ipp
application/isup
application/java-archive           jar
application/java-serialized-object ser
application/java-vm               class
application/javascript            js
application/m3g                   m3g
application/mac-binhex40          hqx
```

application/mac-compactpro	cpt
application/macwriteii	
application/marc	
application/mathematica	nb nbp
application/ms-tnef	
application/msaccess	mdb
application/msword	doc dot
application/mxf	mxf
application/news-message-id	
application/news-transmission	
application/ocsp-request	
application/ocsp-response	
application/octet-stream	bin
application/oda	oda
application/ogg	ogx
application/parityfec	
application/pdf	pdf
application/pgp-encrypted	
application/pgp-keys	key
application/pgp-signature	pgp
application/pics-rules	prf
application/pkcs10	
application/pkcs7-mime	
application/pkcs7-signature	
application/pkix-cert	
application/pkix-crl	
application/pkixcmp	
application/postscript	ps ai eps epsi epsf eps2 eps3
application/prs.alvestrand.titrax-sheet	
application/prs.cww	
application/prs.nprend	
application/qsig	
application/rar	rar
application/rdf+xml	rdf
application/remote-printing	
application/riscos	
application/rss+xml	rss
application/rtf	rtf
application/sdp	
application/set-payment	
application/set-payment-initiation	
application/set-registration	
application/set-registration-initiation	
application/sgml	
application/sgml-open-catalog	
application/sieve	
application/slate	
application/smil	smi smil
application/timestamp-query	
application/timestamp-reply	
application/vemmi	
application/whoispp-query	
application/whoispp-response	
application/wita	
application/x400-bp	
application/xhtml+xml	xhtml xht
application/xml	xml xsl xsd
application/xml-dtd	
application/xml-external-parsed-entity	
application/xspf+xml	xspf
application/zip	zip
application/vnd.3M.Post-it-Notes	
application/vnd.accpac.simply.aso	
application/vnd.accpac.simply.imp	
application/vnd.acucobol	
application/vnd.aether.imp	
application/vnd.android.package-archive	apk
application/vnd.anser-web-certificate-issue-initiation	
application/vnd.anser-web-funds-transfer-initiation	
application/vnd.audiograph	
application/vnd.bmi	
application/vnd.businessobjects	
application/vnd.canon-cpdl	
application/vnd.canon-lips	
application/vnd.cinderella	cdy
application/vnd.claymore	
application/vnd.commerce-battelle	

```

application/vnd.commonspace
application/vnd.comsocaller
application/vnd.contact.cmsg
application/vnd.cosmocaller
application/vnd.ctc-posml
application/vnd.cups-postscript
application/vnd.cups-raster
application/vnd.cups-raw
application/vnd.cybank
application/vnd.dna
application/vnd.dpgraph
application/vnd.dxr
application/vnd.ecdis-update
application/vnd.ecowin.chart
application/vnd.ecowin.filerequest
application/vnd.ecowin.fileupdate
application/vnd.ecowin.series
application/vnd.ecowin.seriesrequest
application/vnd.ecowin.seriesupdate
application/vnd.enliven
application/vnd.epson.esf
application/vnd.epson.msfx
application/vnd.epson.quickanime
application/vnd.epson.salt
application/vnd.epson.ssf
application/vnd.ericsson.quickcall
application/vnd.eudora.data
application/vnd.fdf
application/vnd.ffsns
application/vnd.flographit
application/vnd.framemaker
application/vnd.fsc.weblaunch
application/vnd.fujitsu.oasys
application/vnd.fujitsu.oasys2
application/vnd.fujitsu.oasys3
application/vnd.fujitsu.oasysgp
application/vnd.fujitsu.oasysprs
application/vnd.fujixerox.ddd
application/vnd.fujixerox.docuworks
application/vnd.fujixerox.docuworks.binder
application/vnd.fut-misnet
application/vnd.google-earth.kml+xml      kml
application/vnd.google-earth.kmz          kmz
application/vnd.grafeq
application/vnd.groove-account
application/vnd.groove-identity-message
application/vnd.groove-injector
application/vnd.groove-tool-message
application/vnd.groove-tool-template
application/vnd.groove-vcard
application/vnd.hhe.lesson-player
application/vnd.hp-HPGL
application/vnd.hp-PCL
application/vnd.hp-PCLXL
application/vnd.hp-hpid
application/vnd.hp-hps
application/vnd.httpphone
application/vnd.hzn-3d-crossword
application/vnd.ibm.Minipay
application/vnd.ibm.afplinedata
application/vnd.ibm.modcap
application/vnd.informix-visionary
application/vnd.intercon.formnet
application/vnd.intertrust.digibox
application/vnd.intertrust.nncp
application/vnd.intu.qbo
application/vnd.intu.qfx
application/vnd.irepository.package+xml
application/vnd.is-xpr
application/vnd.japannet-directory-service
application/vnd.japannet-jpnstore-wakeup
application/vnd.japannet-payment-wakeup
application/vnd.japannet-registration
application/vnd.japannet-registration-wakeup
application/vnd.japannet-setstore-wakeup
application/vnd.japannet-verification
application/vnd.japannet-verification-wakeup
application/vnd.koan

```

```

application/vnd.lotus-1-2-3
application/vnd.lotus-approach
application/vnd.lotus-freelance
application/vnd.lotus-notes
application/vnd.lotus-organizer
application/vnd.lotus-screencam
application/vnd.lotus-wordpro
application/vnd.mcd
application/vnd.mediastation.cdkey
application/vnd.meridian-slingshot
application/vnd.mif
application/vnd.minisoft-hp3000-save
application/vnd.mitsubishi.misty-guard.trustweb
application/vnd.mobius.daf
application/vnd.mobius.dis
application/vnd.mobius.msl
application/vnd.mobius.plc
application/vnd.mobius.txf
application/vnd.motorola.flexsuite
application/vnd.motorola.flexsuite.adsi
application/vnd.motorola.flexsuite.fis
application/vnd.motorola.flexsuite.gotap
application/vnd.motorola.flexsuite.kmr
application/vnd.motorola.flexsuite.ttc
application/vnd.motorola.flexsuite.wem
application/vnd.mozilla.xul+xml          xul
application/vnd.ms-artgalry
application/vnd.ms-asf
application/vnd.ms-excel                  xls xlb xlt
application/vnd.ms-lrm
application/vnd.ms-pki.seccat             cat
application/vnd.ms-pki.stl                stl
application/vnd.ms-powerpoint              ppt pps
application/vnd.ms-project
application/vnd.ms-tnef
application/vnd.ms-works
application/vnd.mseq
application/vnd.msign
application/vnd.music-niff
application/vnd.musician
application/vnd.netfpx
application/vnd.noblenet-directory
application/vnd.noblenet-sealer
application/vnd.noblenet-web
application/vnd.novadigm.EDM
application/vnd.novadigm.EDX
application/vnd.novadigm.EXT
application/vnd.oasis.opendocument.chart      odc
application/vnd.oasis.opendocument.database   odb
application/vnd.oasis.opendocument.formula     odf
application/vnd.oasis.opendocument.graphics   odg
application/vnd.oasis.opendocument.graphics-template  otg
application/vnd.oasis.opendocument.image      odi
application/vnd.oasis.opendocument.presentation  odp
application/vnd.oasis.opendocument.presentation-template  otp
application/vnd.oasis.opendocument.spreadsheet ods
application/vnd.oasis.opendocument.spreadsheet-template  ots
application/vnd.oasis.opendocument.text        odt
application/vnd.oasis.opendocument.text-master  odm
application/vnd.oasis.opendocument.text-template  ott
application/vnd.oasis.opendocument.text-web     oth
application/vnd.osa.netdeploy
application/vnd.openxmlformats-officedocument.spreadsheetml.sheet  xlsx
application/vnd.openxmlformats-officedocument.spreadsheetml.template  xltx
application/vnd.openxmlformats-officedocument.presentationml.presentation  pptx
application/vnd.openxmlformats-officedocument.presentationml.slideshow  ppsx
application/vnd.openxmlformats-officedocument.presentationml.template  potx
application/vnd.openxmlformats-officedocument.wordprocessingml.document  docx
application/vnd.openxmlformats-officedocument.wordprocessingml.template  dotx
application/vnd.palm
application/vnd.pg.format
application/vnd.pg.asasli
application/vnd.powerbuilder6
application/vnd.powerbuilder6-s
application/vnd.powerbuilder7
application/vnd.powerbuilder7-s
application/vnd.powerbuilder75

```

application/vnd.powerbuilder75-s	
application/vnd.previewsystems.box	
application/vnd.publishare-delta-tree	
application/vnd.pvi.ptidl	
application/vnd.pwg-xhtml-print+xml	
application/vnd.rapid	
application/vnd.rim.cod	cod
application/vnd.s3sms	
application/vnd.seemail	
application/vnd.shana.informed.formdata	
application/vnd.shana.informed.formtemplate	
application/vnd.shana.informed.interchange	
application/vnd.shana.informed.package	
application/vnd.smaf	mmf
application/vnd.sss-cod	
application/vnd.sss-dtf	
application/vnd.sss-ntf	
application/vnd.stardivision.calc	sdc
application/vnd.stardivision.chart	sds
application/vnd.stardivision.draw	sda
application/vnd.stardivision.impress	sdd
application/vnd.stardivision.math	sdf
application/vnd.stardivision.writer	sdw
application/vnd.stardivision.writer-global	sgl
application/vnd.street-stream	
application/vnd.sun.xml.calc	sxc
application/vnd.sun.xml.calc.template	stc
application/vnd.sun.xml.draw	sxd
application/vnd.sun.xml.draw.template	std
application/vnd.sun.xml.impress	sxi
application/vnd.sun.xml.impress.template	sti
application/vnd.sun.xml.math	sxm
application/vnd.sun.xml.writer	sxw
application/vnd.sun.xml.writer.global	sxg
application/vnd.sun.xml.writer.template	stw
application/vnd.svd	
application/vnd.swiftview-ics	
application/vnd.symbian.install	sis
application/vnd.triscape.mxs	
application/vnd.trueapp	
application/vnd.truedoc	
application/vnd.tve-trigger	
application/vnd.ufdl	
application/vnd.uplanet.alert	
application/vnd.uplanet.alert-wbxml	
application/vnd.uplanet.bearer-choice	
application/vnd.uplanet.bearer-choice-wbxml	
application/vnd.uplanet.cacheop	
application/vnd.uplanet.cacheop-wbxml	
application/vnd.uplanet.channel	
application/vnd.uplanet.channel-wbxml	
application/vnd.uplanet.list	
application/vnd.uplanet.list-wbxml	
application/vnd.uplanet.listcmd	
application/vnd.uplanet.listcmd-wbxml	
application/vnd.uplanet.signal	
application/vnd.vcx	
application/vnd.vectorworks	
application/vnd.vidsoft.vidconference	
application/vnd.visio	vsd
application/vnd.vividence.scriptfile	
application/vnd.wap.sic	
application/vnd.wap.slc	
application/vnd.wap.wbxml	wbxml
application/vnd.wap.wmlc	wmlc
application/vnd.wap.wmlscriptc	wmlsc
application/vnd.webturbo	
application/vnd.wordperfect	wpd
application/vnd.wordperfect5.1	wp5
application/vnd.wrq-hp3000-labelled	
application/vnd.wt.stf	
application/vnd.xara	
application/vnd.xfdl	
application/vnd.yellowriver-custom-menu	
application/x-123	wk
application/x-7z-compressed	7z
application/x-abiword	abw
application/x-apple-diskimage	dmg

application/x-bcpio	bcpio
application/x-bittorrent	torrent
application/x-cab	cab
application/x-cbr	cbr
application/x-cbz	cbz
application/x-cdf	cdf cda
application/x-cdlink	vcd
application/x-chess-pgn	pgn
application/x-core	
application/x-cpio	cpio
application/x-csh	csh
application/x-debian-package	deb udeb
application/x-director	dcr dir dxr
application/x-dms	dms
application/x-doom	wad
application/x-dvi	dvi
application/x-httpd-eruby	rhtml
application/x-executable	
application/x-font	pfa pfb gsf pcf pcf.Z
application/x-freemind	mm
application/x-futuresplash	spl
application/x-gnumeric	gnumeric
application/x-go-sgf	sgf
application/x-graphing-calculator	gcf
application/x-gtar	gtar tgz taz
application/x-hdf	hdf
application/x-httpd-php	phtml pht php
application/x-httpd-php-source	phps
application/x-httpd-php3	php3
application/x-httpd-php3-preprocessed	php3p
application/x-httpd-php4	php4
application/x-httpd-php5	php5
application/x-ica	ica
application/x-info	info
application/x-internet-signup	ins isp
application/x-iphone	iii
application/x-iso9660-image	iso
application/x-jam	jam
application/x-java-applet	
application/x-java-bean	
application/x-java-jnlp-file	jnlp
application/x-jmol	jnz
application/x-kchart	chrt
application/x-kdelnk	
application/x-killustrator	kil
application/x-koan	skp skd skt skm
application/x-kpresenter	kpr kpt
application/x-kspread	ksp
application/x-kword	kwd kwt
application/x-latex	latex
application/x-lha	lha
application/x-lyx	lyx
application/x-lzh	lzh
application/x-lzx	lzx
application/x-maker	frm maker frame fm fb book fbdoc
application/x-mif	mif
application/x-ms-wmd	wmd
application/x-ms-wmz	wmz
application/x-msdos-program	com exe bat dll
application/x-msi	msi
application/x-netcdf	nc
application/x-ns-proxy-autoconfig	pac dat
application/x-nwc	nwc
application/x-object	o
application/x-oz-application	oza
application/x-pkcs7-certreqresp	p7r
application/x-pkcs7-crl	crl
application/x-python-code	pyc pyo
application/x-qgis	qgs shp shx
application/x-quicktimeplayer	qtl
application/x-redhat-package-manager	rpm
application/x-ruby	rb
application/x-rx	
application/x-sh	sh
application/x-shar	shar
application/x-shellscript	
application/x-shockwave-flash	swf swfl

application/x-silverlight	scr
application/x-stuffit	sit sitx
application/x-sv4cpio	sv4cpio
application/x-sv4crc	sv4crc
application/x-tar	tar
application/x-tcl	tcl
application/x-tex-gf	gf
application/x-tex-pk	pk
application/x-texinfo	texinfo texi
application/x-trash	~ % bak old sik
application/x-troff	t tr roff
application/x-troff-man	man
application/x-troff-me	me
application/x-troff-ms	ms
application/x-ustar	ustar
application/x-videolan	
application/x-wais-source	src
application/x-wingz	wz
application/x-x509-ca-cert	crt
application/x-xcf	xcf
application/x-xfig	fig
application/x-xpinstall	xpi
audio/32kadpcm	
audio/3gpp	
audio/amr	amr
audio/amr-wb	awb
audio/amr	amr
audio/amr-wb	awb
audio/annodex	axa
audio/basic	au snd
audio/flac	flac
audio/g.722.1	
audio/l16	
audio/midi	mid midi kar
audio/mp4a-latm	
audio/mpa-robust	
audio/mpeg	mpga mpega mp2 mp3 m4a
audio/mpegurl	m3u
audio/ogg	oga ogg spx
audio/parityfec	
audio/prs.sid	sid
audio/telephone-event	
audio/tone	
audio/vnd.cisco.nse	
audio/vnd.cns.anp1	
audio/vnd.cns.infl	
audio/vnd.digital-winds	
audio/vnd.everad.plj	
audio/vnd.lucent.voice	
audio/vnd.nortel.vbk	
audio/vnd.nuera.ecelp4800	
audio/vnd.nuera.ecelp7470	
audio/vnd.nuera.ecelp9600	
audio/vnd.octel.sbc	
audio/vnd.qcelp	
audio/vnd.rhetorex.32kadpcm	
audio/vnd.vmx.cvsd	
audio/x-aiff	aif aiff aifc
audio/x-gsm	gsm
audio/x-mpegurl	m3u
audio/x-ms-wma	wma
audio/x-ms-wax	wax
audio/x-pn-realaudio-plugin	
audio/x-pn-realaudio	ra rm ram
audio/x-realaudio	ra
audio/x-scpls	pls
audio/x-sd2	sd2
audio/x-wav	wav
chemical/x-alchemy	alc
chemical/x-cache	cac cache
chemical/x-cache-csf	csf
chemical/x-cactvs-binary	cbin cascii ctab
chemical/x-cdx	cdx
chemical/x-cerius	cer
chemical/x-chem3d	c3d
chemical/x-chemdraw	chm

chemical/x-cif	cif
chemical/x-cmdf	cmdf
chemical/x-cml	cml
chemical/x-compass	cpa
chemical/x-crossfire	bsd
chemical/x-csml	csml csm
chemical/x-ctx	ctx
chemical/x-cxf	cxf cef
#chemical/x-daylight-smiles	smi
chemical/x-embl-dl-nucleotide	emb embl
chemical/x-galactic-spc	spc
chemical/x-gameess-input	inp gam gamin
chemical/x-gaussian-checkpoint	fch fchk
chemical/x-gaussian-cube	cub
chemical/x-gaussian-input	gau gjc gjf
chemical/x-gaussian-log	gal
chemical/x-gcg8-sequence	gcg
chemical/x-genbank	gen
chemical/x-hin	hin
chemical/x-isostar	istr ist
chemical/x-jcamp-dx	jdx dx
chemical/x-kinemage	kin
chemical/x-macmolecule	mcm
chemical/x-macromodel-input	mmmd mmod
chemical/x-mdl-molfile	mol
chemical/x-mdl-rdfile	rd
chemical/x-mdl-rxnfile	rxn
chemical/x-mdl-sdfile	sd sdf
chemical/x-mdl-tgf	tgf
#chemical/x-mif	mif
chemical/x-mmCIF	mcif
chemical/x-mol2	mol2
chemical/x-molconn-Z	b
chemical/x-mopac-graph	gpt
chemical/x-mopac-input	mop mopcert mpc zmt
chemical/x-mopac-out	moo
chemical/x-mopac-vib	mvb
chemical/x-ncbi-asn1	asn
chemical/x-ncbi-asn1-ascii	prt ent
chemical/x-ncbi-asn1-binary	val aso
chemical/x-ncbi-asn1-spec	asn
chemical/x-pdb	pdb ent
chemical/x-rosdal	ros
chemical/x-swissprot	sw
chemical/x-vamas-iso14976	vms
chemical/x-vmd	vmd
chemical/x-xtel	xtel
chemical/x-xyz	xyz
image/cgm	
image/g3fax	
image/gif	gif
image/ief	ief
image/jpeg	jpeg jpg jpe
image/naplps	
image/pcx	pcx
image/png	png
image/prs.btif	
image/prs.pti	
image/svg+xml	svg svgz
image/tiff	tiff tif
image/vnd.cns.inf2	
image/vnd.djvu	djvu djv
image/vnd.dwg	
image/vnd.dxf	
image/vnd.fastbidsheet	
image/vnd.fpx	
image/vnd.fst	
image/vnd.fujixerox.edmics-mmr	
image/vnd.fujixerox.edmics-rlc	
image/vnd.mix	
image/vnd.net-fpx	
image/vnd.svf	
image/vnd.wap.wbmp	wbmp
image/vnd.xiff	
image/x-canon-cr2	cr2
image/x-canon-crw	crw

image/x-cmu-raster	ras
image/x-coreldraw	cdr
image/x-coreldrawpattern	pat
image/x-coreldrawtemplate	cdt
image/x-corelphotopaint	cpt
image/x-epson-erf	erf
image/x-icon	ico
image/x-jg	art
image/x-jng	jng
image/x-ms-bmp	bmp
image/x-nikon-nef	nef
image/x-olympus-orf	orf
image/x-photoshop	psd
image/x-portable-anymap	pnm
image/x-portable-bitmap	pbm
image/x-portable-graymap	pgm
image/x-portable-pixmap	ppm
image/x-rgb	rgb
image/x-xbitmap	xbm
image/x-xpixmap	xpm
image/x-xwindowdump	xwd
inode/chardevice	
inode/blockdevice	
inode/directory-locked	
inode/directory	
inode/fifo	
inode/socket	
message/delivery-status	
message/disposition-notification	
message/external-body	
message/http	
message/s-http	
message/news	
message/partial	
message/rfc822	eml
model/iges	igs iges
model/mesh	msh mesh silo
model/vnd.dwf	
model/vnd.flatland.3dml	
model/vnd.gdl	
model/vnd.gs-gdl	
model/vnd.gtw	
model/vnd.mts	
model/vnd.vtu	
model/vrml	wrl vrml
model/x3d+vrml	x3dv
model/x3d+xml	x3d
model/x3d+binary	x3db
multipart/alternative	
multipart/appledouble	
multipart/byteranges	
multipart/digest	
multipart/encrypted	
multipart/form-data	
multipart/header-set	
multipart/mixed	
multipart/parallel	
multipart/related	
multipart/report	
multipart/signed	
multipart/voice-message	
text/cache-manifest	manifest
text/calendar	ics icz
text/css	css
text/csv	csv
text/directory	
text/english	
text/enriched	
text/h323	323
text/html	html htm shtml
text/iuls	uls
text/mathml	mml
text/parityfec	

text/plain	asc txt text pot brf
text/prs.lines.tag	
text/rfc822-headers	
text/richtext	rtx
text/rtf	
text/scriptlet	sct wsc
text/t140	
text/texmacs	tm ts
text/tab-separated-values	tsv
text/uri-list	
text/vnd.abc	
text/vnd.curl	
text/vnd.DMClientScript	
text/vnd.flatland.3dml	
text/vnd.fly	
text/vnd.fmi.flexstor	
text/vnd.in3d.3dml	
text/vnd.in3d.spot	
text/vnd.IPTC.NewsML	
text/vnd.IPTC.NITF	
text/vnd.latex-z	
text/vnd.motorola.reflex	
text/vnd.ms-mediapackage	
text/vnd.sun.j2me.app-descriptor	jad
text/vnd.wap.si	
text/vnd.wap.sl	
text/vnd.wap.wml	wml
text/vnd.wap.wmlscript	wmls
text/x-bibtex	bib
text/x-boo	boo
text/x-c++hdr	h++ hpp hxx hh
text/x-c++src	c++ cpp cxx cc
text/x-chdr	h
text/x-component	htc
text/x-crontab	
text/x-csh	csh
text/x-csrc	c
text/x-dsrc	d
text/x-diff	diff patch
text/x-haskell	hs
text/x-java	java
text/x-literate-haskell	lhs
text/x-makefile	
text/x-moc	moc
text/x-pascal	p pas
text/x-pcs-gcd	gcd
text/x-perl	pl pm
text/x-python	py
text/x-scala	scala
text/x-server-parsed-html	
text/x-setext	etx
text/x-sh	sh
text/x-tcl	tcl tk
text/x-tex	tex ltx sty cls
text/x-vcalendar	vcs
text/x-vcard	vcf
video/3gpp	3gp
video/annodex	axv
video/dl	dl
video/dv	dif dv
video/fli	fli
video/gl	gl
video/mpeg	mpeg mpg mpe
video/mp4	mp4
video/quicktime	qt mov
video/mp4v-es	
video/ogg	ogv
video/parityfec	
video/pointer	
video/vnd.fvt	
video/vnd.motorola.video	
video/vnd.motorola.videop	
video/vnd.mpegurl	mxu
video/vnd.mts	
video/vnd.nokia.interleaved-multimedia	
video/vnd.vivo	

video/x-flv	flv
video/x-la-asf	lsf lsf
video/x-mng	mng
video/x-ms-asf	asf asx
video/x-ms-wm	wm
video/x-ms-wmv	wmv
video/x-ms-wmx	wmx
video/x-ms-wvx	wvx
video/x-msvideo	avi
video/x-sgi-movie	movie
video/x-matroska	mpv mkv
x-conference/x-cooltalk	ice
x-epoc/x-sisx-app	sisx
x-world/x-vrml	vrml vrml wrl

Anexo III - /etc/apache2/sites-available/default-ssl

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>

    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log

    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
    LogLevel warn

    CustomLog ${APACHE_LOG_DIR}/ssl_access.log combined

    Alias /doc/ "/usr/share/doc/"
    <Directory "/usr/share/doc/">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 127.0.0.0/255.0.0.0 ::1/128
    </Directory>

    #   SSL Engine Switch:
    #   Enable/Disable SSL for this virtual host.
    SSLEngine on

    #   A self-signed (snakeoil) certificate can be created by installing
    #   the ssl-cert package. See
    #   /usr/share/doc/apache2.2-common/README.Debian.gz for more info.
    #   If both key and certificate are stored in the same file, only the
    #   SSLCertificateFile directive is needed.
    SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
    SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key

    #   Server Certificate Chain:
    #   Point SSLCertificateChainFile at a file containing the
    #   concatenation of PEM encoded CA certificates which form the
    #   certificate chain for the server certificate. Alternatively
    #   the referenced file can be the same as SSLCertificateFile
    #   when the CA certificates are directly appended to the server
    #   certificate for convinience.
    #SSLCertificateChainFile /etc/apache2/ssl.crt/server-ca.crt

    #   Certificate Authority (CA):
    #   Set the CA certificate verification path where to find CA
    #   certificates for client authentication or alternatively one
    #   huge file containing all of them (file must be PEM encoded)
    #   Note: Inside SSLCACertificatePath you need hash symlinks
    #         to point to the certificate files. Use the provided
    #         Makefile to update the hash symlinks after changes.
    #SSLCACertificatePath /etc/ssl/certs/
    #SSLCACertificateFile /etc/apache2/ssl.crt/ca-bundle.crt

    #   Certificate Revocation Lists (CRL):
    #   Set the CA revocation path where to find CA CRLs for client
```

```

# authentication or alternatively one huge file containing all
# of them (file must be PEM encoded)
# Note: Inside SSLCAREvocationPath you need hash symlinks
#       to point to the certificate files. Use the provided
#       Makefile to update the hash symlinks after changes.
#SSLCARevocationPath /etc/apache2/ssl.crl/
#SSLCARevocationFile /etc/apache2/ssl.crl/ca-bundle.crl

# Client Authentication (Type):
# Client certificate verification type and depth. Types are
# none, optional, require and optional_no_ca. Depth is a
# number which specifies how deeply to verify the certificate
# issuer chain before deciding the certificate is not valid.
#SSLVerifyClient require
#SSLVerifyDepth 10

# Access Control:
# With SSLRequire you can do per-directory access control based
# on arbitrary complex boolean expressions containing server
# variable checks and other lookup directives. The syntax is a
# mixture between C and Perl. See the mod_ssl documentation
# for more details.
#<Location />
#SSLRequire (    %{SSL_CIPHER} !~ m/^(EXP|NULL)/ \
#               and %{SSL_CLIENT_S_DN_O} eq "Snake Oil, Ltd." \
#               and %{SSL_CLIENT_S_DN_OU} in {"Staff", "CA", "Dev"} \
#               and %{TIME_WDAY} >= 1 and %{TIME_WDAY} <= 5 \
#               and %{TIME_HOUR} >= 8 and %{TIME_HOUR} <= 20       ) \
#               or %{REMOTE_ADDR} =~ m/^192\.76\.162\. [0-9]+$/
#</Location>

# SSL Engine Options:
# Set various options for the SSL engine.
# o FakeBasicAuth:
#   Translate the client X.509 into a Basic Authorisation. This means that
#   the standard Auth/DBMAuth methods can be used for access control. The
#   user name is the 'one line' version of the client's X.509 certificate.
#   Note that no password is obtained from the user. Every entry in the user
#   file needs this password: 'xxj3lZMTZzkVA'.
# o ExportCertData:
#   This exports two additional environment variables: SSL_CLIENT_CERT and
#   SSL_SERVER_CERT. These contain the PEM-encoded certificates of the
#   server (always existing) and the client (only existing when client
#   authentication is used). This can be used to import the certificates
#   into CGI scripts.
# o StdEnvVars:
#   This exports the standard SSL/TLS related 'SSL_*' environment variables.
#   Per default this exportation is switched off for performance reasons,
#   because the extraction step is an expensive operation and is usually
#   useless for serving static content. So one usually enables the
#   exportation for CGI and SSI requests only.
# o StrictRequire:
#   This denies access when "SSLRequireSSL" or "SSLRequire" applied even
#   under a "Satisfy any" situation, i.e. when it applies access is denied
#   and no other module can change it.
# o OptRenegotiate:
#   This enables optimized SSL connection renegotiation handling when SSL
#   directives are used in per-directory context.
#SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
<FilesMatch "\.(cgi|shtml|phtml|php)$">
    SSLOptions +StdEnvVars
</FilesMatch>
<Directory /usr/lib/cgi-bin>
    SSLOptions +StdEnvVars
</Directory>

# SSL Protocol Adjustments:
# The safe and default but still SSL/TLS standard compliant shutdown
# approach is that mod_ssl sends the close notify alert but doesn't wait for
# the close notify alert from client. When you need a different shutdown
# approach you can use one of the following variables:
# o ssl-unclean-shutdown:
#   This forces an unclean shutdown when the connection is closed, i.e. no
#   SSL close notify alert is send or allowed to received. This violates
#   the SSL/TLS standard but is needed for some brain-dead browsers. Use
#   this when you receive I/O errors because of the standard approach where
#   mod_ssl sends the close notify alert.
# o ssl-accurate-shutdown:

```

```
# This forces an accurate shutdown when the connection is closed, i.e. a
# SSL close notify alert is send and mod_ssl waits for the close notify
# alert of the client. This is 100% SSL/TLS standard compliant, but in
# practice often causes hanging connections with brain-dead browsers. Use
# this only for browsers where you know that their SSL implementation
# works correctly.
# Notice: Most problems of broken clients are also related to the HTTP
# keep-alive facility, so you usually additionally want to disable
# keep-alive for those clients, too. Use variable "nokeepalive" for this.
# Similarly, one has to force some clients to use HTTP/1.0 to workaround
# their broken HTTP/1.1 implementation. Use variables "downgrade-1.0" and
# "force-response-1.0" for this.
BrowserMatch "MSIE [2-6]" \
    nokeepalive ssl-unclean-shutdown \
    downgrade-1.0 force-response-1.0
# MSIE 7 and newer should be able to use keepalive
BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown

</VirtualHost>
</IfModule>
```

Anexo IV - openssl_autofirmado.txt

```
/etc/apache2/tus-ssl# openssl req -new -nodes -keyout tupaginaweb.key -out tupaginaweb.csr
Generating a 1024 bit RSA private key
.....++++++
.+++++
writing new private key to 'tupaginaweb.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:autofirmado.ssl.empresa-proyecto.com
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
root@debian-servidor-fp:/etc/apache2/tus-ssl# openssl x509 -in tupaginaweb.csr -out
tupaginaweb.crt -req -signkey tupaginaweb.key -days 3650
Signature ok
subject=/C=ES/ST=Some-State/O=Internet Widgits Pty Ltd/CN=autofirmado.ssl.empresa-proyecto.com
Getting Private key
root@debian-servidor-fp:/etc/apache2/tus-ssl#
```

Anexo V - Instalación y configuración de OpenLDAP

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

Instalación de OpenLDAP

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

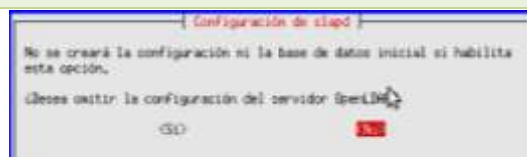
```
// Instalación del servidor LDAP
sudo apt-get install slapd ldap-utils
```

Configuración inicial de OpenLDAP

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

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

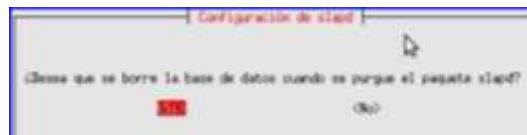
Lo primero que nos pregunta el asistente es si deseamos omitir la configuración del servidor LDAP:



Asistente de configuración de slapd

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

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



Pregunta sobre la eliminación de la base de datos

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



Utilización LDAP versión 2

Con esto habremos concluido la configuración inicial del servidor LDAP.

Arranque y parada manual del servidor LDAP

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

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

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

Anexo VI - Instalación y configuración del servidor OpenLDAP en Debian 6

Escenario: debian-servidor-fp --> IP: 192.168.200.250
Procedimiento a realizar como usuario root:

1) Actualiza el sistema operativo.

```
root@debian-servidor-fp:~# apt-get update
root@debian-servidor-fp:~# apt-get upgrade
```

2) Instala los paquetes necesarios para el funcionamiento de OpenLDAP. La instalación te pedirá una contraseña, como puedes ver a continuación la contraseña es 'admin'

```
root@debian-servidor-fp:~# apt-get install slapd ldap-utils
Administrator password: admin
Confirm password: admin
```

3) 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      1775/slapd
tcp6       0      0 :::389              :::*                  LISTEN      1775/slapd
```

4) Configura el servidor OpenLDAP. Los valores utilizados los puedes ver a continuación del comando

```
root@debian-servidor-fp:~# dpkg-reconfigure slapd
Omit OpenLDAP config ? No
Domain name : proyecto.com
organisation name : proyecto.com
admin password : admin
admin password : admin
database module to use : HDB
delete database when purging the package ? No
Move the previous database ? Si
Allow LDAPv2 ? No
```

5) 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,dc=com', similar a como se expone a continuación:

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

```
root@debian-servidor-fp:~# mv a.txt /etc/ldap/slapd.d/cn=config/olcDatabase=\{1\}hdb.ldif
```

```
root@debian-servidor-fp:~# nano /etc/ldap/slapd.d/cn=config/olcDatabase=\{1\}hdb.ldif
```

```
dn: olcDatabase={1}hdb
objectClass: olcDatabaseConfig
olcDatabase: {1}hdb
olcDbDirectory: /var/lib/ldap
olcSuffix: dc=proyecto,dc=com
olcAccess: {0}to attrs=userPassword,shadowLastChange by self write by anonymous auth by dn="cn=admin,dc=proyecto,dc=com" write by * none
olcAccess: {1}to dn.base="" by * read
olcAccess: {2}to * by self write by dn="cn=admin,dc=proyecto,dc=com" write by * read
olcLastMod: TRUE
olcRootDN: cn=admin,dc=proyecto,dc=com
olcRootPW: e1NTSEF9bThuNDVrOGZCRVhHVz1BYUpud0ZGYk1lQUtvanVsSnE=
olcDbCheckpoint: 512 30
olcDbConfig: {0}set_cachesize 0 2097152 0
olcDbConfig: {1}set_lik_max_objects 1500
olcDbConfig: {2}set_lik_max_locks 1500
olcDbConfig: {3}set_lik_max_lockers 1500
olcDbIndex: objectClass eq
structuralObjectClass: olcHdbConfig
entryUUID: 1e80cb3e-1f44-1030-9fab-8b0calca9cc2
creatorsName: cn=admin,cn=config
createTimestamp: 20110530200658Z
entryCSN: 20110530200658.710565Z#000000#000#000000
modifiersName: cn=admin,cn=config
modifyTimestamp: 20110530200658Z
```

7) Activa los cambios del servidor OpenLDAP

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

8) Testea el servidor OpenLDAP:

```
root@debian-servidor-fp:~# slaptest
hdb_db_open: database "dc=proyecto,dc=com": unclean shutdown detected; attempting recovery.
```

```
hdb_db_open: database "dc=proyecto,dc=com": recovery skipped in read-only mode. Run manual
recovery if errors are encountered.
config file testing succeeded

9) Instala los paquetes necesarios para que Apache funcione con LDAP
root@debian-servidor-fp:~# apt-get install libapache2-mod-vhost-ldap

10) Habilita el módulo LDAP para Apache:
root@debian-servidor-fp:~# a2enmod authnz_ldap

11) Reinicia Apache:
root@debian-servidor-fp:~# /etc/init.d/apache2 restart

12) Crea la estructura básica del dominio LDAP mediante la ejecución de un fichero basica.ldif
root@debian-servidor-fp:~# nano basica.ldif
# Objetos raíz del dominio
dn: dc=proyecto,dc=com
objectClass: top
objectClass: dcObject
objectClass: organization
o: proyecto.com
dc: proyecto
description: Raiz de dominio

# Usuarios
dn: ou=usuarios,dc=proyecto,dc=com
objectClass: organizationalUnit
ou: usuarios

# Grupos
dn: ou=grupos,dc=proyecto,dc=com
objectClass: organizationalUnit
ou: grupos

root@debian-servidor-fp:~# ldapadd -x -D cn=admin,dc=proyecto,dc=com -w admin -f basica.ldif
adding new entry "dc=proyecto,dc=com"

adding new entry "ou=usuarios,dc=proyecto,dc=com"

adding new entry "ou=grupos,dc=proyecto,dc=com"

13) Añadiendo un usuario a LDAP de nombre pruebas y contraseña: 123456 mediante el archivo
usuario.ldif:
root@debian-servidor-fp:~# cat usuario.ldif
# Usuario
dn: uid=pruebas,ou=usuarios,dc=proyecto,dc=com
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
uid: pruebas
sn: daw02
givenName: Pruebas
cn: Pruebas daw02
displayName: Pruebas DAW02
uidNumber: 10000
gidNumber: 10000
userPassword: 123456
gecos: Pruebas DAW02
loginShell: /bin/bash
homeDirectory: /home/pruebas
shadowExpire: -1
shadowFlag: 0
shadowWarning: 7
shadowMin: 8
shadowMax: 999999
shadowLastChange: 10877
mail: pruebas.daw02@proyecto.com
initials: PD

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

14) Reiniciar LDAP y Apache
root@debian-servidor-fp:/etc/apache2/sites-available# /etc/init.d/slaped restart

root@debian-servidor-fp:/etc/apache2/sites-available# /etc/init.d/apache2 restart
```

Anexo VII.- Despliegue aplicación Opencart.

"El movimiento se demuestra andando."

Diógenes de Sínope

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

1. Descarga y descomprime la aplicación:

- ✓ En la página de descarga de OpenCart(<http://www.opencart.com/index.php?route=download/download>) puedes ver los requisitos para la instalación de Opencart: **Web Server** (preferably Apache) , **PHP** (at least 5.2) , **MySQL** , **Curl** , **Fsock**

- ✓ Descarga el último paquete estable de Opencart de la página web de descarga en `/tmp/pruebas`

```
mkdir /tmp/pruebas
wget -c http://opencart.googlecode.com/files/opencart\_v1.4.9.5.zip (6 MB)
```

- ✓ Descomprime el paquete

```
cd /tmp/pruebas
apt-get install unzip
unzip opencart_v1.4.9.5.zip
```

2. Lee el fichero de instalación `install.txt`.

3. Crea el virtualhost para Opencart:

- ✓ Copia la carpeta upload en el servidor web. Para ello genera en `/etc/apache2/sites-available/` un virtualhost de nombre `tienda-virtual` como el siguiente:

```
<VirtualHost 192.168.200.250:80
    DocumentRoot /var/www/tienda-virtual
    ServerName          ww.tienda-virtual.empresa-proyecto.com           ErrorLog
    /var/log/apache2/error_tienda-virtual.log
    CustomLog var/log/apache2/access_tienda-virtual.log "%h %l %u %t \"%r\" %s %b
    \"%{Referer}i\" %I %O"
</VirtualHost>
```

- ✓ Ahora mueve la carpeta `upload` con el nombre `tienda-virtual` en `/var/www/tienda-virtual`
- ✓ Activa el sitio nuevo tienda-virtual: `a2ensite tienda-virtual`
- ✓ Recarga la configuración de Apache: `/etc/init.d/apache2 reload`
- ✓ Verifica que los siguientes ficheros y carpetas tengan permisos de escritura en `/var/www/tienda-virtual/`: `chmod 0755` ó `0777` para: `image/`, `image/cache/`, `image/data/`, `system/cache/`, `system/logs/`, `download/`, `config.php`, `admin/config.php`

4. Crea la base de datos para OpenCart y el usuario con permisos en la misma:

Asegúrate que posees una base de datos mysql para Opencart y un usuario distinto de root con permisos en la misma:

- ✓ Primero, debes crear una nueva base de datos para tu sitio Opencart:

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

donde:

- ➔ `root` es el usuario administrador de MySQL y por lo tanto tiene los privilegios para crear una base de datos.
- ➔ `db_opencart` es el nombre de la base de datos de opencart que acabas de crear.

MySQL te pide la contraseña del usuario root y luego crea los archivos iniciales de la base de datos.

- ✓ Segundo, creas el usuario con privilegios en la base de datos de nuevo se requiere la contraseña de root-.

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

donde:

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

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

- ✓ Tercero, para activar los nuevos cambios ejecuta:

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

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

5. Visita la página principal de tu Opencart, por ejemplo: <http://www.tienda-virtual.empresa-proyecto.com/>
6. Sigue las instrucciones que aparecen en pantalla.
7. Una vez acabada la instalación borra la carpeta `install`.
8. Puedes ya visitar tu tienda online en: <http://www.tienda-virtual.empresa-proyecto.com/> y tu panel de administración en: <http://www.tienda-virtual.empresa-proyecto.com/admin/>

TEMA 3

Contenido

1.- Protección del servidor de aplicaciones.	2
2.- Despliegue de aplicaciones en Tomcat.	4
2.1.- Creación de una aplicación web.	5
2.2.- Despliegue de una aplicación web.	6
2.3.- Implementar el registro de acceso.	7
2.4.- Sesiones persistentes.	8
2.5.- Configurar Tomcat en cluster.	9
3.- El servidor de aplicaciones JBoss.	11
3.1.- Instalación y configuración básica.	12
3.2.- Despliegue de aplicaciones empresariales.	13
3.3.- Estructura de carpetas de una aplicación empresarial. Archivo EAR.	15
4.- Construcción y despliegue automático con Ant.	16
4.1.- Instalación y configuración de Ant.	17
4.2.- El archivo build.xml.	18
4.3.- El objetivo .jar.	19
4.4.- Despliegue de un archivo WAR.	20
5.- El gestor de aplicaciones Web de Tomcat.	23
5.1.- Configuración del gestor.	23
5.2.- Conexión al gestor de aplicaciones web de Tomcat de forma remota.	24
5.3.- Incluir tareas Ant en Tomcat.	25

Configuración y administración de servidores de aplicaciones.

Caso práctico

En la empresa BK programación, Ada, junto con sus empleados Juan y María se ha reunido para evaluar la posibilidad de configurar uno o dos servidores de aplicaciones para instalar en ellos demos, o **versiones beta** (también denominado "betatest", indica un periodo en el que un software está técnicamente acabado, lo cual significa que no se le añadirán de momento más funciones, y presumiblemente será lo suficientemente estable para trabajar con normalidad. En contraste, la versión alfa, versión anterior a la beta, es más inestable y no está completa), de las aplicaciones que desarrollan, de esta manera los clientes, o potenciales clientes, podrían probar los productos de BK programación antes de adquirirlos.

Como resultado de dicha reunión han concluido que, previo paso a la instalación y puesta en funcionamiento de servidores de aplicaciones, sería muy importante evaluar muchos parámetros que afectarían al correcto funcionamiento de los servidores, además de las necesidades de los mismos. Entre los parámetros a evaluar cabe destacar los siguientes:

- ✓ Seguridad de los servidores de aplicaciones: medidas de seguridad a aplicar para evitar posibles ataques o intrusiones.
- ✓ Dimensionamiento del servidor donde se estudian las necesidades físicas del equipo servidor.
- ✓ Tipo de servidor a instalar, características específicas del software de servidor seleccionado (Tomcat, Jboss, etc.).
- ✓ Despliegue de aplicaciones en el servidor donde habría que establecer qué herramientas se deberían utilizar.
- ✓ Administración de las conexiones remotas a los servidores.
- ✓ Escalabilidad de los servidores, a tener en cuenta en función del número de conexiones simultáneas que se pueden establecer.
- ✓ Herramientas de automatización de tareas en el servidor (Ant, etc.).

Debido a la cantidad de parámetros que hay que administrar para poner en correcto funcionamiento los servidores de aplicaciones, Ada ha decidido que sus empleados se documenten de todos y cada uno de ellos y, si cabe, la posibilidad realizar algún curso de formación sobre la administración de servidores de aplicaciones.

1.- Protección del servidor de aplicaciones.

Caso práctico

Una de las primeras preocupaciones que se encuentran los administradores de servidores es la seguridad y protección de los mismos frente a posibles ataques o accesos incontrolados, por dicha causa, **María** se ha puesto a investigar las opciones a configurar, y herramientas a utilizar, para bloquear las posibles vulnerabilidades de los servidores web junto con los problemas de seguridad en las aplicaciones web.

Un servidor de aplicaciones es, usualmente, un software que proporciona una serie de servicios de aplicación a un número indeterminado de computadoras cliente que acceden a dichos servicios vía web; las principales ventajas de este tipo de tecnología es la centralización y disminución de la complejidad en el desarrollo de aplicaciones, sin embargo las aplicaciones web están así más expuestas a ataques.

Hoy en día existen aplicaciones web para casi todo y que tienen acceso a información muy valiosa como, por ejemplo, números de tarjetas de crédito, cuentas bancarias, historiales médicos, información personal, etc. Con lo cual, representan un objetivo interesante al que atacar; estos ataques se pueden clasificar en base a tres niveles:

- ✓ Ataques a la computadora del usuario (cliente).
- ✓ Ataques al servidor.
- ✓ Ataques al flujo de información que se transmite entre cliente y servidor.

En cada uno de los niveles anteriores es necesario garantizar una seguridad mínima para conseguir la seguridad de todo el proceso. A nivel de usuario éstos deben contar con navegadores y plataformas seguras, libres de virus; a nivel del servidor hay que garantizar que los datos no sean modificados sin autorización (integridad) y que sólo sea distribuida a las personas autorizadas (control de acceso) y, en lo que se refiere al tránsito de la información, ésta no debe ser leída (confidencialidad), modificada o destruida por terceros, al mismo tiempo que hay que garantizar un canal de comunicación fiable que no se interrumpa con relativa facilidad.

Para conseguir aplicaciones web seguras hay que establecer mecanismos que garanticen:

- ✓ **Autenticación:** permite identificar, en todo momento, quién es el usuario que está accediendo. Para conseguirlo existen varios métodos:
 - ➔ *Autenticación básica:* solicitud de usuario y clave.
 - ➔ *Autenticación con certificados.*
 - ▀ `HTTP DIGEST AUTH` (HTTP Autenticación de texto implícita).
 - ▀ `HTTP NTLM AUTH` (HTTP Authentication Microsoft NT Lan Manager).
- ✓ **Autorización:** permite, una vez autenticado, determinar a qué datos y módulos de la aplicación puede acceder el usuario.
- ✓ **Validación de entradas,** ya que se puede manipular el código de validación del lado del cliente.
- ✓ **Inyección de comandos SQL:** técnica para explotar aplicaciones web que no validan la información suministrada por el cliente para generar consultas SQL peligrosas.

Para conseguir aplicaciones web seguras hay que utilizar una serie de mecanismos y herramientas entre las cuales destacamos:

- ✓ Deshabilitación de servicios y cuentas no utilizadas.
- ✓ Actualización del sistema operativo y aplicaciones (**parches** (Cuando se aplica asociado a software, se trata de un conjunto de ficheros adicionales al software original de una herramienta o programa informático. Normalmente sirven para solucionar alguna posible carencia, vulnerabilidad, o defecto de funcionamiento)).
- ✓ Fortaleza en las contraseñas.
- ✓ Utilización de Firewalls.
- ✓ Back-ups periódicas.

- ✓ **Análisis periódico de logs** (*registro oficial de eventos durante un rango de tiempo en particular. Para los profesionales en seguridad informática es usado para registrar datos o información sobre quién, qué, cuándo, dónde y por qué un evento ocurre para un dispositivo en particular o aplicación*).
- ✓ **Verificación periódica de servicios activos.**
- ✓ **Cifrado del tráfico.**
- ✓ **Establecimiento de políticas de seguridad.**

Esta web surge con el objetivo de concienciar y ayudar a la gente para aumentar la seguridad en la red, en ella aparece, de forma actualizada, amenazas, ataques, recomendaciones de seguridad, etc.

<http://www.seguridadenlared.org/>

2.- Despliegue de aplicaciones en Tomcat.

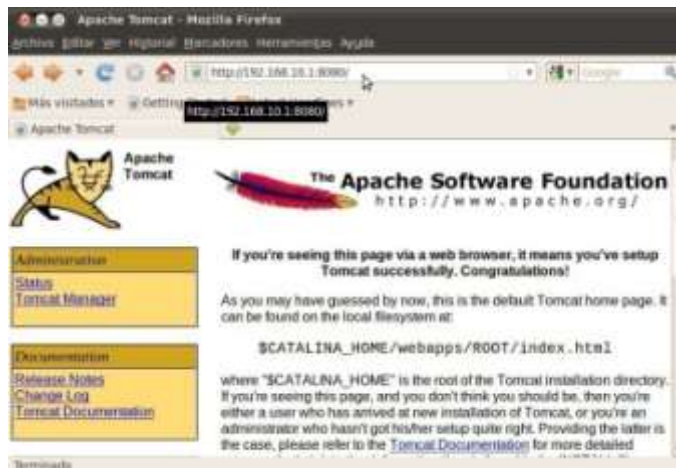
Caso práctico

María, ha montado una máquina Debian 6 con el servidor de aplicaciones Tomcat para que los miembros de **BK programación** puedan desplegar, en dicho servidor, las aplicaciones web que consideren necesarias. Juan ha realizado una primera práctica de despliegue de aplicaciones web y ha documentado todos y cada uno de los pasos que es preciso realizar para que la aplicación web quede totalmente operativa en el servidor, y así cualquier cliente de la empresa pueda disfrutar de la funcionalidad de la aplicación.

Desplegar un servlet consiste en situar una serie de archivos en un contenedor web para que los clientes puedan acceder a su funcionalidad; una aplicación web es un conjunto de servlets, páginas HTML, JSP, clases y otros recursos que se pueden empaquetar de una forma determinada.

Una aplicación web puede ser desplegada en diferentes servidores web manteniendo su funcionalidad y sin ningún tipo de modificación en su código debido a la especificación servlet 2.2. Las aplicaciones web deben organizarse según la siguiente estructura de directorios:

- ✓ **Directorio principal (raíz):** Contendrá los ficheros estáticos (HTML, imágenes, etc...) y JSPs.
 - ➔ Carpeta **WEB-INF**: contiene el fichero "**web.xml**" (descriptor de la aplicación), encargado de configurar la aplicación.
 - Subcarpeta **classes**: contiene los ficheros compilados (servlets, beans).
 - Subcarpeta **lib**: librerías adicionales.
 - ➔ Resto de carpetas para ficheros estáticos.



Una aplicación web puede ser desplegada empleando uno de los siguientes métodos:

- ✓ Por medio de archivos **WAR**.
- ✓ Editando los archivos **web.xml** y **server.xml**, este método es el que se pasa a tratar a continuación.

Los directorios que forman una aplicación compilada suelen ser: **www**, **bin**, **src**, **tomcat**, **gwt-cache**.

La carpeta **www** contiene a su vez una carpeta, con el nombre y ruta del proyecto, que contiene los ficheros que forman la interfaz (HTML, js, css...). La carpeta **bin** contiene las clases de java de la aplicación.

Para desplegar la aplicación en Tomcat se deben realizar los siguientes pasos:

1. Copiar la carpeta contenida en **www** (con el nombre del proyecto) en el directorio **webapps** de Tomcat.
2. Renombrar la nueva carpeta así creada en Tomcat con un nombre más sencillo. Esa será la carpeta de la aplicación en Tomcat.
3. Crear, dentro de dicha carpeta, otra nueva, y darle el nombre **WEB-INF** (respetando las mayúsculas).
4. Crear, dentro de **WEB-INF**, otros dos subdirectorios, llamados **lib** y **classes**.
5. Copiar en **lib** todas las librerías (**.jar**) que necesite la aplicación para su funcionamiento.

6. Copiar el contenido de la carpeta `bin` de la aplicación en el subdirectorío `WEB-INF/classes` del Tomcat.
7. Crear en `WEB-INF` un fichero de texto llamado `web.xml`, con las rutas de los servlets utilizados en la aplicación.
8. Ya puede accederse a la aplicación en el servidor, el modo de hacerlo es poniendo en el navegador la ruta del fichero HTML de entrada, que estará ubicado en la carpeta de la aplicación en Tomcat.

Vamos a partir de una máquina con el sistema operativo Debian 6.0.1 en la cual tenemos el servidor Tomcat corriendo para mostrar el proceso creación y despliegue de aplicaciones. Debido a que pretendemos montar una plataforma `LAMP`, por sus ventajas derivadas de las características del software libre, instalaremos también los siguientes componentes: `MySQL` y `PHP`.

Recordemos, en primer lugar destacar que, para instalar cualquier versión de Tomcat es necesario tener instalado JDK (Kit de desarrollo de Java), ya que el objetivo es que las peticiones a Apache se redirijan a Tomcat empleando un conector proporcionado por Java en este caso.

2.1.- Creación de una aplicación web.

Caso práctico

En la empresa **BK programación**, Juan ha decidido documentar los métodos que resulten más útiles y sencillos a seguir para la creación de una aplicación web, de manera que pueda desplegarse sin ningún tipo de dificultad en el servidor de aplicaciones Tomcat que María ha montado. De esta de manera, los clientes tendrán disponibles todas las funcionalidades de las aplicaciones desarrolladas en la empresa.



El servidor de aplicaciones Tomcat cuenta con una serie de ejemplos, tanto de servlets como de JSP, que sirven de ayuda para aprender a realizar las tareas creación y despliegue de aplicaciones web.

Es muy interesante crear dos variables de entorno: `JAVA_HOME` que indique la ubicación de los archivos binarios de Java y `CATALINA_HOME` que apunta a la ubicación de los scripts (archivo de órdenes o archivo de procesamiento por lotes, es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano) de Tomcat, para ello podemos añadir el siguiente código al archivo `/etc/profile`.

```
CATALINA_HOME=/usr/local/apache-Tomcat-6.0.32/
JAVA_HOME=/usr/lib/jvm/java-6-openjdk/jre/
PATH=$PATH:$JAVA_HOME/bin:$CATALINA_HOME
export PATH JAVA_HOME CATALINA_HOME
```

Actualizamos las variables de entorno mediante el comando:

```
source /etc/profile
```

El lenguaje Javascript se ejecuta del lado del cliente, es un lenguaje interpretado de scripting que no permite acceder a información local del cliente ni puede conectarse a otros equipos de red.

En primer lugar crearemos una carpeta con el nombre que nos interese para identificar la aplicación, en este ejemplo hemos optado por `Aplic_Web` una estructura como la de la siguiente imagen:



La aplicación que pretendemos desarrollar contiene un archivo al que llamaremos `index.jsp` muy sencillo con el siguiente contenido:

```
<html>
  <head><title>C.F. DESARROLLO DE APLICACIONES WEB</title>
  <script language="Javascript">
    function popup(){
      alert("U.T. 3: CONFIGURACION Y ADMINISTRACION DE SERVIDORES DE APLICACIONES");
    }
  </script>
```

```
</head>
<body>
  <h1 align=center>DESPLIEGUE DE APLICACIONES WEB</h1>
  <div align=center>
    <form>
      <input type="button" value="UNIDAD 3" onclick="popup()">
    </form>
  </div>
</body>
</html>
```

para acabar, solamente nos quedaría hacer una copia de la carpeta de nuestra aplicación en `$CATALINA_HOME/webapps` y si, posteriormente desde un navegador, accedemos en local a `http://127.0.0.1:8080/Aplic_Web` tendríamos la aplicación funcionando.

Si el equipo en el que hemos desarrollado la aplicación anterior, y en donde se ha puesto a funcionar, pertenece a una red de computadores y tiene la IP: 192.168.10.1. ¿Podríamos acceder desde otros computadores a la aplicación web? En caso afirmativo, ¿cual sería la URL que deberíamos teclear?

2.2.- Despliegue de una aplicación web.

Uno de los objetivos que se persigue en el momento de desarrollar aplicaciones web, es que éstas puedan ser desplegadas en diferentes servidores web, manteniendo su funcionalidad y sin ninguna modificación de código.

Los WARs simplemente son archivos Java de una aplicación web con una extensión diferente para diferenciarlos de los comunmente usados JARs.

Antes de la especificación Servlet 2.2, era bastante diferente desplegar servlets entre diferentes contenedores de servlets, anteriormente también llamados motores servlet. La especificación 2.2 estandarizó el despliegue entre contenedores, llevando así la portabilidad del código Java un paso más allá.

El método más sencillo para desplegar una aplicación, que sobre todo se utiliza durante la etapa de desarrollo de la misma, es el realizado en el punto anterior, es decir, copiar la carpeta correspondiente a nuestra aplicación en la carpeta `$CATALINA_HOME/webapps`, teniendo en cuenta que la variable `$CATALINA_HOME` es la ruta de los scripts que emplea Tomcat.

Siguendo con la aplicación desarrollada en el punto anterior (`Aplic_Web`), vamos a crear un fichero descriptor del despliegue `web.xml` que es el encargado de describir las características de despliegue de la aplicación.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app 2.5.xsd"
  version="2.5">

  <display-name>Descriptor Aplicacion Aplic_Web</display-name>
  <description>
    Mi primer descriptor web.xml.
  </description>
</web-app>
```

Este archivo lo situaremos en la carpeta `WEB-INF` perteneciente a la aplicación en desarrollo, de forma que la estructura de la carpeta resultante sería el mostrado en esta imagen:



Una vez consideramos terminada nuestra aplicación web podremos generar el archivo `.WAR` perteneciente a la aplicación, para ello podemos aplicar los siguientes comandos:

`#javac -d WEB-INF/classes *.java` este comando tiene como finalidad la compilación de las clases Java de nuestra aplicación.

`#jar cvf Aplic_Web.war WEB-INF` para crear el archivo `.WAR`.

Una vez hecho lo anterior podríamos acceder vía web a: `http://127.0.0.1:8080` y, en el apartado "Administration", accedemos a la opción "Tomcat Manager" y desde la ventana resultante tenemos las opciones que aparecen en la siguiente imagen para desplegar el archivo `.WAR`:



Esta web muestra, de forma amplia, el funcionamiento, configuración, instalación, administración, etc. del servidor de aplicaciones Tomcat, donde también podemos encontrar cómo desplegar aplicaciones.

<http://tomcat.apache.org/>

2.3.- Implementar el registro de acceso.

Caso práctico

Sobre las aplicaciones web que han sido desarrolladas por la empresa BK programación y que ya están accesibles para sus clientes, se ha considerado realizar de algún modo un seguimiento, de manera que se pueda comprobar los accesos que han tenido, en qué momento y qué recursos son más demandados; para ello **Juan**, junto con **María**, han configurado el servidor Tomcat para poder adaptar los logs, de manera que puedan obtener información sobre los accesos a sus aplicaciones.

Para conseguir obtener y poder configurar los registros de acceso a un servidor de aplicaciones Tomcat, como es nuestro caso, empezaremos hablando de las válvulas de registro de acceso de Tomcat, ya que será el método que emplearemos.

Las válvulas del Tomcat son una tecnología introducida a partir de Tomcat 4 que permite asociar una instancia de una clase Java a un contenedor "Catalina". Esta configuración permite que la clase asociada actúe como un pre-procesador de las peticiones. Estas clases se llaman válvulas, y deben implementar la interfaz "org.apache.catalina.Valve" interface o extender de la clase "org.apache.catalina.valves.ValveBase". Las válvulas son propias de Tomcat y no pueden ser usadas en otros contenedores de servlet.

Las válvulas disponibles son:

- ✓ **Access Log Valve**: está implementada por la clase "org.apache.catalina.valves.AccessLogValve". Crea ficheros de `log` para rastrear el acceso a la información de los clientes, registrando información como, por ejemplo, actividad de la sesión del usuario, información de la autenticación del usuario, entre otras. Por ejemplo, el siguiente código:

```
<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
prefix="localhost_access_log." suffix=".txt" pattern="common"/>
```

Indicará que los logs de acceso se almacenarán en el directorio `$CATALINA_HOME/logs` y los archivos de `log` tendrán la nomenclatura con prefijo: `localhost_access_log` y sufijo `.txt` probablemente entre sufijo y prefijo se añadirá la fecha en la que se crea dicho archivo.

- ✓ **Remote Address Filter**: permite comparar la dirección IP del cliente con una o más expresiones regulares y, como resultado de ello, denegar o bien permitir la solicitud presentada por el cliente. Un ejemplo de uso podría ser el siguiente:

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve" deny="127.*">
```

en donde a los clientes que tengan una IP que comienza por 127 se les va a denegar la solicitud.

- ✓ **Remote Host Filter**: es muy parecido al anterior pero con la diferencia que permite comparar por nombre de equipo en lugar de IP.

```
<Valve className="org.apache.catalina.valves.RemoteHostValve" deny="pc_fp.*">
```

- ✓ **Request Dumper**: es una herramienta de depuración que escribe en el **log** el detalle de cada petición realizada.

```
<Valve className="org.apache.catalina.valves.RequestDumperValve"/>
```

Cualquier acceso a **localhost:8080** tendrá asociado una serie de entradas en los logs.

- ✓ **Single Sign On**: cuando queremos que los usuarios puedan identificarse en cualquier aplicación de nuestro virtual host, y que su identidad sea reconocida por cualquier aplicación que esté en ese host.

```
<Valve className="org.apache.catalina.authenticator.SingleSignOn"/>
```

Podemos implementar los ejemplos anteriores en **\$CATALINA_HOME/conf/server.xml**, de este modo dichos cambios afectarán a cualquier aplicación desplegada en el servidor.

"Los sistemas nuevos generan problemas nuevos."

Ley de Murphy

2.4.- Sesiones persistentes.

Caso práctico

Sobre las aplicaciones web que han sido desarrolladas por la empresa **BK programación** y que ya están accesibles para sus clientes, **Ada** ha solicitado a **Juan** y **María** cómo poder, de algún modo, garantizar las sesiones, estableciendo en la configuración de Tomcat sesiones persistentes que aseguren sesiones fiables a las aplicaciones en caso de caída del servidor o de pérdida de conexión.



Las sesiones activas por parte de clientes a aplicaciones web alojadas en servidores web Tomcat, por defecto, están configuradas para mantenerse en caso de posibles pérdidas de conexión con el servidor o posibles reinicios del mismo; a pesar de todo ello es posible establecer un control mayor sobre dichas sesiones.

Por lo que respecta a las sesiones inactivas (pero todavía no caducadas) es posible configurarlas de forma que se almacenen en disco liberando, como consecuencia de ello, los recursos de memoria asociados. Al parar Tomcat las sesiones activas se vuelcan a disco de manera que, al volver a arrancarlo, se podrán restaurar.

Las sesiones con un tiempo de vida que supere un límite se copian automáticamente a disco por seguridad para evitar posibles bloqueos de sesión.

Para configurar las sesiones persistentes tendremos que gestionar el elemento **<Manager>** como un subelemento de **<Context>** de forma que podemos actuar a dos niveles en función de si pretendemos que la configuración establecida se aplique a todas las aplicaciones del servidor o a una aplicación concreta.

Si configuramos las sesiones persistentes de forma global tenemos que manipular el archivo **/conf/context.xml**, mientras que si queremos configurar las sesiones a nivel local a una aplicación web determinada tendríamos que adaptar el archivo **<CATALINA_HOME>/conf/context.xml** correspondiente a la aplicación.

Un ejemplo de configuración podría ser el siguiente (se emplean comentarios para explicar cada uno de los parámetros):

```
<Context>
  <!-- classname especifica la clase del servidor que implementa el gestor, es recomendable
  utilizar el org.apache.catalina.session.PersistentManager -->
  <Manager className="org.apache.catalina.session.PersistentManager">
    <!--saveOnRestart=true para indicar que se guarden todas las sesiones al reiniciar el
    servidor -->
    saveOnRestart="true"
```

```

<!--maxActiveSession cuando se supera el límite aquí establecido se comienzan a enviar a
disco las nuevas sesiones. Se establece un valor -1 para indicar ilimitadas sesiones-->
maxActiveSession="3"
<!--minIdleSwap establece el número mínimo de segundos que transcurren antes de que una
sesión pueda copiarse al disco duro -->
minIdleSwap="0"
<!--maxIdleSwap indica el número máximo de segundos que transcurren antes de que una
sesión pueda copiarse al disco duro -->
maxIdleSwap="60"
<!--maxIdleBackup para indicar el número de segundos desde que una sesión estuvo activa
por última vez hasta que se envíe al disco. La sesión no es eliminada de memoria. Permite
restauración de la sesión en caso de caída del servidor. -->
maxIdleBackup="5">
<!--Store indica cómo y donde almacenar la sesión, están disponibles las siguientes
implementaciones: org.apache.catalina.session.FileStore y
org.apache.catalina.session.JDBCStore -->
<Store className="org.apache.catalina.session.FileStore"/>
</Manager>
</Context>

```

2.5.- Configurar Tomcat en cluster.

Caso práctico

Una vez que se han puesto las aplicaciones que **BK programación** ha terminado de desarrollar en el servidor de Tomcat, se ha observado un incremento exponencial en el número de clientes que acceden a los servicios de dichas aplicaciones, motivo por el cual se ha pensado en establecer algún tipo de cluster sobre el servidor para poder atender eficientemente a las peticiones de los usuarios.

Debido al incremento de las aplicaciones web, la escalabilidad y la disponibilidad se transforma en un recurso transcendental para garantizar el servicio eficiente a los clientes web; la implementación de `clustering` para los servidores de aplicaciones web es una solución eficaz y relativamente sencilla.

La implementación de `clustering` con Tomcat provee:

- ✓ **Escalabilidad:** si para ofrecer un servicio solicitado por un cliente web, un servidor web invierte un tiempo "T", para satisfacer un número elevado de servicios, cabe preguntarse cuánto es el tiempo invertido. La respuesta ideal a la cuestión anterior sería que el tiempo invertido fuese lo más próximo posible al tiempo invertido en una única petición, es decir lo más cercano posible a "T".
Para ello existen dos posibles soluciones: escalado horizontal (implica el incremento del número de servidores), escalado vertical (implica el incremento de los recursos del propio servidor).
- ✓ **Alta disponibilidad:** Tomcat provee `failover`; en el motor del servidor existen dos tipos de `failover` provistos por `clustering`:
 - ➔ `Request-level failover`: Si un servidor cae, los siguientes requerimientos se redireccionarán a otros servidores activos.
 - ➔ `Session-level failover`: En el caso de que un servidor deje de dar servicio, otro servidor del cluster debería proporcionar la sesión a los clientes consiguiendo reducir al mínimo la pérdida de conexión, ello implica replicar la sesión en el cluster en la nueva máquina en el mínimo tiempo posible.
- ✓ **Balanceo de carga:** establecer un método de reparto de la carga de peticiones entre los servidores del cluster, de modo que se minimice el tiempo de respuesta a las solicitudes de los clientes; se consigue empleando algoritmos de distribución de carga.

Las soluciones de `clustering` típicas ofrecen un paradigma de servidor que consiste en ofrecer un sistema basado en ejecución distribuida, a pesar de que existe limitación respecto a la escalabilidad, podemos observar el esquema de Jakarta Tomcat server engine works.

<http://tomcat.apache.org/tomcat-6.0-doc/cluster-howto.html>

El conector del servidor de cluster recibe la petición desde los clientes, y el procesador del servidor de cluster encapsula las peticiones en los objetos "`RequestEntry`" y los escribe en `JavaSpace`. El

conector del `Worker del cluster` toma dichas peticiones y el procesador del `worker del cluster` resuelve las peticiones.

Para establecer una configuración de cluster en Tomcat podremos seguir los siguientes pasos:

- ✓ Todos los atributos de sesion deben implementar `java.io.Serializable`.
- ✓ Descomentar el elemento `cluster` en `server.xml`.
- ✓ Descomentar `Valve` (`ReplicationValve`) en `server.xml`
- ✓ Si las múltiples instancias de Tomcat están en la misma máquina el parámetro `tcpListenPort` tiene que ser único para cada una de las instancias.
- ✓ Establecer en el archivo `web.xml` el elemento `<distributable/>` o bien definirlo de forma `<Context distributable="true"/>`.
- ✓ El atributo `jvmRoutes` tiene que estar definido en el "Engine" `<Engine name="Catalina" jvmRoute="nodeX">` estableciendo su valor al nombre de la instancia en el cluster.
- ✓ Sincronizar la hora de todos los nodos con un servicio `NTP`.
- ✓ Configurar el parámetro `loadbalancer` en modo `"sticky session"`.

Esta web documenta los pasos a seguir para montar un cluster horizontal formado por dos servidores con una instancia de Tomcat corriendo en cada uno de ellos.

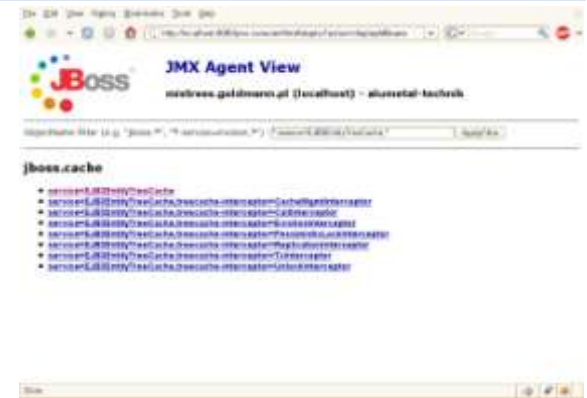
http://es.wikibooks.org/wiki/Cluster_Tomcat_HOWTO

3.- El servidor de aplicaciones JBoss.

Caso práctico

Los empleados de **BK programación** han oído hablar de la importancia del servidor de aplicaciones JBoss, ya que se trata de un servidor de código abierto orientado a aplicaciones e-bussines; siendo, por todo ello, una plataforma que ha adquirido una gran importancia en el mercado, tanto de particulares como de grandes empresas, y que merece la pena estudiar su comportamiento para poder implantar.

Del mismo modo, es interesante establecer el modo a operar para la instalación y configuración del servidor JBoss, así como de todos y cada uno de los pasos necesarios para poder realizar el despliegue de aplicaciones.



El servidor JBOSS es un proyecto de código abierto, con el que se consigue un servidor de aplicaciones basado en J2EE, e implementado al 100 % en Java.

Mientras el Tomcat es un `Servlet Container`, JBoss es un `Application Server`, que soporta funciones de J2EE, las más importantes son los EJB's y el clustering. Tomcat por sí solo simplemente sirve para JSP's y servlets.

JBoss es un servidor de aplicaciones basado en Java mientras que Tomcat es un contenedor de servlets.

Uno de los rasgos más importantes de JBoss es su apoyo a la implementación "*en caliente*". Lo que significa es que implementar un nuevo EJB es tan simple como copiar el archivo correspondiente en el directorio correspondiente. Si esto se hace mientras el bean ya está cargado, JBOSS lo descarga automáticamente, y entonces carga la nueva versión.

JBoss está compuesto por dos partes: un "Servlet Engine" y un "EJB Engine", dentro del "Servlet Engine" se ejecutan exclusivamente las clásicas aplicaciones de un servidor (JSP's y Servlets), mientras el "EJB Engine(Container)" es reservado para aplicaciones desarrolladas alrededor de EJB's o Enterprise Java Bean's.

JBoss es el primer servidor de aplicaciones de código abierto preparado para la producción y certificado J2EE 1.4, ofrece una plataforma de alto rendimiento para aplicaciones de e-business. Combinando una arquitectura orientada a servicios revolucionaria con una licencia de código abierto, JBoss puede ser descargado, utilizado, incrustado y distribuido sin restricciones por la licencia. Por este motivo, es la plataforma más popular de middleware para desarrolladores, vendedores independientes de software y, también, para grandes empresas.

Entre las características destacadas de JBoss destacamos las siguientes:

- ✓ Producto de licencia de código abierto sin coste adicional.
- ✓ Cumple los estándares.
- ✓ Confiable a nivel empresa.
- ✓ Incrustable, orientado a arquitectura de servicios.
- ✓ Flexibilidad consistente.
- ✓ Servicios de middleware para cualquier objeto de Java.

El creador de la primera versión de JBoss fué Marc Fleury quién fundó una empresa de servicios llamada JBoss Inc., adquirida en 2006 por Red Hat.

Por ejemplo, el juego de los Sims online utiliza JBoss así como otros juegos multiusuario.

¿Cuáles de las siguientes son características del servidor de aplicaciones JBoss?

- ☒ Es de código abierto.
- ☒ Está implementado en su totalidad en Java.
- ☒ Es únicamente un "EJB Container".
- ☐ Funciona únicamente en servidores Microsoft Windows.
- ☒ Está orientado a arquitectura de servicios.

3.1.- Instalación y configuración básica.

Vamos a partir de una máquina Debian 6.0.1 Squeeze, en la que realizaremos el proceso de instalación y configuración básica del servidor JBoss y que vamos a estructurar en los siguientes pasos:

1. **Descarga e instalación de Java Development Kit (JDK):** En primer lugar, destacar que, para instalar cualquier versión de JBoss, es necesario tener instalado JDK (Kit de desarrollo de Java), ya que se trata de un servidor de aplicaciones basado e implementado al 100 % en Java, como se ha dicho anteriormente, y puede ser ejecutado en cualquier sistema en el que se encuentre operativo un JDK en su versión 1.5 o superior. **Empezamos buscando el paquete de Java que nos puede interesar. Con el siguiente comando obtendríamos la lista del entorno Java debido a que Debian proporciona varias implementaciones, cada uno de estos paquetes tiene un entorno de desarrollo (JDK) y un tiempo de ejecución conocido (JRE o Java Virtual Machines JVM):**

```
#aptitude search "?provides(java-runtime)"
```

2. **Luego realizamos la instalación de esos paquetes empleando el comando siguiente, ello no supondrá ningún tipo de complicación ya que se encuentran accesibles desde el repositorio que tenemos por defecto:**

```
#apt-get install default-jre openjdk-6-jdk
```

Para instalar la versión JDK de Sun (ahora ya de Oracle) en Debian 6 (Squeeze) tenemos que agregar un repositorio, para ello editamos el archivo `sources.list` mediante el siguiente comando:

```
#nano /etc/apt/sources.list y agregamos la siguiente línea:
```

```
deb http://ftp.ch.debian.org/debian/ squeeze main non-free
```

guardamos el archivo y, a continuación, ejecutamos el comando: `#aptitude update` ó `#apt-get update` y, una vez se realice la actualización, instalaremos los siguientes paquetes Java de Sun mediante el siguiente comando:

```
#aptitude install sun-java6-jre sun-java6-jdk
```

y, una vez instalado, lo seleccionamos mediante:

```
#update-alternatives -config java
```

que mostrará las opciones disponibles y seleccionaremos el número de opción que contiene la máquina virtual de Java de **Sun/Oracle**.

3. **Descarga e instalación de JBoss Application Server 6.0:** Se pueden descargar las distintas versiones del servidor JBoss del siguiente enlace, en este caso hemos decidido descargar el paquete `jboss-as-distribution-6.0.0.Final.zip`.

<http://www.jboss.org/jbossas/downloads/>

Para proceder a su instalación simplemente nos situamos en la carpeta donde deseemos instalarlo, en nuestro caso lo haremos en `"/usr/local/jboss/"` y, una vez allí, descomprimos el paquete mediante:

```
#unzip jboss-as-distribution-6.0.0.Final.zip
```

4. **Crear el usuario de JBoss que posee y dirige JBoss:** Es recomendable ejecutar JBoss con una cuenta de usuario no root, con privilegios mínimos. Para ello crearemos un grupo JBoss y un

usuario llamado JBoss al que pondremos contraseña y agregaremos al grupo creado; podemos hacerlo del siguiente modo:

```
#groupadd jboss
#useradd -s /bin/bash -g jboss jboss
#passwd jboss
#usermod -d /usr/local/jboss/jboss-6.0.0.Final/ jboss
```

5. **Establecer las variables de entorno `JAVA_HOME` y `JBOSS_HOME`:** Estas variables son interesantes para indicar las rutas donde se ha instalado Java y JBoss. Estas rutas serán empleadas en los archivos de configuración de dichas aplicaciones, para ello simplemente agregamos, en nuestro caso, el siguiente contenido al archivo `/etc/profile`:

```
JAVA_HOME=/usr/lib/jvm/java-6-sun/jre
JBOSS_HOME=/usr/local/jboss/jboss-6.0.0.Final
PATH=$PATH:$JAVA_HOME/bin:$JBOSS_HOME/bin
export PATH JAVA_HOME JBOSS_HOME
```

posteriormente ejecutaríamos `#source /etc/profile` para que el sistema recoja el contenido de las variables creadas sin necesidad de reiniciar el equipo.

6. **Crear un script para automatizar JBoss con los parámetros/funcionalidades "start/stop/restart" y configurar JBoss para que se ejecute como un servicio:** Existe un script llamado `"jboss_init_redhat.sh"` en la carpeta `$JBOSS_HOME/bin` que nos va a servir para crear el script que administre el servidor de JBoss; para ello copiamos dicho script a `/etc/init.d` y lo renombramos a `jboss` `#cp $JBOSS_HOME/bin/jboss_init_redhat.sh /etc/init.d/jboss` luego editamos el fichero copiado, en donde tenemos que reemplazar las siguientes líneas adaptándolas a nuestra configuración, en nuestro caso:

```
JBOSS_HOME=${JBOSS_HOME:-"/usr/local/jboss/jboss-6.0.0.Final"}
JAVAPATH=${JAVAPATH:-"/usr/java/jdk1.6.0_24"}
```

y añadir la línea `JBOSS_HOST="0.0.0.0"` permitiendo así acceder a JBoss desde cualquier IP.

7. **Acceder a la consola de administración de JBoss:** Asegurarse que JBoss se ha iniciado y de que conseguimos acceder a la consola JBoss desde las siguientes direcciones: `http://ip equipo:8080` y también `http://localhost:8080` si se accede desde el propio servidor.
8. **Cambiar la contraseña de administrador de JBoss:** Editamos para tal fin el archivo `"/usr/local/jboss-6.0.0.Final/server/default/conf/props/jmx-console-users.properties"` en donde introducimos la contraseña que decidamos a continuación de `admin=`.



Para realizar la instalación y configuración básica del servidor JBoss 6.0, debemos seguir, de forma secuencial, todos y cada uno de los siguientes pasos:

1. **Descarga e instalación de Java Development Kit (JDK),** requisito indispensable para poder funcionar el servidor.
2. **Descarga e instalación de JBoss Application Server 6.0:** Se trata de un software libre.
3. **Crear el usuario de JBoss, que posee y dirige JBoss,** debido a que es recomendable no trabajar con el usuario root de una máquina, para administrar un servidor web.
4. **Establecer las variables de entorno `JAVA_HOME` y `JBOSS_HOME`:** agilizarán el proceso de configuración del servidor ya que son empleadas por muchos ficheros de configuración.
5. **Crear un script para automatizar JBoss con los parámetros/funcionalidades "start/stop/restart" y configurar JBoss para que se ejecute como un servicio,** aunque no es necesario, pero resulta más cómodo arrancar el servidor como un servicio más.
6. **Acceder a la consola de administración de JBoss,** desde donde podemos administrar el servidor desde un entorno web.

3.2.- Despliegue de aplicaciones empresariales.

JBoss, adquirida por Red Hat en 2006, es líder del mercado en ofrecer soluciones middleware Open Source de nivel empresarial. JBoss Middleware Enterprise está compuesto por un conjunto de plataformas y frameworks certificados y soportados con el nivel de calidad profesional que ofrece Red Hat.

Las soluciones de JBoss Enterprise Middleware se distribuyen vía las "JBoss Subscription", que incluyen el software certificado y actualizaciones, herramientas de gestión, políticas de mantenimiento a largo plazo y un soporte técnico líder en la industria. Las suscripciones están disponibles tanto para uso en producción como para desarrollo.

Las plataformas JBoss Enterprise, que se detallan a continuación, integran múltiples proyectos y componentes, los más populares de la comunidad JBoss.org en distribuciones certificadas, estables y seguras, con una única vía de parches y actualizaciones.

- ✓ **JBoss Enterprise Application Platform.**
 - ➔ Diseñada para construir, desplegar y albergar servicios, y aplicaciones Java.
 - ➔ Integra el servidor de aplicaciones JBoss AS en cluster, un sistema de mapeo y persistencia O/R y, además, un potente framework para la construcción de aplicaciones de nueva generación Web 2.0.
- ✓ **JBoss Enterprise Web Platform.**
 - ➔ Para aplicaciones web en Java y aplicaciones ricas basadas en Internet (RIA).
- ✓ **JBoss Enterprise Web Server.**
 - ➔ Una única solución empresarial basada en open source para servidores web basados en tecnología Apache y Tomcat.
- ✓ **JBoss Enterprise Portal Platform.**
 - ➔ Diseñado para construir y desplegar portales para la interacción de usuario SOA y la presentación personalizada.
 - ➔ Integra un framework de portal, funcionalidades CMS con workflow y JBoss Enterprise Application Platform.
- ✓ **JBoss Enterprise SOA Platform.**
 - ➔ Integra aplicaciones y orquesta servicios para automatizar procesos de negocio en una arquitectura orientada a servicios.
 - ➔ Se construye sobre un bus de servicios e integra un motor de reglas, automatización de proceso de negocio y JBoss Enterprise Application Platform.
- ✓ **JBoss Enterprise BRMS.**
 - ➔ Un sistema basado en open source empresarial para administrar reglas de negocio que facilita el desarrollo, el acceso y la gestión de los cambios de políticas y reglas de negocio.
- ✓ **JBoss Enterprise Data Services Platform.**
 - ➔ Acaba con la desconexión entre los diversos orígenes de datos empresariales que existen y los innovadores formatos de datos que requieren los nuevos proyectos, aplicaciones y arquitecturas.

La estructura de una aplicación web en su forma más sencilla, debe contener la siguiente estructura de directorios:

```
META-INF/  
  manifest.mf  
WEB-INF/  
  classes/  
  src/  
  lib/  
  web.xml
```

conteniendo la carpeta **META-INF**, en aplicaciones **.jar**, el archivo **manifest.mf**, que contiene la lista de contenidos de la aplicación, y que son generados al momento de crearla. El directorio **WEB-INF** contiene todos los archivos necesarios para ejecutar la aplicación, y estructura su contenido en las carpetas **classes** que contiene las clases compiladas para la aplicación, **lib** con las librerías necesarias para la aplicación y **src**, con el código fuente de la aplicación.

Una vez que la aplicación JEE está correctamente construida, se realiza el empaquetado con el comando:

```
#jar cvf nombre_aplicacion.jar carpetas/ficheros_a_empaquetar
```

Una vez tenemos la aplicación `.jar` para desplegarla, únicamente la copiamos a la carpeta "`$JBOSS_HOME/server/default/deploy`" y el propio JBoss nos dará un mensaje similar a `deploy, ctxPath = / nombre_aplicacion`, lo que quiere decir que la aplicación ha sido desplegada correctamente; esto se conoce como despliegue en caliente.

3.3.- Estructura de carpetas de una aplicación empresarial. Archivo EAR.

En el mundo Java EE tenemos tres posibles tipos de aplicaciones: aplicaciones web, objetos distribuidos EJBs y aplicaciones empresariales, que no son más que un conjunto de las dos anteriores aplicaciones.

Una aplicación empresarial Java EE (archivo `.EAR`) es un conjunto de módulos, siendo un módulo una aplicación web completa (empaquetada en un archivo `.war`) o conjunto de objetos distribuidos EJBs (empaquetados en un archivo `.jar`).

Podemos resumir que la estructura del archivo `EAR` es:

- ✓ `/*.war`: Archivos war.
- ✓ `/*.jar`: Archivos (ejb) jar.
- ✓ `/META-INF/application.xml`: Descriptor de despliegue del módulo `EAR`, en donde se dan de alta y se declaran el nombre y descripción de la aplicación que se despliega, y los diferentes módulos web y `EJB` que forman la aplicación.

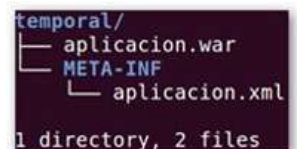
Vamos a suponer una estructura lo más sencilla posible para una aplicación web como la siguiente, y que es la que constituye el archivo "`aplicacion.war`":



donde observamos una página estática "`index.html`" y un descriptor del despliegue "`web.xml`", a partir de esta estructura pretendemos construir nuestro propio archivo `EAR` que contendrá un solo archivo `WAR` con una página `HTML` estática.

Una vez situados en la carpeta "`aplicacion`", mediante el comando `#jar cvf aplicacion.war *` generaremos el archivo `.WAR` correspondiente a la aplicación; podremos comprobar que se trata de un formato similar a los archivos `.zip` probando a abrirlo con un programa compresor.

Para construir el archivo `.EAR`, como mínimo, tendremos que crear un descriptor de despliegue al que llamaremos "`aplicacion.xml`", para ello creamos una carpeta llamada "`temporal`" en donde situamos el archivo "`aplicacion.war`"; en la misma ruta creamos una carpeta llamada "`META-INF`" donde vamos a crear el descriptor; quedando la estructura del siguiente modo:



Nos situamos dentro de la carpeta "`temporal`" y creamos el archivo `.ear` mediante el comando:

```
#jar cvf aplicacion.ear *
```

y tendremos así el archivo `.ear` correspondiente a la aplicación creada.

¿Cuáles de las siguientes afirmaciones son correctas?

- ☐ Un archivo `.war` puede estar formado por varios archivos `.ear`.
- ☒ Un archivo `.ear` puede estar formado por varios archivos `.war`.
- ☒ El comando `#jar cvf` permite generar archivos `.war`.
- ☒ El comando `#jar cvf` permite generar archivos `.ear`.
- ☒ Un archivo `.ear` puede contener archivos `.jar`.

4.- Construcción y despliegue automático con Ant.

Caso práctico

En la empresa BK Programación, para agilizar el proceso de construcción de aplicaciones web, han pensado en la automatización del proceso con la ayuda de la herramienta Ant que se emplea para la realización de tareas mecánicas y repetitivas, normalmente durante la fase de compilación y construcción.

A la hora de implantar dicha herramienta se han propuesto, además, documentar el procedimiento de instalación, configuración y puesta en funcionamiento de dicha herramienta.

ANT (siglas de "Another Neat Tool", en español "Otra Herramienta Pura", que en inglés significan "hormiga") fue creado por James Duncan Davidson mientras realizaba la transformación del proyecto **Solar** de Sun Microsystems en código abierto (concretamente la implementación del motor JSP/Servlet de Sun, que luego se llamaría Jakarta Tomcat).

Apache Ant es una herramienta usada en programación para la realización de tareas mecánicas y repetitivas, normalmente se centra en la fase de compilación y construcción (build). Es similar al "`make`" empleado en Linux, pero desarrollado en Java; posee la ventaja de no depender de los comandos `shell` de cada sistema operativo, ya que se basa en archivos de configuración XML y clases Java, siendo idónea como solución multi-plataforma.

Podemos destacar aspectos y/o funciones de las que **Ant** se va a ocupar:

- ✓ Compilación.
- ✓ Generación de documentación.
- ✓ Empaquetamiento.
- ✓ Ejecución, etc.

Es utilizado en la mayoría de los proyectos de desarrollo de Java y funciona a partir de un script de ensamblado, en formato XML (`build.xml`) que posteriormente se explicará con más detalle; además es fácilmente extensible e integrable con muchas herramientas empleadas por los desarrolladores, por ejemplo el editor Jedit o el IDE Netbeans.

Trabajar sin **Ant** implica una compilación manual de todos los ficheros `.java` (sin un control de los que han sido modificados y de los que no) incluir los `classpath` relativos adecuados, tener los ficheros `.class` mezclados con el código fuente...; sin embargo con **Ant**, en el fondo, no estás más que automatizando tareas, para que, al final, con un solo comando, puedas compilar desde cero tu proyecto, ejecutar pruebas unitarias, generar la documentación, empaquetar el programa...

Como limitaciones a tener en cuenta:

- ✓ Al ser una herramienta basada en XML, los archivos Ant deben ser escritos en XML.
- ✓ La mayoría de las antiguas herramientas, como `<javac>`, `<exec>` y `<java>` tienen malas configuraciones por defecto, valores para opciones que no son coherentes con las tareas más recientes.
- ✓ Cuando se expanden las propiedades en una cadena o un elemento de texto, las propiedades no definidas no son planteadas como error, sino que se dejan como una referencia sin expandir.

Para trabajar con **Ant** se necesita:

- ✓ JDK en versión 1.4 o superior, ya que Ant no deja de ser una aplicación Java.
- ✓ Un parser XML. Da igual cual, si se ha bajado la versión binaria de Ant no hay por qué preocuparse, porque ya incluye uno.

ANT (Another Neat Tool)

¿Qué es?

Es una herramienta que permite automatizar el proceso de ensamblado de

	aplicaciones web Ensamblado = construcción + despliegue Similar a la herramienta make de linux
¿Para qué sirve?	Se ocupa de: ✓ Compilación ✓ Generación de documentación ✓ Empaquetamiento ✓ Ejecución...
Ventajas	Automatiza tareas, para que al final con un solo comando: ✓ Puedes compilar desde cero tu proyecto, ✓ ejecutar pruebas unitarias, ✓ generar la documentación, ✓ empaquetar el programa... No depende de los comandos Shell de cada sistema operativo, ya que se basa en archivos XML y clases Java, siendo idónea como solución multi-plataforma.
¿Cómo funciona?	Funciona a partir de un script de ensamblado en formato XML llamado build.xml, definido en base a proyecto, targets y tasks ✓ Proyecto ➔ Uno por archivo y contiene targets ✓ Target ➔ Con un nombre y dependencias hacia otros targets ➔ Contiene un conjunto de tasks ✓ Tasks ➔ Operaciones básicas (javac, java, jar, etc)

En esta página podemos encontrar toda la información que nos pueda interesar para comenzar a trabajar con la herramienta Ant.

<http://ant.apache.org/>

4.1.- Instalación y configuración de Ant.

Vamos a partir de una máquina con el sistema operativo Debian 6.0.1 en donde realizaremos la instalación de **Ant**, en primer lugar comprobamos si tenemos instalado Java, podemos hacerlo empleando el siguiente comando:

```
#java -version
```

recordemos que, como requisito para la instalación de Ant, es imprescindible una versión JDK 1.4 ó superior.

Posteriormente procederemos a la descargar del paquete binario de Ant, que podemos descargarlo de la siguiente forma:

```
#wget http://ant.apache.org/bindownload.cgi/apache-ant-1.8.2-bin.tar.gz
```

y una vez hemos descargado el archivo binario lo descomprimos empleando la instrucción:

```
#tar -zxvf apache-ant-1.8.2-bin.tar.gz
```

luego movemos la carpeta "`apache-ant-1.8.2`" creada a "`/usr/local`".

Lo único que falta es crear la variable `ANT_HOME` y actualizar la variable `PATH`.

- ✓ `ANT_HOME`: Indica el directorio raíz de instalación de Ant, de acuerdo a las instrucciones anteriores esta ruta sería: `/usr/local/apache-ant-1.8.2`.
- ✓ `PATH`: Define la ruta de acceso para los binarios del sistema; la modificación de esta variable permite acceder a los ejecutables de Ant desde cualquier directorio.

Podemos hacerlo agregando al archivo `/etc/profile` el siguiente contenido:

```
ANT_HOME=/usr/local/apache-ant-1.8.2/  
PATH=$PATH:$ANT_HOME/bin
```

y luego, para que el sistema recoja los cambios realizados, empleamos el comando: `#source /etc/profile`.

Para comprobar que **ant** se ha instalado correctamente desde una consola de shell ejecutamos el comando siguiente: `#ant` y deberíamos obtener un mensaje similar a:

```
Buildfile: build.xml does not exist!  
Build failed
```

con lo que la herramienta **ant** estaría correctamente instalada y configurada para desempeñar su función en nuestra máquina.

En el siguiente vídeo podemos ver que se muestra cómo realizar la instalación del paquete Ant en un equipo con sistema operativo Microsoft Windows 7.

http://www.youtube.com/watch?feature=player_embedded&v=bcY4ZF1jt4o

La primera parte del vídeo nos explica cómo descargar el paquete **Ant** desde su web de descarga, en dicha página podemos observar varios formatos y revisiones para el paquete Ant y, en nuestro caso, se selecciona el .zip y se realiza la descarga del mismo.

Luego se extrae el archivo .zip descargado en la carpeta que nos interese, en este caso "c:\kwit\apache-ant-1.8.2", en donde vemos la estructura de carpetas que Ant contiene, entre otras, *bin*, *docs*, *etc*, *lib*...

Una vez instalado el paquete se pasa a configurar las variables de entorno de la aplicación; para ello se accede a "Panel de control, configuraciones avanzadas y variables de entorno", se selecciona la variable *PATH* y se actualiza su valor con la ruta donde se ha instalado Ant seguido de \bin, es decir, para este caso concreto sería: "c:\kwit\apache-ant-1.8.2\bin" y, por último, se abre un intérprete de comandos empleando el comando **cmd** y, mediante la orden **ant -version** se comprueba que la instalación ha sido correcta y que queda la aplicación operativa.

4.2.- El archivo build.xml.

Como hemos dicho, **Ant** se basa en ficheros XML, normalmente configuramos el trabajo a hacer con nuestra aplicación en un fichero llamado `build.xml`, así que vamos a ver algunas de las etiquetas con las que podemos formar el contenido de este archivo.

```
<?xml version="1.0" ?>  
- <wxSIPUA>  
  <Username>Hubert</Username>  
  <Password>lala</Password>  
</wxSIPUA>
```

- ✓ **project**: Este es el elemento raíz del fichero XML y, como tal, solamente puede haber uno en todo el fichero, el que se corresponde a nuestra aplicación Java.
- ✓ **target**: Un `target` u objetivo es un conjunto de tareas que queremos aplicar a nuestra aplicación en algún momento. Se puede hacer que unos objetivos dependan de otros, de forma que eso lo trate Ant automáticamente.
- ✓ **task**: Un `task` o tarea es un código ejecutable que aplicaremos a nuestra aplicación, y que puede contener distintas propiedades (como por ejemplo el classpath). **Ant** incluye ya muchas básicas, como compilación y eliminación de ficheros temporales, pero podemos extender este mecanismo si nos hace falta. Luego veremos algunas de las disponibles.
- ✓ **property**: Una propiedad o `property` es, simplemente, algún parámetro (en forma de par nombre-valor) que necesitamos para procesar nuestra aplicación, como el nombre del compilador, etc. Ant incluye ya las más básicas, como son `BaseDir` para el directorio base de nuestro proyecto, `ant.file` para el path absoluto del fichero `build.xml`, y `ant.java.version` para la versión de la JVM.

Pasamos a ver un simple ejemplo de archivo `build.xml`:

```
<?xml version="1.0"?>

<project name="ProbandoAnt" default="compilar" basedir=".">
  <!-- propiedades globales del proyecto -->
  <property name="fuente" value="." />
  <property name="destino" value="classes" />

  <target name="compilar">
    <javac srcdir="${fuente}" destdir="${destino}" />
  </target>
</project>
```

Este sencillo fichero requiere poca explicación, simplemente declaramos el proyecto indicando, la acción a realizar por defecto (`default="compilar"`), e indicamos que el directorio base es el actual (`basedir="."`).

Después indicamos en sendas etiquetas `property` los directorios de origen y de destino (`property name="fuente" value="."` y `property name="destino" value="classes"`).

Por último declaramos un `target` llamado `compilar`, que es el que hemos declarado como por defecto.

En este objetivo tenemos una única tarea, la de compilación `javac`, a la que por medio de los atributos `srcdir` y `destdir` le indicamos los directorios fuente y destino, que recogemos de las propiedades anteriormente declaradas con `${fuente}` y `${destino}`.

Lo único que nos queda es compilar nuestro código, así que, simplemente, estando situados en el directorio donde tenemos nuestro `build.xml`, desde una ventana de MS-DOS o terminal GNU/Linux, podemos hacer:

```
# [PATH_TO_ANT] ant
```

Esto funciona así porque hemos declarado `compilar` como el objetivo por defecto, aunque podría ser otro así que por regla general pondríamos:

```
# [PATH_TO_ANT] ant nombre_objetivo
```

Ant se basa en ficheros XML, normalmente configuramos el trabajo a hacer con nuestra aplicación en un fichero llamado `build.xml`.

4.3.- El objetivo `.jar`.

Para explicar el contenido de este apartado lo vamos a hacer mediante un ejemplo. En primer lugar creamos un fichero `build.xml` en la raíz de nuestro proyecto y definimos su nombre:

```
<project name="Proyecto">
</project>
```

Ant, al igual que otras herramientas de construcción, se basa en el concepto de objetivos o `targets` cuya definición engloba tanto las dependencias previas como los pasos a seguir para conseguirlo.

Vamos a comenzar definiendo un objetivo de preparación llamado `init` que será el encargado de crear un directorio `classes` donde guardaremos los ficheros `.class` resultantes de la compilación y el directorio `build` para el `.jar` final. Para ello basta incluir dentro de `<project>` las siguientes líneas:

```
<target name="init">
  <mkdir dir="classes" />
  <mkdir dir="build" />
</target>
```

Como podemos ver los objetivos se delimitan con etiquetas `<target>` y un nombre. Dentro de ellos se enumeran los pasos que se han de seguir para alcanzar el objetivo, en este caso ha de crear directorios.

Si queremos alcanzar el objetivo `init` basta con realizar:

```
#ant init
Buildfile: build.xml

init:
  [mkdir] Created dir: /home/profesor/proyecto/classes
  [mkdir] Created dir: /home/profesor/proyecto/build
BUILD SUCCESSFUL
Total time: 0 seconds
```

Es hora de compilar nuestro proyecto, vamos a definir el objetivo `compile`. Ahora bien, la compilación depende de la creación del directorio "`classes`" que se realiza en el objetivo anterior. Con esto en cuenta basta con incluir:

```
<target name="compile" depends="init">
  <javac srcdir="src" destdir="classes" />
</target>
```

La dependencia se fija en la declaración del `target` de tal manera que se garantiza su cumplimiento antes de comenzarla. Nuestro código está en el directorio "`src`" y el resultado de la compilación se lleva al directorio "`classes`".

Importante notar que esta vez estamos usando `<javac>` esto es lo que **Ant** denomina tarea. Hay muchas tareas predefinidas.

Con nuestro proyecto compilado vamos a generar el `.jar` que distribuiremos haciendo uso de un nuevo objetivo llamado `build`.

```
<target name="build" depends="compile">
  <jar destfile="build/proyecto.jar" basedir="classes" />
</target>
```

Comprobamos que hay una dependencia de `compile` y se utiliza la tarea `jar` que se encarga de empaquetar todo el contenido del directorio `classes` en el fichero `proyecto.jar`.

Finalmente incluiremos un nuevo objetivo para limpiar todo el entorno, el objetivo `clean`:

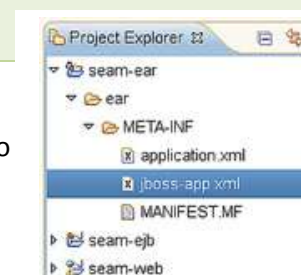
```
<target name="clean">
  <delete dir="classes" />
  <delete dir="build" />
</target>
```

Elimina los directorios de trabajo dejando el entorno limpio del proceso de compilación. Resumiendo nuestro fichero `build.xml` es:

```
<project name="Proyecto">
  <target name="init">
    <mkdir dir="classes" />
    <mkdir dir="build" />
  </target>
  <target name="compile" depends="init">
    <javac srcdir="src" destdir="classes" />
  </target>
  <target name="build" depends="compile">
    <jar destfile="build/proyecto.jar" basedir="classes" />
  </target>
  <target name="clean">
    <delete dir="classes" />
    <delete dir="build" />
  </target>
</project>
```

4.4.- Despliegue de un archivo WAR.

En la arquitectura Java EE, los componentes web y los ficheros con contenido estático, como imágenes, son llamados **recursos web**.



Un **módulo web** es la más pequeña unidad de un recurso web que se pueda utilizar y desplegar. Un módulo web Java EE corresponde con una **aplicación web**, como se define en la especificación de Java Servlet.

Además de los componentes web y los recursos web, un módulo web puede contener otros ficheros:

- ✓ Clases utilitarias del lado del servidor (beans *(componente software que tiene la particularidad de ser reutilizable y así evitar la tediosa tarea de programar los distintos componentes uno a uno)* para bases de datos, carritos de compras y demás). A menudo estas clases cumplen con la arquitectura `JavaBeans`.
- ✓ Clases del lado del cliente (applets *(componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo un navegador web)* y clases utilitarias).

Un módulo web tiene una estructura específica. El directorio más alto de la jerarquía de directorios de un módulo web es el **raíz de documento** de la aplicación. Es donde las páginas JSP, clases y archivos del **lado del cliente**, y los recursos estáticos como imágenes, son almacenados.

El directorio raíz de los documentos contiene un subdirectorio llamado `WEB-INF`, que contiene los siguientes ficheros y directorios:

- ✓ `web.xml`: El descriptor de despliegue de aplicación.
- ✓ `classes`: Un directorio que contiene las clases del lado del servidor: componentes Servlets, clases utilitarias y JavaBean.
- ✓ `tags`: Un directorio que contiene ficheros de etiquetas, que son implementaciones de librerías de etiquetas.
- ✓ `lib`: Un directorio que contiene los archivos JAR de las librerías llamadas por las clases del lado del servidor.

Un módulo web debe ser empaquetado en un `WAR` en ciertos escenarios de despliegue y cuando se quiera distribuir el módulo web. Se empaqueta un módulo web en un `WAR` ejecutando el comando `jar` en un directorio ubicado en el formato de un módulo, utilizando la utilidad `Ant` o utilizando la herramienta IDE de su elección.

Un módulo web puede ser desplegado como una estructura de ficheros sin empaquetar o puede ser empaquetado en un fichero `JAR` conocido como un archivo web (`WAR`). Dado que el contenido y uso de los ficheros `WAR` difieren de aquellos ficheros `JAR`, el nombre del fichero `WAR` utiliza una extensión `.WAR`. El módulo web descrito es portátil, se puede desplegar en cualquier contenedor web que cumpla con la especificación Java Servlet.

Para desplegar un `WAR` en un servidor de aplicaciones, el fichero debe contener un **descriptor de despliegue** en tiempo de ejecución. El descriptor de despliegue es un fichero XML que contiene información como el contexto raíz de la aplicación web y la relación de los nombres portátiles de los recursos de aplicación a los recursos del servidor de aplicación.

Existen una serie de tareas para `Ant` que podemos utilizar para la gestión de aplicaciones, entre las cuales destacamos:

- ✓ `<deploy>`: Despliega una aplicación web.
- ✓ `<start>`: Inicia una aplicación web.
- ✓ `<stop>`: Para una aplicación.
- ✓ `<undeploy>`: Repliega (desinstala) una aplicación.
- ✓ `<trycatch>`: Evita que falle un build aunque falle alguna tarea.

Se pueden emplear diversos tipos de servidores de aplicaciones web junto con la herramienta `Ant`, por ejemplo JBoss o Tomcat.

Para desplegar un WAR con la herramienta Ant, abrimos una ventana de terminal o línea de comando en el directorio donde se ha construido y empaquetado el WAR y ejecutamos **ant deploy**.

Rellena los huecos con los conceptos adecuados:

En la arquitectura Java EE, los componentes web y los ficheros con contenido estático, como imágenes, son llamados **recursos web**. Un **módulo web** es la más pequeña unidad de un recurso web que se pueda utilizar y desplegar.

El **descriptor de despliegue** es un fichero XML que contiene información como el contexto raíz de la aplicación web y la relación de los nombres portátiles de los recursos de aplicación a los recursos del servidor de aplicación.

Para desplegar un WAR con la herramienta Ant, abrimos una ventana de terminal o línea de comando en el directorio donde se ha construido y empaquetado el WAR y ejecutamos **ant deploy**.

5.- El gestor de aplicaciones Web de Tomcat.

Caso práctico

En la empresa BK programación disponen de un servidor de aplicaciones web Tomcat. Debido a las opciones que éste proporciona han decidido profundizar en el funcionamiento de éste, pero centrándose en la administración de aplicaciones a desplegar desde la interfaz web que Tomcat proporciona, el "Gestor de Aplicaciones Web de Tomcat".

Una vez arrancado en el equipo servidor el Tomcat mediante el script "catalina.sh" que se encuentra en la carpeta /bin del directorio de instalación de Tomcat, en nuestro caso

"/usr/local/apache-Tomcat-6.0.32/", desde un navegador podremos acceder a Tomcat mediante la URL:

- ✓ <http://localhost:8080> si accedemos desde la propia máquina en la que está corriendo Tomcat.
- ✓ http://ip_servidor:8080 si accedemos desde cualquier otra máquina de la red.



Mediante el enlace "Tomcat Manager" accedemos al gestor de aplicaciones Web de Tomcat. Esta página permite desplegar un proyecto contenido en un fichero de extensión **war**, como ya hemos visto en el punto "[2.2 Despliegue de una aplicación web](#)" de este tema, o simplemente copiar la carpeta que contiene de la aplicación a la carpeta **webapps** que se encuentra en el directorio de instalación de Tomcat.

Vamos al **Tomcat Manager** y allí podremos ver un listado de las aplicaciones web que hay disponibles en el servidor. Podemos comprobar, en nuestro caso, que si tenemos en la carpeta **usr/local/apache-Tomcat-6.0.32/webapps/** la carpeta de la aplicación "Aplic Web" que desarrollamos al principio de este tema, ya se mostraría en el listado que el gestor de aplicaciones de Tomcat nos ofrece, o simplemente accediendo desde un navegador a la URL: http://ip_servidor:8080/nombre_aplicacion (en el caso genérico), para nuestro caso podemos probar con http://localhost:8080/Aplic_Web.

Si estás trabajando como administrador de sistemas en una empresa (supongamos que es **BK programación**), en la que eres el encargado de administrar, entre otras, un máquina en la que hay un servidor de aplicaciones web Tomcat.

¿Cómo solicitarías a los desarrolladores de aplicaciones que te enviasen las aplicaciones a desplegar en dicho servidor?

5.1.- Configuración del gestor.

Todos los ficheros de configuración se encuentran en la carpeta **conf** en la ruta de instalación de Tomcat. Esa ruta la referenciamos anteriormente con la variable de entorno **CATALINA_HOME**, es decir **%CATALINA_HOME%/conf**. En esta ruta encontramos una carpeta denominada **catalina/localhost/** en donde se almacena la configuración web del Tomcat en dos archivos .xml: **host-manager.xml** y **manager.xml**.

Para realizar la administración del servidor desde el entorno web, instalaremos un paquete adicional, Tomcat6-admin, podemos hacerlo mediante la instrucción:

```
#apt-get install Tomcat6-admin
```

y, para acceder a la administración, es necesario crear el rol "manager" y un usuario con dicho rol, para ello podemos seguir el siguiente procedimiento:

- ✓ Editamos el archivo de usuarios de Tomcat: `#nano $CATALINA_HOME/conf/Tomcat-users.xml`.
- ✓ Añadimos las líneas estableciendo un contenido para y :


```
<role rolename="manager">
<user username="usuario" password="clave" roles="manager"/>
```
- ✓ Reiniciamos el Tomcat y, a través de la URL `http://ip_servidor:8080/manager/html`, podemos desinstalar, recargar e instalar aplicaciones.
- ✓ Para habilitar el host-manager tendríamos que realizar los mismos pasos pero estableciendo el rol `admin`, y desde `http://ip_servidor:8080/host-manager/html` tendríamos el servicio operativo.



Podemos asegurar Tomcat estableciendo que se permita el acceso a este contexto únicamente a las direcciones IP de los equipos desde los que operan los administradores, esto lo podemos configurar en el archivo: `$CATALINA_HOME/work/Catalina/localhost/manager/context.xml`.

```
<Context path="/manager" privileged="true" antiResourceLocking="false" docBase="/opt/apache-tomcat6/webapps/manager">
  <Valve className="org.apache.catalina.valves.RemoteAddrValve" allow="127.0.0.1,
  direccion ip1, direccion ip2" />
</Context>
```

Para acceder a la administración de Tomcat es necesario crear el rol "manager", "admin" y usuario/s con dicho rol, para ello es necesario editar el archivo de usuarios de Tomcat `$CATALINA_HOME/conf/Tomcat-users.xml`.

5.2.- Conexión al gestor de aplicaciones web de Tomcat de forma remota.

Un servidor Apache-Tomcat consta de 3 componentes principales:

- ✓ **Catalina**: es el contenedor de Servlet de Tomcat. Implementa las especificaciones de Sun para servlets y Java Server Pages (JSP).
- ✓ **Coyote**: es el conector HTTP que soporta el protocolo HTTP1.1 para el servidor web o para el contenedor de aplicaciones.
Coyote escucha las conexiones entrantes en un puerto TCPdeterminado y redirige las peticiones al motor Tomcat para así procesar las peticiones y mandar una respuesta de vuelta al cliente.
- ✓ **Jasper**: es el motor JSP de Tomcat; compila las páginas JSP en código java en servlets que puedan ser manejados por Catalina.

En tiempo de ejecución, cualquier cambio en un archivo JSP Jasper lo detecta y lo recompila.

Los modos de operación de Tomcat pueden ser:

1. Servidor de aplicaciones:
 - ✓ Tomcat necesita un servidor que actúe como frontend (Apache, IIS...).
 - ✓ El contenido estático es servido por el frontend.
 - ✓ Las peticiones a servlets y JSPs son redirigidas a Tomcat por el servidor web.

- ✓ Recibe peticiones en protocolos específicos como AJP que son enviados por el frontend.
- 2. Standalone:
 - ✓ No hay un servidor web que actúe de frontend.
 - ✓ Todos los contenidos son servidos por Tomcat.
 - ✓ Recibe peticiones HTTP.

Los conectores son los componentes que proporcionan la interfaz externa al servidor, concretamente el conector HTTP1.1 basado en `Coyote` es el conector por defecto para Tomcat. Los conectores se definen en el archivo:

`$CATALINA_HOME/conf/server.xml`, aquí tenemos un ejemplo:

```
<Connector port="8080"
  protocol="HTTP/1.1"
  maxThreads="150"
  connectionTimeout="2000"
  redirectPort="8443"/>
```

debido a establecer medidas de seguridad para conexiones web al servidor, podremos configurar para un conector HTTP/1.1 con SSL lo siguiente:

```
<Connector port="8080"
  protocol="HTTP/1.1"
  maxThreads="150"
  scheme="https"
  secure="true"
  clientAuth="false"
  sslProtocol="TLS"/>
```

en donde vemos que se han establecido los atributos `scheme` para el protocolo, y `secure` para establecer que se trata de un conector SSL.

Rellena los huecos con los conceptos adecuados:

Un servidor Apache-Tomcat consta de 3 componentes principales:

- ✓ **Catalina**: es el contenedor de Servlet de Tomcat. Implementa las especificaciones de Sun para servlets y Java Server Pages (JSP).
- ✓ **Coyote**: es el conector HTTP que soporta el protocolo HTTP1.1 para el servidor web o para el contenedor de aplicaciones.
Coyote escucha las conexiones entrantes en un puerto **TCP** determinado y redirige las peticiones al motor **Tomcat** para así procesar las peticiones y mandar una respuesta de vuelta al cliente.
- ✓ **Jasper**: Es el motor JSP de **Tomcat**; compila las páginas JSP en código java en servlets que puedan ser manejados por **Catalina**.

Los modos de operación de Tomcat pueden ser **Servidor de aplicaciones** y **Standalone**.

5.3.- Incluir tareas Ant en Tomcat.

Como ya hemos visto anteriormente, `Ant` es una herramienta de construcción de software que permite automatizar tareas repetitivas en el proceso de compilación, enlazado, despliegue, etc.

Tomcat define una serie de librerías que le permiten automatizar tareas como el despliegue y repliegue de aplicaciones web, mediante `Ant`.

Para integrar las dos herramientas anteriores podemos seguir las siguientes operaciones:

- ✓ Descargar `Ant`.
- ✓ Descomprimir el fichero.
- ✓ Configurar las variables de entorno `ANT_HOME` para que apunte a la raíz de la distribución.
- ✓ Configurar la variable `PATH` para añadir la ruta hasta el directorio `<ANT_HOME>/bin`.

- ✓ Copiar el fichero `<Tomcat_HOME>/lib/catalina-ant.jar` en `<ANT_HOME>/lib`.

Para instalar una aplicación web, se le indica a `Tomcat Manager` que un nuevo contexto está disponible, empleando para ello el comando `#ant install` que funciona tanto con archivos `.WAR` como si se indica la ruta al directorio de la aplicación no empaquetada. Es necesario tener en cuenta que el comando anterior no implica un despliegue permanente; si se reinicia Tomcat las aplicaciones previamente instaladas no van a estar disponibles.

Despliegue permanente de aplicaciones web:

- ✓ Sólo funciona con archivos `*.WAR`.
- ✓ No se pueden desplegar directorios no empaquetados.
- ✓ Se sube `*.WAR` al Tomcat y se arranca.
- ✓ Permite el despliegue remoto.
- ✓ Un contenedor web remoto no puede acceder al directorio de la máquina local.

El comando `#ant deploy` se emplea para el despliegue permanente de las aplicaciones, y para ello es necesario:

- ✓ Que el `Tomcat Manager` se esté ejecutando en la localización especificada por el atributo `url`.
- ✓ El despliegue de una aplicación en el contexto especificado por el atributo `path` y la localización contenida en los archivos de la aplicación web especificada con el atributo `war`.

Podemos establecer el siguiente ejemplo:

```
<deploy url="http://localhost:8080/manager"
  path="mywebapp"
  war="file:/path/to/mywebapp.war"
  username="username" password="password" />
```

El archivo `build.xml` de una aplicación llamada "Hola" para `"ant deploy"` podría ser el siguiente:

```
<target name="deploy" description="Deploy web application" depends="build">
  <deploy url="${url}" username="${username}"
    password="${password}"
    path="${path}" war="file:${build}/${example}.war"/>
</target>
<taskdef name="deploy" classname="org.apache.catalina.ant.DeployTask" />
<property name="url" value="http://localhost:8080/manager" />
<property name="path" value="/${example}" />
<property name="example" value="hola" />
```

TEMA 4

Contenido

1.- Servicio de transferencia de ficheros.....	2
1.1.- ¿Cómo funciona?	3
1.2.- Cliente FTP.	4
1.3.- Tipos de usuarios.....	5
1.4.- Modos de conexión del cliente.	6
1.5.- Tipos de transferencia de archivos.	7
1.6.- Establecer permisos en ftp.	8
1.7.- Servicio de transferencia de archivos en modo texto.	9
1.7.1.- Comandos ftp.....	11
1.8.- Servicio de transferencia de archivos en modo gráfico.	12
1.9.- Servicio de transferencia de archivos desde el navegador.	14
1.10.- Asegurando el servicio de transferencia de archivos.	15
1.11.- El servicio de transferencia de archivos en el proceso de despliegue de una aplicación Web.	16
2.- Instalación del servidor proftpd.	18
2.1.- Configuración de proftpd.	19
2.2.- Configurar el servidor como ftp privado.	21
2.3.- Configurar el servidor como ftp privado y anónimo.....	21
2.4.- Configurar el servidor como ftp anónimo.	22
2.5.- Configurar el servidor ftp con múltiples dominios.	23
2.6.- Virtualhosts basados en nombre.	24
2.7.- Virtualhosts basados en IP.....	25
2.8.- Cuotas de disco para los usuarios (I).....	26
2.8.1.- Cuotas de disco para los usuarios (II).....	27
2.9.- Acceso seguro mediante TLS.	28
Anexo I - PAM.....	33
¿Qué es PAM?.....	33
Grupos de gestión.....	33
Arquitectura	34
Configuración.....	35
Enfoques de la organización de la configuración.....	35
Reglas	35
Servicio.....	35
Tipo	36
Control	36
Ruta.....	37
Argumentos	37
Algunos módulos disponibles	37
pam console.so.....	37
Grupo	37
pam cracklib.so.....	38
Grupo	38
pam deny.so	38
Grupo	38
pam env.so.....	38
Grupo	38
pam limits.so	39
Grupo	39
pam nologin.so	39
Grupo	39
pam permit.so	39
Grupo	39
pam rootok.so	40
Grupo	40
pam securetty.so	40
Grupo	40
pam stack.so	40
Grupo	40
pam wheel.so	40
Grupo	40
pam xauth.so	40
Grupo	40
Ejemplos de configuración	40
login.....	41
passwd	42
su	42
other	43
Valores devueltos por los módulos de PAM.....	43
authentication	43
account	44
password.....	44
session.....	44
Anexo II - proftpd.conf.....	45
Anexo III - tls.conf	48
Anexo IV - tls2.conf	49

Instalación y administración de servidores FTP

Caso práctico

La empresa BK Programación se ha dedicado últimamente a la creación de entornos web en servidores dedicados y compartidos. Siempre se ha decantado para la instalación, administración y configuración de servidores de software libre. Y es por ello que varias empresas se han puesto en contacto con BK Programación para contratarles sus servicios. Una de estas empresas con las que trabaja BK Programación tuvo un problema subiendo un archivo a su dominio (Nombre por el cual se reconoce a un grupo de dispositivos o equipos conectados a la red. Éstos pueden ser nombres locales, no existentes en Internet, pero, en general, son utilizados para su uso en Internet, por ejemplo: debian.org), por lo cual se ha recibido una llamada dirigida a soporte técnico en BK Programación. En ésta se mantuvo la siguiente conversación:

- ✓ Hola, buenos días. BK Programación, ¿en qué puedo ayudarle?
- ✓ Hola, buenos días. Tengo un problema en el servidor de nuestra página, no puedo subir un archivo de vídeo.
- ✓ ¿No le aparece la opción de subida o, en medio del proceso, se le corta la conexión?
- ✓ Sí, soy capaz de empezar a subir el archivo pero, pasado un tiempo, el proceso se corta.
- ✓ ¿Cuánto pesa el archivo? Es decir, ¿qué tamaño posee?
- ✓ No sé, espere un momento... —pasado un tiempo—, sí..., mire, el archivo ocupa 300 MB.
- ✓ Vale, parece claro, tal como está procediendo hasta ahora no podrá subir el archivo, debido a la limitación establecida en el servidor web para la subida de archivos, por lo tanto debe utilizar su cuenta ftp para la transferencia de archivos.
- ✓ Y eso, ¿cómo procedo? ¿está estipulado en el contrato?
- ✓ Sí, no se preocupe. El contrato estándar ya establece una cuenta ftp por dominio, pero eso sí, dependiendo del contrato poseerá una cuota de disco u otra. En cuanto al método para proceder, aparece explicado en nuestra página web, paso por paso, en la documentación que podrá encontrar en la pestaña descargas.
- ✓ Pues, poseo el contrato estándar.
- ✓ Bien, entonces posee una cuota de 2 GB.
- ✓ Vale, gracias, entonces ¿cuándo podré contar con la cuenta ftp para subir el archivo?
- ✓ Ya la tiene operativa, solamente debe seguir los pasos del documento que le he comentado. Si tiene cualquier problema no dude en contactar de nuevo con nosotros.
- ✓ De acuerdo, muchas gracias. Hasta luego.
- ✓ Hasta luego.

Este tipo de incidencias son típicas en BK Programación, por lo cual Ada, la directora de BK Programación ha establecido un protocolo de actuación según el tipo de incidencia. Las incidencias primero serán atendidas por una unidad de servicio telefónico, en caso de que no se encuentre la solución dentro de las posibles será escalada a su correspondiente área técnica, así las incidencias de servidores serán escaladas a María, las de programación a Juan y éstos derivarán la incidencia a un técnico de la empresa. En el caso de la incidencia por llamada telefónica anterior la solución ya existía dentro de las posibles con lo cual la incidencia no fue escalada.

Las posibles soluciones fueron proporcionadas por el personal responsable para cada área. Así como la incidencia que hablamos era sobre servidores fue propuesta la solución por María. Además, en este caso, María había estudiado varios servidores ftp y se decantó por la versatilidad, funcionalidad y seguridad del servidor ftp ProFTPD.

1.- Servicio de transferencia de ficheros.

Caso práctico

María, sabía que, llegado el momento, las empresas a las que darían soporte web necesitarían subir archivos a sus dominios, por lo cual necesitarían una alternativa a la aplicación web destinada para tal fin: un servicio ftp. Así realizó un estudio sobre servidores ftp y se decantó por la versatilidad, funcionalidad y seguridad del servidor ftp **ProFTPD**. En ese estudio quería llegar a saber del servidor ftp lo siguiente:

1. ¿Cómo funciona?
2. Posibilidades de autenticación y control de acceso.
3. Seguridad. ¿Es posible cifrar la transferencia de archivos?
4. ¿Permite cuotas de disco?
5. ¿Permite cuotas de subida y bajada de archivos?
6. ¿Qué clientes ftp soporta?

Pero antes de ponerlo en producción necesitaba probarlo, es por eso que construyó el siguiente escenario de pruebas, similar al escenario de producción para una empresa que ofrezca sus servicios web por medio de la infraestructura proporcionada por BK Programación y totalmente transparente al cliente final:

- ✓ Sistema Operativo: Debian GNU/Linux 6.0
- ✓ Servidor FTP: ProFTPD
- ✓ Configuración de Red:
 - Servidor FTP: 192.168.200.250
 - Cliente de pruebas, desde donde se lanza el cliente ftp: 192.168.200.100

Hoy en día encontramos muchísima información en Internet, es más, en muchas ocasiones cuando buscamos una determinada información es muy probable que tengamos que filtrarla, ya que encontramos demasiada. Pero, una vez encontrada ¿la podemos guardar? ¿y descargar? Y si es así ¿cómo fue subida?

Normalmente para subir archivos en Internet, ya sean de texto, imágenes, vídeo... hubo que emplear algún método de transferencia de archivos para ubicarlos.

Uno de los métodos más empleados como servicio de transferencia de archivos se realiza mediante el servicio ftp. Éste utiliza el protocolo FTP empleando la arquitectura cliente-servidor. Así el servidor ftp esperará peticiones para transferir los archivos y el cliente ftp, ya sea por terminal o de modo gráfico, realizará esas peticiones.

Uno de los principales problemas, a pesar de ser uno de los métodos más utilizados del protocolo FTP es la no seguridad de la información, esto es, la transferencia tiene lugar sin cifrar la información transferida. Este no sólo es un problema del protocolo FTP sino de muchos de los protocolos utilizados en Internet, puesto que en el comienzo de Internet no se preveía su expansión actual y no se pensaba en asegurar la información mediante cifrado, sino simplemente asegurar el buen funcionamiento. Hoy en día existen extensiones sobre el protocolo FTP que aseguran el cifrado en la transferencia, como FTPS, empleando el cifrado SSL/TLS (*protocolos criptográficos que proporcionan comunicaciones seguras por una red, comúnmente Internet*).

No confundir FTPS con SFTP, ya que este último es implementado con otro servicio, el servicio SSH, y es utilizado para conexiones remotas seguras a través de un terminal de comandos.

En el siguiente enlace, página web del RFC (*serie de documentos en los que se detalla prácticamente todo lo relacionado con la tecnología de la que se sirve Internet: protocolos, recomendaciones, comunicaciones...*) 959 sobre FTP, puedes encontrar traducido el estándar RFC sobre FTP.

<http://www.rfc-es.org/rfc/rfc0959-es.txt>

En el siguiente enlace, página web del RFC 4251 sobre SSH, puedes encontrar el estándar RFC sobre SSH.

<http://www.ietf.org/rfc/rfc4251.txt>

1.1.- ¿Cómo funciona?



El protocolo FTP emplea una arquitectura cliente/servidor, siendo el cliente ftp quien solicita la transferencia de archivos y el servidor ftp quien ofrece los archivos. Pertenecen a la familia de protocolos de red TCP (*uno de los protocolos fundamentales en Internet. Garantiza que los datos serán entregados en su destino sin errores y, una vez recogidos, puestos en el mismo orden en que se transmitieron*) y por lo tanto es un protocolo orientado a conexión, esto es, el cliente ftp necesita establecer una conexión con el servidor para empezar la transferencia de ficheros. Si no se establece la conexión ésta no tiene lugar.

Puesto que FTP es un protocolo que no utiliza una autenticación de usuarios y contraseña cifrada, se considera un protocolo inseguro y no se debería utilizar a menos que sea absolutamente necesario. Verás que existen otras alternativas al FTP, como por ejemplo el protocolo FTPS, para mantener comunicaciones cifradas. Aún así, el protocolo FTP está muy extendido en Internet ya que a menudo los usuarios necesitan transferir archivos entre máquinas sin importar la seguridad.

El protocolo FTP requiere de dos puertos (*números utilizados en las comunicaciones cliente/servidor, en transmisiones TCP o UDP comprendidos entre 1 y 65535, que indican por donde tiene lugar la conexión con un servidor. Están estandarizados, esto es, un servidor suele estar activo siempre por definición en un puerto determinado, pero éste puede que sea modificado en la configuración del servidor. Por ejemplo un servidor web suele esperar en el puerto TCP 80*) TCP en el servidor para su funcionamiento, a diferencia de la mayoría de los protocolos utilizados en Internet que solamente requieren un puerto en el servidor. Un puerto es necesario para establecer el control de la conexión y otro se utiliza para el control de la transmisión, es decir, un puerto se utiliza para establecer la conexión entre el cliente y el servidor y otro para la transferencia de datos.

Los puertos TCP del servidor en cuestión, suelen ser el 21 para el control de la conexión y otro a determinar según el modo de conexión: podría ser el 20 o incluso uno mayor de 1024. Hay que tener en cuenta que estos puertos pueden ser modificados en la configuración del servidor, así no es obligatorio que los puertos 21 y 20 sean los asignados al servidor FTP, pero sí son los que éste maneja por defecto. El puerto 21 también es conocido como puerto de comandos y el puerto 20 como puerto de datos.

La ventaja que supone utilizar el protocolo FTP se basa en su alto rendimiento y sencillez, que lo hacen una opción conveniente para la transferencia de archivos a través de Internet.

A través de un cliente ftp descargas un archivo a tu equipo desde el servidor ftp: ¿cuáles de las siguientes afirmaciones son correctas teniendo en cuenta que el archivo puede descargarse sin problemas?



El servidor ftp posee dos puertos TCP: uno para el control de la transmisión y otro para la transferencia de datos.

- ☐ El servidor ftp posee siempre los puertos TCP 21 y 20: el 21 para el control de la transmisión y el 20 para la transferencia de datos.
 - ☒ El servidor ftp posee los puertos TCP 21 y 20: el 21 para el control de la transmisión y el 20 para la transferencia de datos.
 - ☐ El servidor ftp, configurado por defecto, posee el puerto TCP 21 válido para el control de la transmisión y para la transferencia de datos.
- Siempre deben existir dos puertos TCP, uno para el control de la transmisión y otro para la transferencia de datos, éstos en un servidor ftp suelen ser el 21 y el 20 respectivamente, pero pueden ser modificados.*

1.2.- Cliente FTP.

Lo importante es no dejar de hacerse preguntas.

Albert Einstein

Para poder establecer una conexión con el protocolo FTP son necesarias dos partes: un servidor y un cliente.

Existen múltiples tipos de clientes ftp, desde clientes en terminal de comandos, como `ftp` o `lftp`, clientes gráficos como `gftp` o `FileZilla`, hasta un cliente ftp en los navegadores mediante `ftp://`

¿Cuál elegir? Pues, como todo, depende:

- ✓ ¿Conoces la consola ftp? Si te manejas con soltura en la consola ftp puedes pensar en un cliente ftp de comandos que permita utilizar la tecla "tabulador" después de escribir unos caracteres para complementar los nombres de archivos.
- ✓ ¿Cuál es el uso que necesitas? ¿Para qué lo vas a utilizar? A lo mejor solamente quieres visitar un servidor ftp y descargar un archivo sin tener que andar instalando nuevos programas. En este caso puedes utilizar el cliente ftp del navegador, `ftp://`
- ✓ ¿Quieres reanudar la conexión en caso de corte en la misma? En este caso mejor un cliente tipo gráfico.
- ✓ ¿Deseas facilidad de manejo? Un cliente terminal de comandos suele ser menos interactivo que uno gráfico, debes saber manejarte con comandos en la consola ftp, mientras que en un cliente gráfico puedes manejarte a través de clics del ratón. Los clientes gráficos suelen ser más amigables y por lo tanto más utilizados.
- ✓ ¿Qué tipo de conexión quieres establecer? ¿cifrada? ¿no cifrada? Dependiendo del tipo de conexión debes emplear un cliente u otro, ya que no todos los clientes ftp permiten conexiones cifradas.
- ✓ ¿Deseas recordar conexiones (sitios (Plantilla de configuración para recordar perfiles de configuración a servidores FTP)))? Pues lo mismo, no todos los clientes ftp lo permiten.

Un cliente ftp muy recomendable es el cliente gráfico ftp FileZilla, ya que posee las siguientes características:

- ✓ Fácil de usar.
- ✓ Soporta FTP, FTP sobre SSL/TLS (FTPS) y SFTP.
- ✓ Compatibilidad con múltiples plataformas: se ejecuta en Windows, Linux, BSD, Mac OS X y más.
- ✓ Soporte ipv6.
- ✓ Disponible en varios idiomas.
- ✓ Soporta y reanuda la transferencia de archivos de gran tamaño (mayores de 4 GB).
- ✓ Interfaz de usuario con pestañas.
- ✓ Potente administrador de sitios y cola de transferencia.
- ✓ Marcadores.
- ✓ Arrastrar y soltar.
- ✓ Permite configurar límites de velocidad de transferencia.
- ✓ Nombre de filtros.
- ✓ Directorio de comparación.

- ✓ Asistente de configuración de la red.
- ✓ Edición de archivos remoto.
- ✓ Automantenimiento de la conexión.
- ✓ HTTP(*protocolo usado en cada transacción de la World Wide Web*)/1.1, SOCKS5 y soporte de FTP-Proxy.
- ✓ Fichero de registro.
- ✓ Sincronización de directorios de navegación.
- ✓ Búsqueda de archivos remoto.

En el siguiente enlace puedes acceder a la página web oficial de FileZilla donde puedes descargarlo y encontrar documentación sobre el mismo.

<http://filezilla-project.org/>

1.3.- Tipos de usuarios.

¿Qué usuarios se pueden conectar al servidor ftp? ¿cualquiera? ¿sólo los usuarios del sistema?

Bien, típicamente existen dos tipos de usuarios:

- ✓ **Usuarios anónimos:** usuarios que tienen acceso y permisos limitados por el sistema de archivos. Al conectarse al servidor FTP sólo deben introducir una contraseña simbólica, normalmente cualquier dirección de correo -real o ficticia-, por ejemplo: `a@`.
- ✓ **Usuarios del sistema:** aquellos que disponen de una cuenta en la máquina que ofrece el servicio FTP. Al conectarse al servidor FTP deben introducir su contraseña de sistema.



Pero en ciertos servidores, como el servidor ProFTPD, existe una tercera posibilidad muy interesante: **usuarios virtuales**. Los usuarios virtuales poseen acceso y permisos al servidor FTP sin necesidad de ser usuarios del sistema, por lo tanto si un usuario virtual quisiera acceder al sistema operativo como si fuese un usuario del sistema, ya sea de forma local o remota no podría, pues su cuenta de usuario no existe en el sistema. Los usuarios virtuales tienen definida una contraseña propia y pueden estar definidos en ficheros de autenticación (de texto) con el mismo formato que los del sistema operativo GNU/Linux `/etc/passwd`, directorios `LDAP` (*protocolo de acceso unificado a un conjunto de información sobre una red*), bases de datos `SQL` (*lenguaje de consulta estructurado es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en éstas. Una de sus características es el manejo del álgebra y el cálculo relacional permitiendo efectuar consultas con el fin de recuperar de una forma sencilla información de interés de una base de datos, así como también hacer cambios sobre ella*) y servidores `RADIUS` (*protocolo de autenticación y autorización para aplicaciones de acceso a la red o movilidad IP. Una de las características más importantes del protocolo RADIUS es su capacidad de manejar sesiones, notificando cuando comienza y termina una conexión*).

Dependiendo del servidor ftp, podrás tener unos métodos de autenticación de usuarios u otros, por ejemplo en el servidor ftp ProFTPD se permite los siguientes métodos:

- ✓ Ficheros de autenticación del sistema operativo: `/etc/passwd` y `/etc/shadow`: Para ello usa las directivas `AuthUserFile` y `AuthGroupFile`.
<http://www.proftpd.org/docs/howto/AuthFiles.html>
- ✓ Usuarios virtuales definidos mediante ficheros de autenticación (de texto) propios, distintos de los del sistema operativo: para ello también usa las directivas `AuthUserFile` y `AuthGroupFile`.
- ✓ Autenticación PAM (*mecanismo de autenticación flexible que permite abstraer las aplicaciones y otro software del proceso de identificación. También permite construir políticas diferentes de autenticación para cada servicio*): Es necesario establecer la directiva `AuthPAMAuthoritative` a 'on'.
http://www.proftpd.org/docs/directives/linked/config_ref_AuthPAM.html
- ✓ Bases de datos SQL, tales como MySQL o Postgres. Para ello emplea el módulo `mod_sql`; más información sobre el uso de `mod_sql` lo puedes encontrar en el *HowTo SQL*.
<http://www.proftpd.org/docs/howto/SQL.html>
- ✓ LDAP: Para ello emplea el módulo `mod_ldap`.

- ✓ RADIUS: Para ello emplea el módulo `mod_radius`.

Mediante la directiva `UserPassword` (http://www.proftpd.org/docs/directives/linked/config_ref_UserPassword.html) se puede crear una contraseña para un usuario particular que sobrescribe la contraseña del usuario en `/etc/passwd` (o `/etc/shadow`), esta contraseña es solamente efectiva dentro del contexto en el cual la directiva es aplicada, esto es, no se modifica el fichero `/etc/passwd` (o `/etc/shadow`) sino que se da la posibilidad de que el usuario emplee otra contraseña distinta de la definida en los ficheros del sistema operativo.

En el siguiente anexo encontrarás más información sobre PAM.

Información sobre PAM

1.4.- Modos de conexión del cliente.

Si en una transferencia de archivos mediante el protocolo FTP el cliente posee un cortafuegos configurado para impedir acceso local a puertos TCP menores de 1024, ¿es posible la transferencia?

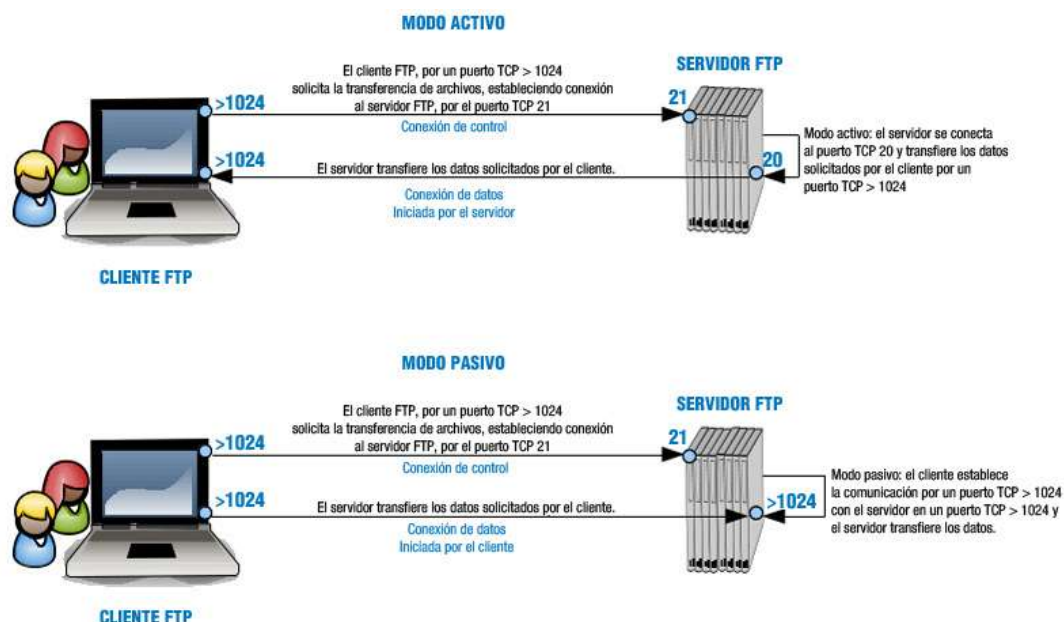
Ya se ha comentado que el servidor FTP a diferencia de otros servidores necesita dos puertos TCP para hacer posible la transferencia de archivos. Ahora bien, ¿son estos puertos siempre los mismos o no? ¿son independientes del tipo de cliente y servidor o no? Pues, básicamente depende de dos factores: del modo de conexión del cliente ftp y de la configuración del servidor ftp.

A priori, si no modificamos la configuración del servidor ftp esté otorgará siempre el puerto TCP 21 para el canal de conexión de control. Es el puerto del canal de transmisión de datos, el que varía, ¿cómo?, pues según el modo de conexión del cliente ftp, que puede ser activo o pasivo.

Cuando una aplicación cliente FTP inicia una conexión a un servidor FTP, abre el puerto 21 en el servidor. Se utiliza este puerto para arrojar todos los comandos al servidor. Cualquier petición de datos desde el servidor se devuelve al cliente a través de otro puerto TCP del servidor dependiendo del modo de conexión del cliente. Así:

- ✓ El modo activo es el método original utilizado por el protocolo FTP para la transferencia de datos a la aplicación cliente. Cuando el cliente FTP inicia una transferencia de datos, el servidor abre una conexión desde el puerto 20 en el servidor para la dirección IP y un puerto aleatorio sin privilegios (mayor que 1024) especificado por el cliente. Este arreglo implica que la máquina cliente debe poder aceptar conexiones en cualquier puerto superior al 1024. Con el crecimiento de las redes inseguras, tales como Internet, es muy común el uso de cortafuegos para proteger las máquinas cliente. Debido a que estos cortafuegos en el lado del cliente normalmente rechazan las conexiones entrantes desde servidores FTP en modo activo, se creó el modo pasivo.
- ✓ La aplicación FTP cliente es la que inicia el modo pasivo, de la misma forma que el modo activo. El cliente FTP indica que desea acceder a los datos en modo pasivo y el servidor proporciona la dirección IP y el puerto aleatorio, sin privilegios (mayor que 1024) en el servidor. Luego, el cliente se conecta al puerto en el servidor y descarga la información requerida.

A continuación puedes ver una imagen que muestra el funcionamiento de los dos modos: el activo y el pasivo.



En sistemas GNU/Linux es típico encontrar el archivo `/etc/services` que contiene una lista de puertos TCP/UDP relacionado con los servicios estándar que trabajan en los mismos. Ejecuta el comando `cat /etc/services | grep ftp` y encontrarás todos los puertos y servidores relacionados con la cadena `ftp`.

1.5.- Tipos de transferencia de archivos.

¿Es lo mismo descargar/subir mediante FTP un archivo de vídeo, que uno de texto o uno ejecutable?

Desde el punto de vista de FTP, los archivos se agrupan en dos tipos:

- ✓ **Archivos ASCII** (código de caracteres basado en el alfabeto latino. ASCII es, en sentido estricto, un código de siete bits, lo que significa que usa cadenas de bits representables con siete dígitos binarios (que van de 0 a 127 en base decimal) para representar información de caracteres. El código ASCII define así una relación entre caracteres específicos y secuencias de bits): son archivos de texto plano (.txt, .ps, .html...)
- ✓ **Archivos binarios**: todo lo que no son archivos de texto: ejecutables (.exe), imágenes (.jpg, .png ...), archivos de audio (.mp3, .wav ...), vídeo (.avi, .mov ...) , etcétera.

Es muy importante saber con qué tipo de archivos estás trabajando en la transferencia ya que si no utilizas las opciones adecuadas puedes destruir la información del archivo. El servidor ftp permite configurar la transferencia de archivos según el tipo del mismo, es por eso que al ejecutar el cliente FTP, antes de transferir un archivo, debes utilizar uno de los siguientes comandos o poner la correspondiente opción en un programa con interfaz gráfica:

- ✓ `ascii` para tipos de archivos ascii.
- ✓ `binary` para tipos de archivos binarios.

Relaciona cada extensión de archivo con el tipo de transferencia ftp correspondiente, escribiendo el número del tipo de transferencia en el cuadro correspondiente:

Extensión de archivo	Relación	Tipo de Transferencia	Relación	Extensión de archivo
txt (texto).	1	1. ascii. 2. binario.	1	ps (postscript).
html (página web).	1		2	mp3 (audio).
doc (documento).	2		2	tar (comprimido).
zip (comprimido).	2		2	tgz (comprimido).
bz2 (comprimido).	2			

1.6.- Establecer permisos en ftp.

El protocolo FTP sigue los permisos establecidos en entornos de tipo UNIX y sus similares GNU/Linux, con lo cual existen tres grupos de permisos en el siguiente orden: propietario, grupo y otros:

- ✓ **Propietario(user=u):** El creador o el que ha subido el archivo al servidor FTP.
- ✓ **Grupo(group=g):** Se refiere a un grupo de usuarios que posee la propiedad del archivo, al que probablemente pertenece el propietario.
- ✓ **Otros(others=o):** Son el resto de usuarios no propietarios o que no pertenecen al grupo indicado. Son el resto del mundo.



Cada grupo a su vez puede tener tres permisos en el siguiente orden: lectura, escritura y ejecución, identificados respectivamente por una 'r', una 'w' y una 'x'. La ausencia de permiso es identificada con el carácter '-'. Cada permiso tiene un equivalente numérico, así: r=4, w=2, x=1 y -=0. Por ejemplo: rw- identifica permiso de lectura y escritura o lo que es lo mismo **4+2+0=6**

En un sistema operativo tipo GNU/Linux mediante el comando 'ls -l' puedes ver los permisos asignados a ficheros y directorios, por ejemplo si la salida del anterior comando es:

```
-rw-r--r-- 1 alumno clase 0 jun 20 01:15 prueba1.txt
```

significa que,

- ✓ prueba1.txt es un fichero ya que -rw-r--r-- comienza con '-', si fuese un directorio aparecería una 'd'
- ✓ rw-r--r-- identifica los permisos del fichero prueba1.txt, que divididos 3 a 3 representan de izquierda a derecha: propietario, grupo, otros.
- ✓ rw- identifican los permisos del usuario propietario, en este caso alumno. Por lo tanto alumno posee los permisos de **lectura** y **escritura** sobre el fichero prueba1.txt o lo que es lo mismo **4+2+0=6**
- ✓ r-- identifican los permisos del grupo propietario, en este caso clase. Por lo tanto clase posee solamente el permiso de **lectura** o lo que es lo mismo **4+0+0=4**
- ✓ r-- identifican los permisos de los otros (resto del mundo). Por lo tanto todos los usuarios que no son alumno y aquellos que no pertenecen al grupo clase poseen solamente el permiso de **lectura** o lo que es lo mismo **4+0+0=4**

Por lo tanto los permisos rw-r--r-- equivalen a 644.

Es conveniente que le des un vistazo al manual de chmod y umask: man chmod y man umask.

Por otro lado en un sistema GNU/Linux, en principio, no todos los usuarios del sistema tienen acceso por ftp, así existe un fichero /etc/ftpusers que contiene una lista de usuarios que no tienen permiso de acceso por FTP. Por razones de seguridad al menos los siguientes usuarios deberían estar listados en este fichero: root, bin, uucp, news. Ten en cuenta que las líneas en blanco y las líneas que comiencen por el carácter '#' serán ignoradas.

Ejecutas en una consola de comandos en la ruta /home/alumno el comando ls -l obteniendo la siguiente salida:

```
drwxr-x--- 1 alumno clase 0 jun 20 01:16 Documentos
```

Entonces, con esa información puedes deducir que:

- ☒ **Documentos es un directorio con permisos 750.**
- ☐ Documentos es un fichero con permisos 750.
- ☒ **Documentos pertenece al usuario propietario alumno y al grupo propietario clase. Además el usuario alumno posee permisos de modificación, mientras que el grupo clase y los**

demás no poseen ese permiso.



Documentos pertenece al grupo propietario alumno y al usuario propietario clase. Además el usuario alumno posee permisos de modificación, mientras que el grupo clase y los demás no poseen ese permiso.

Documentos es un directorio ya que **drwxr-x---** comienza con 'd', si fuese un fichero aparecería un '-'

- ✓ **rw**xr-x--- identifica los permisos del directorio Documentos, que divididos 3 a 3 representan de izquierda a derecha: propietario, grupo, otros.
- ✓ **rw**x identifican los permisos del usuario propietario, en este caso **alumno**. Por lo tanto **alumno** posee los permisos de lectura, escritura y ejecución sobre el directorio **Documentos** o lo que es lo mismo **4+2+1=7**
- ✓ **r-x** identifican los permisos del grupo propietario, en este caso **clase**. Por lo tanto **clase** posee los permisos de lectura y ejecución o lo que es lo mismo **4+0+1=5**
- ✓ --- identifican los permisos de los **otros** (resto del mundo). Por lo tanto todos los usuarios que no son **alumno** y aquellos que no pertenecen al grupo **clase** no poseen permisos o lo que es lo mismo **0+0+0=0**

Por lo tanto los permisos **rw**xr-x--- equivalen a **750**.

1.7.- Servicio de transferencia de archivos en modo texto.

Como se comentó anteriormente, existen varios tipos de clientes ftp, entre los cuales los clientes en modo texto desde siempre estuvieron incorporados en las distribuciones GNU/Linux.

De entre los clientes tipo texto cabe destacar dos: el cliente en modo texto **ftp** y el cliente en modo texto **lftp**. En GNU/Linux Debian 6 se dispone del cliente modo **ftp** en una instalación básica. Para poder utilizarlo en el sistema simplemente hay que ejecutarlo como comando: el comando **ftp**.

Vamos a ver, a continuación, el comportamiento del cliente en modo texto **ftp** en la conexión al servidor **ftp.rediris.es**:

1. Básicamente la sintaxis es la siguiente:

```
ftp [-pingvd] [host [port]]
```

donde

- ✓ **host** identifica el servidor ftp
- ✓ **port** identifica el puerto, por defecto 21, por lo cual si conectas a un servidor ftp configurado en ese puerto no es necesario escribirlo, ya se considera.

Puedes ver la ayuda del comando ftp mediante: **man ftp** ó **info ftp**.

2. Al ejecutar el comando se abrirá una consola propia de ftp en la cual puedes introducir comandos ftp para: abrir conexión, moverse por rutas, descargar archivos ... Es muy típico ejecutarlo con el parámetro host, esto es, con el servidor ftp al cual quieres conectar:

```
root@debian-servidor-fp:~# ftp ftp.rediris.es
```

También puedes ejecutar el comando sin parámetros, de esta forma abrirás directamente la consola ftp y deberás actuar con ella a través de los comandos de la misma:

```
root@debian-servidor-fp:~# ftp
ftp> o
(to) ftp.rediris.es
```

3. A continuación se pedirá usuario y contraseña para establecer la conexión. En el caso del servidor de rediris puedes conectar mediante un usuario cualquiera y una contraseña cualquiera. Es muy típico en servidores ftp que exista un usuario anónimo, cuya contraseña sea cualquier dirección de correo -real o ficticia-, por ejemplo: **a@**.
4. Ahora en la consola ftp puedes ejecutar comandos, ¿cuales? Pues los que estén habilitados, y ¿cuales están habilitados? Lo puedes saber ejecutando el comando **help**.

En la siguiente imagen puedes ver el ejemplo de conexión ftp mediante el cliente en modo texto ftp al servidor ftp.rediris.es:

```

root@debian-servidor-fp:~# ftp
ftp> o
(To) ftp.rediris.es
Connected to zeppo.rediris.es.
220----- Welcome to Pure-FTPd [privsep] [TLS] -----
220-You are user number 38 of 3000 allowed.
220-
220- Bienvenido al FTP anónimo de RedIRIS.
220- Welcome to the RedIRIS anonymous FTP server.
220-
220-Local time is now 07:19. Server port: 21.
220-Only anonymous FTP is allowed here.
220-IPv6 connections are also welcome on this server.
220 You will be disconnected after 5 minutes of inactivity.
Name (ftp.rediris.es root):
331- RedIRIS - Red Académica y de Investigación Española
331- RedIRIS - Spanish National Research Network
331-
331- ftp://ftp.rediris.es -> http://ftp.rediris.es
331-
331-Debido a una incidencia hardware del almacenamiento que utiliza el servicio, éste
331-no estará disponible hasta nuevo aviso. Rogamos disculpen las molestias que este
331-fallo les pueda ocasionar. Estamos trabajando para resolver esta incidencia lo antes
331-posible.
331 Any password will work:
Password:
230 Any password will work:
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> help
Commands may be abbreviated. Commands are:
!          debug         mdir          qc           send
$          dir            egot         sendport    site
account   disconnect      mkdir        put          size
append    exit                rls          pwd          status
ascii     from                eode         quit         struct
bell      get                 eodtime      quote        system
binary    glob               eput         recv         unique
bye        hash              newer        reget        tenex
case      help              nmap         rstatus      tick
cd         idle              nlist        rhelp        trace
cdup      image             ntrans       rename       type
chmod     lcd               open         reset        user
close     ls                prompt       restart      unmask
cr        lacdef            passive      rmdir        verbose
delete    mdelete           proxy        runique      ?
ftp>

```

A continuación puedes ver el mismo ejemplo de conexión al servidor `ftp: ftp.rediris.es` utilizando el cliente en modo texto `lftp`:

1) Actualizar sistema operativo GNU/Linux Debian 6 Squeeze:

```

root@debian-servidor-fp:~# apt-get update
root@debian-servidor-fp:~# apt-get upgrade

```

2) Instalar cliente ftp en modo texto: lftp

```

root@debian-servidor-fp:~# apt-get install lftp

```

3) Ver ayuda comando lftp

```

root@debian-servidor-fp:~# lftp --help

```

Uso: lftp [OPCS] <servidor>

'lftp' es la primera orden ejecutada por lftp después de los archivos de configuración.

```

-f <archivo>      ejecuta órdenes del archivo y sale
-c <ords>         ejecuta las órdenes y sale
--help           muestra esta ayuda y sale
--version        muestra la versión de lftp y sale

```

Las demás opciones son las mismas que en la orden 'open'

```

-e <ord>          ejecuta la orden justo después de seleccionar
-u <usuario>[,<clave>] usa usuario/clave para autenticación
-p <puerto>      usa el puerto para una conexión
<sitio>          servidor, URL o nombre de señalador

```

4) Establecer conexión con el servidor ftp: ftp.rediris.es

```

root@debian-servidor-fp:~# lftp ftp.rediris.es
lftp ftp.rediris.es:~>

```

Como puedes ver conecta directamente, a diferencia del comando ftp, ya que no pide usuario y contraseña. Esto es debido a que automáticamente comprueba si la conexión mediante un usuario anónimo es posible, y si es posible introduce unos valores para establecer la conexión.

5) Introducir comando help para saber que comandos dispones dentro de la consola ftp:

```

lftp ftp.rediris.es:~> help
!<orden-de-shell> (órdenes) alias [<nombre> [<valor>]] bookmark [SUBORDEN]
cache [SUBORDEN]
cat [-b] <archivos> cd <dir remoto> chmod [OPTS] modo archivo. close [-a]
[re]cls [opts] [path/] [pattern] debug [<nivel>|off] [-o <archivo>] du [options] <dirs>
exit [<código>|bg]
get [OPCS] <arch_r> [-o <arch_l>] glob [OPTS] <cmd> <args> help [<ord>]

```

```
history -w file|-r file|-c|-l [cnt] jobs [-v]
kill all|<númtarea> lcd <dirlocal> lftp [OPCS] <servidor> ls [<args>]
mget [OPCS] <archivos> mirror [OPCS] [remoto [local]] mkdir [-p] <dirs> module nombre
[args] more <archivos>
mput [OPCS] <archivos> mrm <archivos> mv <archivo1> <archivo2> [re]nlist [<args>]
open [OPCS] <servidor> pget [OPCS] <arch_r> [-o <arch_l>] put [OPCS] <arch_l> [-o
<arch_r>] pwd [-p] queue [OPTS] [<cmd>]
quote <orden> repeat [OPTS] [delay] [command] rm [-r] [-f] <archivos> rmdir [-f] <dirs>
scache [<núm_sesión>] set [OPT] [<var> [<val>]] site <orden_directa_al_sitio> source
<archivo>
torrent [-O <dir>] <file> user <usuario|URL> [<clave>] version wait [<númtarea>] zcat
<archivos>
zmore <archivos>
lftp ftp.rediris.es:~>
```

1.7.1.- Comandos ftp.

En la consola ftp pueden estar disponibles múltiples comandos, algunos de los más empleados son los recogidos en la siguiente tabla:

ABRIR/CERRAR CONEXIÓN	
COMANDO/S Y ARGUMENTOS	EXPLICACIÓN
open servidor	Inicia conexión remota con un servidor ftp.
close / disconnect	Finalizan la sesión ftp sin cerrar la consola ftp.
bye / quit / exit	Terminan la sesión ftp y salen de la consola ftp.
!	Sale a línea de comandos del sistema operativo temporalmente sin cortar la conexión. Para volver, teclea exit en la línea de comandos.
AYUDA	
COMANDO/S Y ARGUMENTOS	EXPLICACIÓN
? / help	Muestra una lista de los comandos disponibles.
? comando / help comando	Muestra la información relativa al comando.
TRABAJAR CON DIRECTORIOS	
COMANDO/S Y ARGUMENTOS	EXPLICACIÓN
cd directorio	Cambia de directorio en el servidor remoto.
lcd directorio	Cambiarse de directorio en el equipo local (cliente ftp).
dir directorio / ls directorio	Listan el contenido del directorio remoto actual.
pwd	Muestra el directorio activo en el servidor.
lpwd	Muestra el directorio activo en el equipo local (cliente ftp).
rmdir directorio	Elimina un directorio vacío en el servidor.
mkdir directorio	Crea un directorio en el servidor.Crea un directorio en el servidor.
TRABAJAR CON FICHEROS	
COMANDO/S Y ARGUMENTOS	EXPLICACIÓN
delete archivo	Borrar un archivo en el servidor remoto.
mdelete patrón	Borrar varios archivos según un patrón.
get archivo	Obtiene archivo en el equipo cliente desde el servidor remoto.
mget archivos	Obtiene varios archivos desde el servidor remoto.
put archivo	Envía un archivo al servidor remoto.
mput archivos	Envía varios archivos al servidor remoto.
rename archivo	Cambia el nombre a un archivo en el servidor.
ascii	Para configurar y transferir archivos tipo ascii.
binary	Para configurar y transferir archivos tipo binario.
less archivo	Leer contenido de archivo mediante el comando less.
TRABAJAR CON PERMISOS	
COMANDO/S Y ARGUMENTOS	EXPLICACIÓN

chmod	Cambio de permisos en el servidor remoto.
umask	Configura el sistema de permisos en el lado remoto.

Observa la siguiente tabla con más comandos ftp.

COMANDO	USO
\$	Ejecuta macro
account	Envía comando a la cuenta del servidor remoto
append	Concatena un archivo
bell	Sonido de campanilla cuando el comando se ha completado
case	Mapeo de letras iguales
cdup	Cambiar al directorio padre en el servidor remoto
cr	Retorno de carro
debug	Configura modo de errores
form	Configurar formato de transferencia de archivos
glob	Transponer nombre de archivo local con un metacarácter
hash	Imprimir el metacarácter "#" por cada buffer transferido
idle	Configurar el tiempo disponible en el lado remoto
image	Para configurar y transferir archivos tipo binario
less	Ver el contenido de un archivo pudiendo subir y bajar por el mismo mediante las flechas
macdef	Define una macro
mdir	Lista contenido de varios directorios remotos
mls	Lista contenido de varios directorios remotos
mode	Configura el modo de transferencia
modtime	Modo de reloj
newer	Recibe el archivo remoto si es más nuevo que el de la máquina local
nlist	Lista el contenido de varios directorios remotos
nmap	Configura nombre de archivo de acuerdo a plantilla
ntrans	Configura tabla de traducción para mapeo de nombres de archivos
prompt	Fuerza la ejecución de múltiples comandos
proxy	Comando para conexión alternativa
sendport	Activa/desactiva use del comando PORT para cada conexión de de datos
set	
site	Envía un comando específico a la máquina remota
size	Muestra el tamaño de un archivo
struct	Configura la estructura de la transferencia de los archivos
sunique	Activa/desactiva almacenamiento único sobre la máquina remota
tenex	Transferencia de archivos de tipo tenex
trace	Activa/desactiva trazado de transferencia de paquetes
type	Configura el tipo de archivo a transferir
user	Envía información de usuario nuevo
verbose	Activa/Desactiva modo de entrega de información completa

1.8.- Servicio de transferencia de archivos en modo gráfico.

El servicio de transferencia de ficheros está bien, pero obliga a entender el funcionamiento de un servidor ftp mediante el uso de sus comandos. La verdad, es que no es muy interactivo, ¿entonces..., no existe la posibilidad de trabajar de otro modo más interactivo? Pues si, mediante clientes ftp de modo gráfico o mediante el navegador, ya que éste incorpora su propio cliente ftp.

Típicamente los clientes gráficos se comportan todos igual, esto es, tienen una interfaz parecida, básicamente presentan una ventana partida en dos secciones: la de la izquierda suele representar el equipo cliente ftp (*desde donde se intenta establecer la conexión*) y la de la derecha suele representar el equipo servidor ftp (quién recibe la conexión). Luego suelen existir, en alguna zona determinada de la ventana: en el centro entre las dos secciones, arriba de las dos secciones, etc una serie de botones, usualmente representados como flechas que indican la posibilidad de subir o descargar archivos. Incluso dependiendo del cliente en modo gráfico es posible guardar los datos de las conexiones como plantillas, de tal forma que la próxima vez que intentes establecer la conexión con un mismo servidor ftp en vez de tener que rellenar los campos referentes a la conexión puedes hacerlo a través de la plantilla que ya posee el valor de esos campos.

Dentro de los clientes ftp en modo gráfico cabe destacar dos: **gftp** y **filezilla**. A continuación puedes ver un ejemplo de como utilizarlos para establecer una conexión con un servidor ftp, el servidor **ftp.rediris.es**:

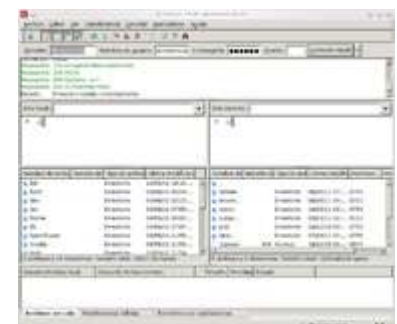
1. Cliente en modo gráfico **gftp**.

- ✓ **Servidor:** Escribe aquí el nombre o IP del servidor FTP: **ftp.rediris.es**
- ✓ **Puerto:** Escribe aquí el puerto TCP de la conexión de control, por defecto: **21**. Puedes omitirlo siempre y cuando sea el 21.
- ✓ **Usuario:** Escribe aquí el usuario con permisos de conexión en el servidor ftp. En la imagen puedes ver que no se ha escrito nada, esto es debido que el servidor **ftp.rediris.es** permite la entrada a cualquier usuario y el cliente gráfico **gftp** al intentar conectar te pedirá un usuario que tenga permisos para la conexión. Pulsas en cancelar y **gftp** cubrirá los campos usuario y contraseña, entrando al servidor ftp.
- ✓ **Contraseña:** Escribe aquí la contraseña del usuario con permisos de conexión en el servidor ftp. En la imagen puedes ver que no se ha escrito nada, esto es debido a la misma causa que en el campo Usuario.



2. Cliente en modo gráfico **filezilla**.

- ✓ **Servidor:** Escribe aquí el nombre o IP del servidor FTP: **ftp.rediris.es**
- ✓ **Nombre de usuario:** Escribe aquí el usuario con permisos de conexión en el servidor **ftp**. **Filezilla**, al contrario que **gftp**, no cubre los datos usuario y contraseña si tú no escribes nada en los campos, entonces debes escribir un nombre de usuario, por ejemplo **anonymous**, y una contraseña (*cualquier secuencia de caracteres*).
- ✓ **Contraseña:** Escribe aquí la contraseña del usuario con permisos de conexión en el servidor ftp. En la imagen puedes ver que se ha escrito una secuencia de caracteres punto, lo que significa que a la hora de escribir caracteres en ese campo no se muestra su valor por seguridad. Es necesario escribir una contraseña por lo comentado en el campo anterior: Nombre de usuario.
- ✓ **Puerto:** Escribe aquí el puerto TCP de la conexión de control, por defecto: **21**. Puedes omitirlo siempre y cuando sea el 21.



A continuación puedes ver información de la instalación del cliente en modo gráfico **gftp**.

1. Actualizar sistema operativo GNU/Linux Debian 6 Squeeze:

```
root@debian-servidor-fp:~# apt-get update
root@debian-servidor-fp:~# apt-get upgrade
```

2. Instalar cliente ftp en modo gráfico: gftp

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

Te proponemos el siguiente enlace de un vídeo práctico sobre la instalación y uso de FileZilla en una distribución GNU/Linux basada en Debian.

http://www.youtube.com/watch?feature=player_embedded&v=nBm-2rgkf5Y

Resumen:

Se ve una consola de comandos del usuario `root`, y en la misma el contenido del script `FileZilla Perfiles.sh`, a saber:

```
#!/bin/bash
# Actualizar repositorios
apt-get update
# Actualizar sistema operativo
apt-get upgrade
# Buscar paquete filezilla
apt-cache search filezilla
# Instalar paquete filezilla
apt-get install filezilla
# Arrancar filezilla en la consola del usuario del sistema: alumno
su -c filezilla alumno
# Dirigirse a Gestor de Sitios para crear los perfiles a conexiones ftp
```

Se ejecuta el script mediante el comando:

```
sh FileZilla_Perfiles.sh
```

Una vez acabada la ejecución del script aparece el cliente ftp gráfico **FileZilla**. En la interface arriba a la izquierda aparece el primer **icono Abrir el Gestor de Sitios**. Se hace click en el mismo y aparece un panel con dos secciones: la de la izquierda donde aparecen los Sitios configurados y la de la derecha donde aparecen las opciones configuradas de cada Sitio Seleccionado:

- ✓ En la sección de la izquierda no existe ningún sitio configurado y se pulsa en el botón **Nuevo Sitio**, apareciendo una caja de texto donde se escribe el nombre del sitio a configurar (nombre del perfil a guardar). Se escribe **REDIRIS** y se activa la sección de la derecha.
- ✓ En la sección de la derecha, al tener seleccionado **REDIRIS**, se escribe `ftp.rediris.es` en la caja de texto **Servidor** y se pulsa el botón **Aceptar**, desapareciendo el panel de configuración.

Se vuelve a pulsar en el **icono Abrir el Gestor de Sitios** y aparece el nuevo sitio(perfil) configurado REDIRIS. Ahora se pulsa el botón Conectar en la sección de la derecha del panel y se establece la conexión con el servidor `ftp.rediris.es`.

1.9.- Servicio de transferencia de archivos desde el navegador.

El navegador web también puede ejercer de cliente ftp y, puesto que la mayoría de los sistemas operativos cuentan con un navegador en su instalación, es una de las herramientas más usadas para transferencia de archivos.

Para poder usar el navegador como cliente ftp solamente debes escribir en la barra de dirección una dirección URL tipo, como la siguiente:

```
ftp://nombre_servidor_ftp:puerto
```

donde,

- ✓ `ftp://` indica que el protocolo que deseas que interprete el navegador sea el ftp.
- ✓ `nombre_servidor_ftp` representa el nombre o la IP del servidor ftp.
- ✓ `puerto` indica el puerto TCP, por defecto 21. Puedes omitirlo siempre y cuando sea el 21.

Si el servidor ftp permite la conexión a un usuario anónimo, al ejecutar `ftp://nombre_servidor_ftp:puerto` entrarás directamente al servidor ftp, esto es, el navegador no preguntará qué usuario y contraseña necesitas para establecer la conexión.

En la siguiente imagen puedes ver como puedes acceder al servidor ftp de rediris utilizando el navegador:



Así, lo único que tienes que hacer es escribir en la dirección URL: `ftp://ftp.rediris.es` y pulsar **Enter**, con lo cual, automáticamente, conectas con el servidor ftp, pudiendo visitar las carpetas y ver los ficheros como si de un explorador de archivos se tratará.

Para descargar las carpetas o archivos simplemente debes pulsar con el botón derecho del ratón sobre ellos y elegir la opción **Guardar enlace como...** -que aparece en Firefox y es similar en otros navegadores-.

Pero no todo van a ser ventajas al utilizar el navegador como cliente ftp, puesto que otros clientes tienen la posibilidad de continuar las descargas cuando estás sufrieron algún tipo de interrupción, cosa que no pasa con el cliente ftp del navegador, como por ejemplo el cliente gráfico FileZilla que soporta y reanuda la transferencia de archivos de gran tamaño(> 4 GB).

1.10.- Asegurando el servicio de transferencia de archivos.

Bien, pero ¿qué pasa con los datos en la transferencia? ¿viajan cifrados? ¿no? Pues empleando el protocolo ftp cualquiera que tenga acceso al canal de transmisión podrá ver en texto claro todo lo que se transmite, esto es, los datos no se cifran. Esto puede carecer de importancia, o no, según el contexto de la transmisión. Así, puede que a un organismo público no le importe compartir información a través de ftp y que los datos en la transferencia viajen sin cifrar y, sin embargo, a una empresa si le interese que los datos viajen cifrados.

Entonces, cuando interese asegurar el servicio de transferencia de archivos debes descartar el protocolo ftp y empezar a pensar en otras alternativas, como: `ftps` o `sftp`.

FTPS es una extensión del protocolo FTP que asegura el cifrado en la transferencia mediante los protocolos **SSL/TLS**. Permite tres tipos de funcionamiento:

- ✓ **SSL Implícito:**
 - ➔ Como conexiones HTTPS.
 - ➔ Usa los puertos 990 y 989.
- ✓ **SSL Explícito**
 - ➔ El cliente usa los mismos puertos estándar FTP: 20 y 21 pero se efectúa el cifrado en ellos.
 - ➔ Usa AUTH SSL.
- ✓ **TLS Explícito:**
 - ➔ Similar a SSL Explícito pero usa AUTH TLS.

El cifrado al que nos referimos es el cifrado de clave pública o asimétrico: **clave pública(kpub)** y **clave privada(kpriv)**. La **kpub** interesa publicarla para que llegue a ser conocida por cualquiera, la **kpriv** no interesa que nadie la posea, solo el propietario de la misma. Ambas son necesarias para que la comunicación sea posible, una sin la otra no tienen sentido, así una información cifrada mediante la **kpub** solamente puede ser descifrada mediante la **kpriv** y una información cifrada mediante la **kpriv** sólo puede ser descifrada mediante la **kpub**.

En el cifrado asimétrico podemos estar hablando de individuos o de máquinas, en nuestro caso hablamos de máquinas y de flujo de información entre el **cliente ftp(A)** y el **servidor ftp(B)**. Ver la siguiente tabla como ejemplo de funcionamiento del cifrado asimétrico:

Funcionamiento del cifrado asimétrico.



$A(\text{inf}) \rightarrow \text{inf cifrada} \rightarrow B [\text{descifrar inf}] \rightarrow B(\text{inf}) = A(\text{inf})$

$A(\text{inf}) \rightarrow \text{inf cifrada} = [(\text{inf})]k_{\text{pub}B} \rightarrow B [\text{inf. cifrada}]k_{\text{priv}B} \rightarrow B(\text{inf}) = A(\text{inf})$

Identificación

A	Cliente ftp.
inf cifrada = $[(\text{inf})]k_{\text{pub}B}$	Información cifrada mediante la clave pública de B obtenida a través de un certificado digital.
$[\text{inf. cifrada}]k_{\text{priv}B}$	Información descifrada mediante la clave privada de B.
B	Servidor ftp.

En el siguiente enlace encontrarás más información sobre asegurar FTP con TLS.

<http://tools.ietf.org/html/rfc4217>

En este otro enlace encontrarás más información sobre el protocolo TLS.

<http://tools.ietf.org/html/rfc4346>

1.11.- El servicio de transferencia de archivos en el proceso de despliegue de una aplicación Web.

¿Cómo actualizas una página web de forma remota? ¿Qué servicios suelen ofrecer las empresas de alojamiento web para que puedas subir archivos a tus aplicaciones? ¿Cuánto tiempo se mantiene la conexión subiendo un archivo?

Suele ser típico que cualquier aplicación web en Internet disponga de la posibilidad de subir archivos mediante una configuración del código fuente de la misma, una aplicación propia o una aplicación de terceros, como los paneles de administración web.



Si empleas una aplicación web para subir archivos debes tener en cuenta cuánto tiempo puedes mantener la conexión abierta con el servicio web y cuál es el tamaño máximo de subida de un archivo. Estas cuestiones suelen ser típicas de la configuración del servidor web. Por la contra, si empleas un servidor ftp dependerá de éste las cuestiones anteriores.

Se suele configurar el servidor web con unos parámetros: tiempo de conexión y tamaño máximo de subidas de archivos diferentes del servidor ftp, de tal forma que para archivos de tamaño no muy grandes se puedan emplear aplicaciones web y no se sufra un corte en la subida de archivos y, para archivos grandes, se emplee el servidor ftp.

Normalmente las empresas de alojamiento web (hosting) permiten la subida de archivos mediante un servidor ftp y poseen documentos sobre cómo operar con éste, esto es, documentación que explica cómo conectarse a sus servidores ftp a través de algún cliente ftp, como por ejemplo: `FileZilla`, `Cute FTP`, `Fetch` o `Transmit`. También suelen permitir usar `SCP` o `SFTP` para transferir ficheros de forma segura mediante un canal cifrado. Por ejemplo en Filezilla se puede establecer la conexión de forma cifrada directamente, sólo con indicar como puerto `TCP` el número del servidor `SSH`, por defecto, `22`.

También debes saber que muchos editores web permiten subir tu aplicación web al servidor con el protocolo FTP, esto te puede resultar más sencillo que el uso de una aplicación de FTP independiente.

Eso si, sea cual sea el método ftp que utilices para subir archivos y actualizar tu web, se desaconseja el uso de aplicaciones no actualizadas que podrían comprometer la seguridad de tu web.

A continuación puedes ver errores típicos, junto con sus soluciones, que puedes encontrar al subir tu aplicación mediante un servidor FTP:

- ✓ Tu cliente FTP muestra el error `access denied`, o similar, cuando subes o borras ficheros y carpetas: Comprueba que tu usuario FTP tenga permisos suficientes sobre la carpeta o fichero en la que desees subir o que desees borrar.
- ✓ Tus páginas no son reconocidas de forma automática al acceder a tu dominio: Los servidores GNU/Linux son sensibles a mayúsculas y minúsculas por lo que verifica el nombre de tus archivos.
- ✓ El cliente de FTP te muestra el mensaje `too many connections from your IP address`: Esto quiere decir que existen más conexiones abiertas con el servidor FTP desde la misma dirección IP de las permitidas. En ese caso, asegúrate que no exista ninguna aplicación, como un cortafuegos, que pueda estar bloqueando las conexiones abiertas, y provocando, de esta forma, que se establezcan más intentos de conexión de los necesarios.

2.- Instalación del servidor proftpd.

Caso práctico

Bien —pensó María— ya es la hora, una vez configurado el servidor web, alojada la página y comprobada su visibilidad a través de Internet, toca permitir la subida de archivos de gran tamaño a la carpeta upload preparada para tal fin. Así que, déjame pensar, María, ¿qué debo hacer?:



1. Esta empresa maneja archivos de gran tamaño, por lo que habrá que configurar un servidor ftp para que en la subida de archivos la conexión no se corte, que es lo más probable que ocurra a través de la subida de archivos mediante la propia aplicación web.
2. Mirar qué servicio han contratado, puesto que el nivel de cuota en disco puede variar según el servicio.
3. Crear un usuario virtual para que pueda subir archivos, —¿cómo lo haré? ¿mediante base de datos SQL? —Uhm..., creo que lo mejor será crear un usuario virtual y, como solamente se trata de un usuario, pues lo crearé mediante un archivo de autenticación.
4. Si se necesita crear otro usuario, pues otro en el archivo de autenticación y listo. ¿Y si necesito crear grupos? Pues lo mismo, en un archivo de autenticación de grupos.
5. Se necesita que la comunicación sea cifrada, con lo cual se debe emplear el protocolo SSL. — ¿Uhm...? Mejor el protocolo TLS, que es el sucesor de SSL. Pero, claro, si empleo cifrado voy a tener que utilizar otros puertos TCP distintos del servidor ftp que maneja por defecto: el 21 para la conexión de control y el 20 para la conexión de datos.
6. Ya está, mejor empleo TLS Explícito, de tal forma que puedo seguir utilizando los mismos puertos 20 y 21 para el cifrado.
7. Y todo esto no debe modificar la configuración ya realizada para otras empresas.

Pues clarísimo, lo que tengo que hacer es utilizar el servidor **ProFTPD**.

Entonces:

- ✓ Primero, comprobar si en este servidor dedicado está instalado y, si no lo está, instalarlo.
- ✓ Segundo, configurar el servidor proftpd de la siguiente forma:
 - Configuración independiente para esta empresa.
 - Usuario virtual en un archivo de autenticación.
 - Cifrado TLS Explícito para asegurar el cifrado.
 - Cuota de disco.
 - Permisos de subida en la carpeta upload correspondiente.

Pues, manos a la obra María, que se va haciendo tarde.

¿Por qué ProFTPD? Pues porque es un servidor FTP bajo licencia GPL altamente configurable, así permite:

- ✓ Usuarios virtuales con:
 - LDAP
 - BBDD: MySQL, PostgreSQL...
 - Ficheros de autenticación (ficheros de texto).
- ✓ Personalizar opciones según usuario/grupo.
- ✓ Seguridad mediante cifrado SSL/TLS.
- ✓ Configuraciones independientes mediante virtualhosts.

Para instalar el servidor proftpd en un sistema Operativo Debian 6.0 (squeeze) ejecutar el comando `apt-get install proftpd`. En la instalación deberás elegir si ProFTPD va a ejecutarse como un servicio desde `inetd` o como un **servidor independiente**. Ambas opciones tienen sus ventajas. Si sólo recibes unas pocas conexiones FTP diarias, probablemente sea mejor ejecutar ProFTPD desde `inetd` para ahorrar recursos. Por otro lado, con más tráfico, ProFTPD debería ejecutarse como un servidor independiente para evitar crear un proceso nuevo por cada conexión entrante.

En la instalación se crearán los usuarios `proftpd` y `ftp` con grupo `nogroup` y sin posibilidad de acceso a una consola del sistema. Se puede comprobar en el fichero `/etc/passwd` donde encontrarás nuevas líneas similares a las siguientes:

```
proftpd:x:106:65534::/var/run/proftpd:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
```

Te proponemos el siguiente enlace a un vídeo práctico sobre la instalación y uso de proftpd en una distribución GNU/Linux basada en Debian.

http://www.youtube.com/watch?feature=player_embedded&v=ijol1lTDpcl

Resumen

Se ve una consola de comandos del usuario **root**, y en la misma el contenido del script **Instalacion_ProFTPD.sh**, a saber:

```
#!/bin/bash
# Actualizar repositorios
apt-get update
# Actualizar sistema operativo
apt-get upgrade
# Buscar paquete proftpd
apt-cache search proftpd
# Instalar paquete proftpd-basic
apt-get install proftpd-basic
# Ruta de configuración de ProFTPD: /etc/proftpd
ls -l /etc/proftpd
# Servicio proftpd: Posibilidades
/etc/init.d/proftpd
# Servicio proftpd: Posibilidades
service proftpd

# Con esta configuración cualquier usuario del sistema puede acceder por ftp

# Modificar /etc/proftpd/proftpd.conf y descomentar las líneas correspondientes a Anonymous
# Puedes recargar la nueva configuración con:
## /etc/init.d/proftpd reload
## ó
## service proftpd reload

# Puedes reiniciar el servicio mediante:
## /etc/init.d/proftpd restart
## ó
## service proftpd restart

# Con esta nueva configuración puede aceptar por ftp el usuario anónimo
# anonymous o su alias: ftp

# Para desinstalar proftpd
# apt-get remove proftpd
```

Se ejecuta el script mediante el comando:

```
sh Instalacion_ProFTPD.sh
```

Una vez acabada la ejecución del script se ejecuta el comando **ftp**. A continuación se pulsa la telca **'o'** para abrir una conexión a un servidor ftp, se escribe **localhost** y se conecta al servidor local ftp. Para establecer la conexión se escribe el nombre del usuario del sistema **usuario-privado** y su contraseña. La conexión se ha establecido. Ahora dentro del servidor ftp, se ejecutan los comandos: **ls**, **pwd**, **help** y **quit**.

Una vez desestablecida la conexión se desinstala el servidor proftpd mediante el comando:

```
apt-get remove proftpd.
```

2.1.- Configuración de proftpd.

Su configuración es similar a la configuración del Servidor Apache, con lo cual si posees conocimientos sobre Apache tendrás mucho ganado, así tiene:

- ✓ Un fichero de configuración principal **/etc/proftpd/proftpd.conf**
- ✓ La posibilidad de configurar hosts virtuales (*dominio independiente que se puede alojar en un mismo servidor ftp*) o virtualhosts (*hosts virtuales*), de tal forma que un mismo servidor ftp puede alojar múltiples dominios

con sus configuraciones correspondientes, y todo lo que no esté incluido en la definición de cada virtualhost se heredarán de la configuración principal.

- ✓ Configuración a través de directivas.
- ✓ Contextos de configuración: global, directorio, virtualhost, anonymous.
- ✓ Modularización. Al igual que Apache se pueden activar/desactivar funcionalidades a través de módulos.

En el siguiente enlace encontrarás la página web oficial de ProFTPD.

<http://www.proftpd.org/>

En el siguiente enlace encontrarás información sobre las directivas de ProFTPD.

<http://www.proftpd.org/localsite/Userguide/linked/ref-directives.html>

Una vez instalado ProFTPD existirán dos ficheros de especial interés:

- ✓ El fichero `/etc/ftpusers`, ya comentado, que contiene una lista de usuarios que no tienen permiso de acceso por FTP. Por razones de seguridad al menos los siguientes usuarios deberían estar listados en este fichero: `root`, `bin`, `uucp`, `news`. Ten en cuenta que las líneas en blanco y las líneas que comiencen por el carácter '#' serán ignoradas.
- ✓ El fichero de configuración principal (`/etc/proftpd/proftpd.conf`)

En el fichero `proftpd.conf`:

- ✓ Las líneas en blanco y las líneas que comiencen por el carácter '#' serán ignoradas.
- ✓ Las líneas que comienzan por `Include` recogerán la configuración de los ficheros que la acompañan.
- ✓ `User proftpd` y `Group nogroup` identifican al usuario y grupo con el que se ejecuta proftpd.
- ✓ El soporte `LDAP`, `SQL`, `TLS`, `virtualhosts` y `cuotas` están desactivados, ver líneas: `#Include /etc/proftpd/ldap.conf`, `#Include /etc/proftpd/sql.conf`, `#Include /etc/proftpd/tls.conf`, `#Include /etc/proftpd/virtuals.conf` y `QuotaEngine off`.
- ✓ El mensaje de bienvenida se encuentra en el fichero `welcome.msg`
- ✓ Está configurado por defecto el modo de conexión ftp activo en el puerto TCP 21.
- ✓ Los usuarios que puedan conectarse por ftp:
 - ➔ Necesitan una consola de comandos activa, esto es, debe poseer una consola presente dentro del fichero `/etc/shells`
 - ➔ Pueden moverse por todo el sistema de ficheros, esto es, no están encerrados (jaula chroot) en sus directorios `/home`, puesto que la directiva `DefaultRoot ~` está comentada. Por seguridad sería conveniente descomentar la línea y recargar la configuración del servidor.
- ✓ Para evitar ataques de denegación de servicio solamente se permiten 30 conexiones simultáneas: `MaxInstances 30`
- ✓ Los permisos para los ficheros y directorios creados en la conexión ftp son: `644` y `755` respectivamente, ya que, `umask 022 022`, donde el primer grupo de tres números identifican los permisos de los ficheros y el segundo grupo identifica los permisos de los directorios.
- ✓ Encontrarás al final del mismo un ejemplo de configuración para usuarios anónimos.

Una vez retocada la configuración del servidor proftpd sólo reconocerá estos cambios cuando recargues su configuración, con lo cual debes ejecutar el comando:

```
/etc/init.d/proftpd restart
```

Si la configuración es correcta, y no quieres reiniciar proftpd, puedes recargar la configuración mediante el comando:

```
/etc/init.d/proftpd reload
```

A continuación veremos distintas configuraciones del servidor proftpd.

2.2.- Configurar el servidor como ftp privado.

Una vez instalado el servidor proftpd, en Debian 6 (squeeze), disponemos de un archivo de configuración </etc/proftpd/proftpd.conf>

Como has podido comprobar en la sección anterior, posee una configuración tipo por defecto. Ésta ya permite la conexión a tu servidor. ¿Con qué usuarios? Con cualquier usuario del sistema que posea una consola de comandos activa definida en </etc/shells>.

¿Cómo? Pues simplemente ejecutando cualquier cliente ftp que establezca una conexión con el puerto TCP 21 a tu servidor ftp. Por ejemplo utilizando el cliente de comandos ftp sería:

```
ftp usuario_del_sistema@servidor_ftp
```

donde,

servidor_ftp: puede ser el nombre de tu servidor ftp en </etc/hosts> o resuelto por DNS, o la IP de tu servidor ftp

Si no eres capaz de conectar revisa la configuración de tu cortafuegos y la sección 1.6 donde se exponen soluciones a problema típicos en conexiones ftp.

En el siguiente enlace encontrarás ejemplos de configuraciones del servidor ProFTPD.

<http://www.proftpd.org/docs/example-conf.html>

2.3.- Configurar el servidor como ftp privado y anónimo.

Igual te interesa contar en la configuración de tu servidor ftp con un usuario anónimo, el cual establecerá la conexión con cualquier contraseña y tendrá permisos diferentes a los usuarios del sistema (privados). Normalmente los permisos del usuario anónimo en un servidor ftp se establecerán para que solamente pueda moverse por los directorios y descargar archivos, nunca subirlos, esto es, normalmente el usuario anónimo no podrá crear ni eliminar ficheros y directorios.

Para hacer que proftpd permita conexiones mediante usuarios del sistema y usuarios anónimos debes modificar el fichero </etc/proftpd/proftpd.conf>

La modificación consiste en retocar su configuración activando a mayores la conexión mediante usuarios anónimos, puesto que los usuarios del sistema por defecto ya poseen acceso mediante ftp y se conectan con la misma contraseña del sistema.

Por lo tanto, al final del fichero incorpora las siguientes líneas:

```
# Inicio de la configuración Anonymous
# Usuario anónimo que entrará en el directorio ~ftp, esto es, en la variable $HOME del
usuario ftp
# En Debian 6 (squeeze) ~ftp=/home/ftp . Este directorio será la raíz de los directorios
en la conexión
# ftp, esto es, /home/ftp estará enjaulado (chroot) de tal forma que aunque el usuario
anónimo ftp
# quisiera acceder a otros directorios situados fuera de /home/ftp no podrá acceder.
<Anonymous ~ftp>
# Después de hacer login anónimo mediante ftp el servidor se ejecuta con el usuario ftp y
con el grupo
# nogroup
User      ftp
Group     nogroup
# La siguiente línea permite hacer login con el usuario "anonymous" igual que si fuera el
usuario "ftp"
UserAlias  anonymous ftp
# Cambios de apariencia, todos los ficheros parecerán pertenecer al usuario y grupo ftp
DirFakeUser  on ftp
DirFakeGroup on ftp
# No es necesario tener una shell en /etc/shells
RequireValidShell off
```

```
# Limitar el máximo número de logins anónimos concurrentes a 10
MaxClients      10
# Mensaje de conexión en el fichero welcome.msg
DisplayLogin     welcome.msg
# No permitir ESCRITURA en cualquier directorio al usuario anonymous, alias del usuario
ftp
<Directory *>
  <Limit WRITE>
    DenyAll
  </Limit>
</Directory>
# Fin de la configuración Anonymous
</Anonymous>
```

No olvides recargar la nueva configuración para que los cambios tengan efecto, ejecutando el comando:

```
/etc/init.d/proftpd restart ó /etc/init.d/proftpd reload.
```

Esta configuración permitirá conectar al servidor ftp mediante el usuario **ftp** o por su alias **anonymous** empleando una contraseña cualquiera. Una vez conectado solamente tendrá acceso al contenido de la carpeta `/home/ftp` y no podrá subir ni eliminar nada de ella.

2.4.- Configurar el servidor como ftp anónimo.

Puedes configurar proftpd para que permita conexiones mediante usuarios anónimos obligando a conectar sin poseer una contraseña del sistema, esto es, conectando con una contraseña cualquiera. Para ello debes modificar de nuevo el fichero </etc/proftpd/proftpd.conf>

Por lo tanto, cambia la configuración del usuario anonymous ftp de tal forma que al final del fichero aparezcan las siguientes líneas:

```
<Anonymous ~ftp>
  User      ftp
  Group     nobody
  # No es necesario tener una shell en /etc/shells
  RequireValidShell off
  # No se requiere contraseña en la conexión
  AnonRequirePassword off
  # No permitir ESCRITURA en cualquier directorio al usuario ftp
  <Directory *>
    <Limit WRITE>
      DenyAll
    </Limit>
  </Directory>
</Anonymous>
```

Puedes crear un usuario anónimo con carácter privado, esto es, que requiera contraseña para establecer la conexión. Por ejemplo, en la configuración siguiente se convierte el usuario del sistema 'invitado', para el servidor ftp, en un usuario anónimo que requiere contraseña para establecer la conexión. Además, solamente tendrá permisos de escritura desde cualquier equipo que conecte mediante la dirección de red 192.168.200.

```
<Anonymous ~invitado>
  User      invitado
  Group     nobody
  # Se requiere la contraseña de sistema del usuario invitado en la conexión
  AnonRequirePassword on
  # No permitir ESCRITURA en cualquier directorio al usuario invitado a no ser que establezca
  conexión
  # de la red 192.168.200.
  <Directory *>
    <Limit WRITE>
      Order allow, deny
      Allow from 192.168.200.
      Deny from all
    </Limit>
  </Directory>
</Anonymous>
```

Puedes convertir cualquier usuario privado (del sistema) que posea una consola de comandos válida en `/etc/shell` en un usuario anónimo. Por ejemplo en las configuraciones anteriores sólo tendrías que sustituir el usuario `'ftp'` y el usuario `'invitado'` por el nombre de un usuario existente en el sistema operativo.

En el siguiente enlace encontrarás información sobre la directiva Order de ProFTPD.

http://www.proftpd.org/docs/directives/linked/config_ref_Order.html

En el siguiente enlace encontrarás información sobre la directiva Allow de ProFTPD.

http://www.proftpd.org/docs/directives/linked/config_ref_Allow.html

En el siguiente enlace encontrarás información sobre la directiva Deny de ProFTPD.

http://www.proftpd.org/docs/directives/linked/config_ref_Deny.html

Según lo visto anteriormente, ¿cómo permitirías al usuario invitado establecer conexión desde las redes: 192.168.200 y 10.0.200?

Añadiendo en la misma línea de la directiva **Allow from 192.168.200**, la nueva red separando las redes mediante una coma, tal que así: **Allow from 192.168.200.,10.0.200**.

Y si además, ¿quisieras permitir el acceso desde los dominios `tuhostA.tudominio.edu`, `tuhostB.tudominio.edu` y `tuhostC.tudominio.edu`?

Pues, de la misma forma que anteriormente, añadiendo en la misma línea de la directiva **Allow from 192.168.200** las nuevas redes o dominios separándolos mediante signos coma, tal que así:

Allow from 192.168.200., 10.0.200., tuhostA.tudominio.edu, tuhostB.tudominio.edu, tuhostC.tudominio.edu

2.5.- Configurar el servidor ftp con múltiples dominios.

Anteriormente hemos visto cómo poder configurar el servidor ftp con múltiples usuarios, pero todos pertenecientes al mismo sitio/dominio, entonces, ¿no se puede configurar usuarios pertenecientes a distintos dominios en el mismo servidor ftp? La respuesta es que sí, sí se puede, ¿cómo?, mediante la configuración de hosts virtuales o virtualhosts. Éstos básicamente lo que hacen es permitir que un mismo servidor ftp pueda alojar múltiples dominios, así configurando hosts virtuales podemos alojar: **empresa1.com, empresa2.com,..., empresaN.com** en el mismo servidor ftp. Cada empresa tendrá su virtualhost único e independiente de los demás.

Aunque como se ha comentado anteriormente cada virtualhost es único e independiente de los demás, todo aquello que no esté incluido en la definición de cada virtualhost se heredará de la configuración principal: `proftpd.conf` (`/etc/proftpd/proftpd.conf`). Así, si quieres definir una directiva común en todos los virtualhost no debes modificar cada uno de los virtualhost introduciendo esa directiva sino que debes definir esa directiva en la configuración principal del servidor ftp ProFTPD, de tal forma que todos los virtualhost heredarán esa directiva, por ejemplo en `proftpd.conf` puedes encontrar la directiva `TimeoutIdle 1200`, que establece la directiva `TimeoutIdle` igual a 1200 segundos, esto es, indica el número máximo de segundos que puede estar un usuario sin hacer nada, pasado ese tiempo se cierra la conexión ftp.

En la definición de la directiva VirtualHost podemos poner la IP del servidor FTP ó bien el nombre DNS correspondiente. En nuestro escenario, la IP_Servidor_FTP=192.168.200.250, `ftp.empresa1.com` y `ftp.empresa2.com` identifican a la misma máquina.

Hay que tener en cuenta que si las IP empleadas son **IP privadas**, sin existencia en Internet, siempre que se haga referencia a las mismas a través de nombre de dominios, deberá existir un **servidor DNS**

que las resuelva en local o bien, en su defecto, deberán existir las entradas correspondientes en el fichero del sistema local `/etc/hosts`.

Independientemente de si configuras virtualhosts basados en IP o en nombre, puedes utilizar usuarios del sistema, pero también puedes crear los usuarios virtuales que quieras en un fichero similar a `/etc/passwd` y llamarlo en la configuración mediante la directiva `AuthUserFile`, entonces:

✓ Ejecuta el siguiente comando que creará un fichero de autenticación para usuarios virtuales,

```
ftppasswd --passwd --name user-empresal --file /etc/passwd.usuarios.virtuales --uid 107 --home /var/ftp/empresal --shell /bin/false
```

donde,

- `ftppasswd`, es el comando que permite crear los usuarios virtuales.
- `--passwd`, es el parámetro que pedirá la contraseña del usuario.
- `--name user-empresal`, identifica al usuario virtual de nombre `user-empresal`.
- `--file /etc/passwd.usuarios.virtuales`, creará, en caso de no existir, o modificará, en caso de existir el fichero de autenticación de usuarios virtuales.
- `--uid 107`, es el identificador perteneciente al usuario del sistema **ftp**. Se puede saber ejecutando el comando: `id ftp`.
- `--home /var/ftp/empresal`, identifica a donde se conecta el usuario.
- `--shell /bin/false`, identifica una consola de comandos que no permite conexión como usuario del sistema.

Ejecuta también el comando:

```
ftppasswd --passwd --name user-empresa2 --file /etc/passwd.usuarios.virtuales --uid 107 --home /var/ftp/empresa2 --shell /bin/false
```

A continuación prosigues, dependiendo si deseas configurar virtualhosts basados en IP o virtualhosts basados en nombre.

2.6.- Virtualhosts basados en nombre.

En la definición de la directiva `VirtualHost` podemos poner la IP del servidor FTP ó bien el nombre DNS correspondiente. En nuestro escenario, la `IP Servidor FTP=192.168.200.250`, `ftp.empresa1.com` y `ftp.empresa2.com` identifican a la misma máquina y a la misma IP. Ahora si, cada virtualhost, así como el servidor principal, deben servir en un puerto TCP distinto.

¿Cómo lo haces? Sigues el procedimiento:

1. En la configuración de ProFTPD (`/etc/proftpd/proftpd.conf`) debes activar la configuración del fichero `virtuals.conf` descomentando la línea:

```
Include /etc/proftpd/virtuals.conf
```

2. Agrega la configuración virtualhost para empresa1 en el fichero `/etc/proftpd/virtuals.conf`

```
<VirtualHost 192.168.200.250>
    Port 2121
    ServerName "Servidor FTP empresa1"
    AuthUserFile /etc/passwd.usuarios.virtuales
    DefaultRoot /var/ftp/empresa1/
    RequireValidShell off
</VirtualHost>
```

3. Agrega la configuración virtualhost para empresa2 en el fichero `/etc/proftpd/virtuals.conf`

```
<VirtualHost ftp.empresa2.com>
    Port 2122
    AuthUserFile /etc/passwd.usuarios.virtuales
    ServerName "Servidor FTP empresa2"
    DefaultRoot /var/ftp/empresa2/
    RequireValidShell off
</VirtualHost>
```

4. Configura permisos en las carpetas `/var/ftp/empresa1/` y `/var/ftp/empresa2/` para los usuarios virtuales:

```
chown ftp /var/ftp/empresa1/ /var/ftp/empresa2/
```

5. Recarga la configuración del servidor.

```
/etc/init.d/proftpd restart
```

Explicación fichero virtualhost:

- <VirtualHost IP_Servidor_FTP>** → Inicio etiqueta **virtualhost**: define la IP del servidor ftp. También puede ser **<VirtualHost Nombre_DNS_Servidor_FTP>**
- Port numero** → Identifica el puerto TCP por el que espera la conexión el servidor FTP
- ServerName "Servidor FTP empresaX"** → Configura el nombre que se muestra en la conexión de los usuarios.
- DefaultRoot /var/ftp/empresaX/** → Definición de la ruta que sirve ProFTPD, en este caso: **/var/ftp/empresaX/** mediante la directiva **DefaultRoot**, esto es, indica que los usuarios cuando conecten con el servidor ftp estarán enjaulados en la ruta **/var/ftp/empresaX/**, con lo cual no podrán acceder a otro directorio que no esté contenido dentro de éste.
- RequireValidShell off** → No es necesario tener una Shell declarada en el fichero **/etc/shells**
- </VirtualHost>** → Fin de la etiqueta **VirtualHost**: fin de la definición de este virtualhost para la empresa1.

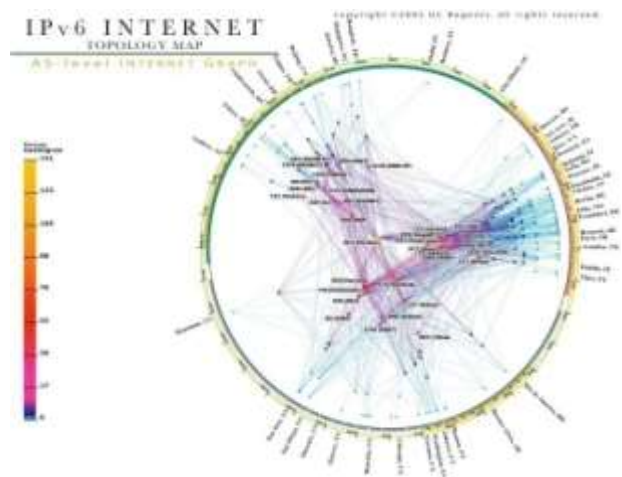
2.7.- Virtualhosts basados en IP.

En la definición de la directiva **VirtualHost** podemos poner la IP del servidor FTP o bien el nombre DNS correspondiente. En nuestro escenario, la **IP_Servidor_FTP=192.168.200.250**, **ftp.empresa1.com** y **ftp.empresa2.com** identifican a la misma máquina y a distintas IP. Ahora es indiferente el puerto TCP en el que sirve cada virtualhost, así como el servidor principal, esto es, puede ser el mismo o no ya que ahora cada puerto está relacionado con una IP distinta.

La IP que debemos poner ahora en la definición de la directiva **Virtualhost** cambia, cada IP corresponde a una interfaz de red del servidor FTP, en nuestro escenario:

IP_Servidor_FTP=192.168.200.250,
ftp.empresa1.com se identifica con
192.168.200.251 y **ftp.empresa2.com** con
192.168.200.252

Este método no aporta ventajas sobre el anterior, es más, aún puede ser más difícil de mantener si las IP del servidor FTP se modifican con cierta frecuencia.



¿Cómo lo haces? Sigues el mismo procedimiento usado para los virtualhost basados en nombre, únicamente se diferencia en la configuración de los virtualhost, así:

1. Modifica la configuración virtualhost para empresa1 en el fichero **/etc/proftpd/virtuals.conf**

```
<VirtualHost 192.168.200.251>
  ServerName "Servidor FTP empresa1"
  AuthUserFile /etc/passwd.usuarios.virtuales
  DefaultRoot /var/ftp/empresa1/
  RequireValidShell off
</VirtualHost>
```

2. Agrega la configuración virtualhost para empresa2 en el fichero **/etc/proftpd/virtuals.conf**

```
<VirtualHost ftp.empresa2.com>
  AuthUserFile /etc/passwd.usuarios.virtuales
  ServerName "Servidor FTP empresa2"
```

```
DefaultRoot /var/ftp/empresa2/
RequireValidShell off
</VirtualHost>
```

3. Configura permisos en las carpetas `/var/ftp/empresa1/` y `/var/ftp/empresa2/` para los usuarios virtuales:

```
chown ftp /var/ftp/empresa1/ /var/ftp/empresa2/
```

4. Recarga la configuración del servidor.

```
/etc/init.d/proftpd restart
```

Explicación fichero virtualhost:

`<VirtualHost IP_Servidor_FTP>` →

Inicio etiqueta virtualhost: define la IP del servidor ftp.

También puede ser `<VirtualHost Nombre_DNS_Servidor_FTP>`

`ServerName "Servidor FTP empresaX"` →

Configura el nombre que se muestra en la conexión de los usuarios.

`DefaultRoot /var/ftp/empresaX/` →

Definición de la ruta que sirve ProFTPD, en este caso: `/var/ftp/empresaX/` mediante la directiva `DefaultRoot`, esto es, indica que los usuarios cuando conecten con el servidor ftp, estarán enjaulados en la ruta `/var/ftp/empresaX/`, con lo cual no podrán acceder a otro directorio que no esté contenido dentro de éste.

`RequireValidShell off` →

No es necesario tener una shell declarada en el fichero `/etc/shells`.

`</VirtualHost>` →

Fin de la etiqueta `VirtualHost`: fin de la definición de este virtualhost para la empresa1.

2.8.- Cuotas de disco para los usuarios (I).

La capacidad de almacenamiento no es infinita, por lo tanto será interesante saber cómo crear cuotas de disco para los usuarios y ya puestos para los usuarios en los virtualhosts.

El archivo `/etc/proftpd/proftpd.conf` llama mediante la directiva `Include` al archivo `/etc/proftpd/modules.conf` en el que están activadas las cuotas (`LoadModule mod_quotatab.c`, `LoadModule mod_quotatab_file.c`), luego para activarlas tienes que sustituir en el archivo `/etc/proftpd/proftpd.conf` el código:

```
<IfModule mod_quotatab.c>
    QuotaEngine off
</IfModule>
```

por el código siguiente:

```
<IfModule mod_quotatab.c>
    QuotaEngine on
    QuotaLog /var/log/proftpd/quota.log
    <IfModule mod_quotatab_file.c>
        QuotaLimitTable file:/etc/proftpd/ftpquota.limittab
        QuotaTallyTable file:/etc/proftpd/ftpquota.tallytab
    </IfModule>
</IfModule>
```

donde,

`<IfModule mod_quotatab.c> ... </IfModule>` →

Indica que si el módulo `mod_quotatab.c` está cargado en el archivo `/etc/proftpd/modules.conf` se realizarán las directivas que contengan.

`QuotaEngine on` →

Activa las cuotas.

`QuotaLog /var/log/proftpd/quota.log` →

Indica el archivo de registro sobre cuotas.

`<IfModule mod_quotatab_file.c> ... </IfModule>` →

Indica que si el módulo `mod_quotatab_file.c` está cargado en el archivo

`/etc/proftpd/modules.conf` se realizarán las directivas que contengan.

`QuotaLimitTable file:/etc/proftpd/ftpquota.limittab` → Indica el archivo sobre el límite de cuotas `Limit`.

`QuotaTallyTable file:/etc/proftpd/ftpquota.tallytab` → Indica el archivo sobre el límite de cuotas `Tally`.

Para ProFTPD existen básicamente dos tipos de cuotas: `limit` y `tally`.

- ✓ `Limit`: Es la cuota que te interesa si estás pensando en restringir el espacio en disco a los usuarios. Éste puede ser `soft`, cuando existe un espacio de gracia (tamaño en bytes) que puede sobrepasar el límite, o `hard` cuando no existe un espacio de gracia.
- ✓ `Tally`: Utilizado cuando quieres limitar el número de ficheros que se utilizan.

Para mayor información sobre las cuotas puedes visitar la documentación oficial de ProFTPD sobre `mod_quotatab`.

http://www.proftpd.org/docs/contrib/mod_quotatab.html

La forma más sencilla de crear las cuotas es hacer el símil `upload=espacio en disco`, con lo cual los archivos subidos no pueden ocupar más del que queramos darle como espacio en disco, esto es, los bytes subidos funcionan como espacio en disco, ya que no existe diferencia entre ellos, pues los bytes cargados a través de FTP se almacenan automáticamente en el disco, por lo que deberías emplear la cuota tipo `limit`.

Puedes crear las cuotas mediante el comando `ftpquota`:

```
# ftpquota --create-table --type=limit --table-path=/etc/proftpd/ftpquota.limittab
# ftpquota --create-table --type=tally --table-path=/etc/proftpd/ftpquota.tallytab
```

Por ejemplo, si quisieras limitar a un usuario de nombre `user-empresal` el espacio de subida en 4 GB:

```
# ftpquota --add-record --type=limit --name=user-empresal --quota-type=user \
--bytes-upload=4 --units=Gb --table-path=/etc/proftpd/ftpquota.limittab
```

Y si quisieras limitar la subida y bajada a 4 GB y 2 GB respectivamente al usuario `user-empresa2`:

```
# ftpquota --add-record --type=limit --name=user-empresa2 --quota-type=user \
--bytes-upload=4 --bytes-download=2 --units=Gb --table-path=/etc/proftpd/ftpquota.limittab
```

2.8.1.- Cuotas de disco para los usuarios (II).

Bien, pero, ¿cómo verificar el funcionamiento de las cuotas?. Y si quisieras comprobar la cuota de un usuario, ¿es posible? ¿Y si quisieras actualizarla? ¿Y desactivarlas para algún usuario? ¿Y borrarlas?

Pues, utilizas el comando `ftpquota` como sigue:

- ✓ Para ver los registros de cuotas, esto es, a quién se le está ejerciendo las cuotas:

```
# ftpquota --show-records --type=limit --table-path=/etc/proftpd/ftpquota.limittab
-----
Name: user-empresal
Quota Type: User
Per Session: False
Limit Type: Hard
Uploaded bytes: 4294967296.00
Downloaded bytes: unlimited
Transferred bytes: unlimited
Uploaded files: unlimited
Downloaded files: unlimited
Transferred files: unlimited
```

- ✓ Para actualizar la cuota de un usuario, por ejemplo, `user-empresal`:

```
# ftpquota --update-record --type=limit --name=user-empresal --quota-type=user \
--bytes-upload=2300 --units=Mb --table-path=/etc/proftpd/ftpquota.limittab
```

con lo cual, si compruebas de nuevo los registros, verás que los cambios surgieron efecto:

```
# ftpquota --show-records --type=limit --table-path=/etc/proftpd/ftpquota.limittab
-----
Name: user-empresa1
Quota Type: User
Per Session: False
Limit Type: Hard
Uploaded bytes: 2411724800.00
Downloaded bytes: unlimited
Transferred bytes: unlimited
Uploaded files: unlimited
Downloaded files: unlimited
Transferred files: unlimited
```

✓ Para desactivar la cuota de un usuario debes borrar el registro, por ejemplo, user-empresa1:

```
# ftpquota --delete-record --type=limit --name=user-empresa1 --quota-type=user
```

con lo cual, si compruebas de nuevo los registros, verás que los cambios surgieron efecto:

```
# ftpquota --show-records --type=limit --table-path=/etc/proftpd/ftpquota.limittab
ftpquota: (empty table)
```

Puedes ver la ayuda del comando `ftpquota` mediante: `ftpquota --help`.

No olvides recargar la configuración del servidor ProFTPD: `/etc/init.d/proftpd restart`.

2.9.- Acceso seguro mediante TLS.

En Debian 6 ([squeeze](#)) al instalar el paquete `proftpd` ya se puede establecer la conexión por TLS, siempre y cuando se configure el archivo `/etc/proftpd/tls.conf` y procedas como sigue:

1. Edita el archivo `/etc/proftpd/proftpd.conf` y descomenta la línea:

```
Include /etc/proftpd/tls.conf
```

2. Crea las claves, pública y privada, para la conexión cifrada:

✓ Método 1: Instalación del paquete `openssl` (*Paquete de herramientas de administración y bibliotecas relacionadas con la criptografía, que suministran funciones criptográficas, entre otros a navegadores web, para acceso seguro a sitios mediante el protocolo HTTPS*) y ejecución del comando `openssl`.

```
# apt-get install openssl
# openssl req -x509 -newkey rsa:1024 -keyout /etc/ssl/private/proftpd.key -out \
/etc/ssl/certs/proftpd.crt -nodes -days 3650
Generating a 1024 bit RSA private key
..+++++
.....+++++
writing new private key to '/etc/ssl/private/proftpd.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Madrid
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:EMPRESA1
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []: ftp.empresa1.com
Email Address []:
```

✓ Método 2: Comando `proftpd-gencert`. Los dos comandos anteriores se resumen en uno, con la salvedad que el certificado será válido solamente durante 1 año y no 10:

```
# proftpd-gencert
```

3. Modifica los permisos:

```
# mv /etc/ssl/private/proftpd.key /etc/ssl/
# chmod 0600 /etc/ssl/proftpd.key
# chmod 0644 /etc/ssl/certs/proftpd.crt
```

4. Modifica el fichero `/etc/proftpd/tls.conf` como se indica en el fichero ejemplo [tls.conf](#)

5. Recarga la configuración del servidor ProFTPD:

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

6. Comprueba la configuración mediante el usuario del servidor ftp `invitado`, creado anteriormente, con un cliente FTPES, es decir, un cliente ftp que permita la conexión por TLS como FileZilla.

Puedes verificar en tiempo real las conexiones con el servidor ftp revisando los archivos de registro mediante los comandos:

```
tail -f /var/log/proftpd/proftpd.log
tail -f /var/log/proftpd/tls.log
```

7. Si deseas, puedes hacer valer la configuración para todos los usuarios, incluso aquellos pertenecientes a virtualhost, modificando el fichero `/etc/proftpd/tls.conf` como se indica en el fichero ejemplo [tls2.conf](#).

Te proponemos el siguiente enlace de un vídeo práctico sobre cómo configurar TLS en el servidor ProFTPD y cómo configurar una plantilla deFileZilla que soporta conexión TLS. La configuración se realiza sobre una distribución GNU/Linux basada en Debian.

http://www.youtube.com/watch?feature=player_embedded&v=jvdR5nZ30gE

Resumen:

Se ve una consola de comandos del usuario `root`, donde se muestra el contenido del script `Instalacion_ProFTPD_TLS_Filezilla_FTPES.sh`, a saber:

```
#!/bin/bash
# Actualizar repositorios
apt-get update
# Actualizar sistema operativo
apt-get upgrade
# Buscar paquete proftpd
apt-cache search proftpd
# Instalar paquete proftpd-basic
apt-get install proftpd-basic
# Ruta de configuración de ProFTPD: /etc/proftpd
ls -l /etc/proftpd
# Servicio proftpd: Posibilidades
/etc/init.d/proftpd
# Servicio proftpd: Posibilidades
service proftpd

# Con esta configuración cualquier usuario del sistema puede acceder por ftp

# Modificar /etc/proftpd/proftpd.conf y descomentar la línea: Include /etc/proftpd/tls.conf
TEMPORAL=`mktemp`
cat /etc/proftpd/proftpd.conf | sed "s/\\#Include    \\etc\\/proftpd\\/tls.conf/Include
\\etc\\/proftpd\\/tls.conf/g" > $TEMPORAL
mv $TEMPORAL /etc/proftpd/proftpd.conf

# Modificar /etc/proftpd/tls.conf para que tenga el siguiente contenido:
cat > /etc/proftpd/tls.conf << EOF
#####Fichero
/etc/proftpd/tls.conf#####
#
# Proftpd sample configuration for FTPS connections.
#
# Note that FTPS impose some limitations in NAT traversing.
# See http://www.castaglia.org/proftpd/doc/contrib/ProFTPD-mini-HOWTO-TLS.html
# for more information.
#

<IfModule mod_tls.c>
<global>
TLSEngine on
TLSLog /var/log/proftpd/tls.log
</global>

TLSProtocol SSLv23

<global>
TLRSACertificateFile /etc/ssl/certs/proftpd.crt
TLRSACertificateKeyFile /etc/ssl/private/proftpd.key
TLSOptions NoCertRequest
TLSVerifyClient off
TLSRequired on
TLSPRenegotiate required off
</global>
</IfModule>
```

```
#####Fin
/etc/proftpd/tls.conf#####

EOF

# Generar el certificado
proftpd-gencert

# Puedes recargar la nueva configuración con:
## /etc/init.d/proftpd reload
## ó
## service proftpd reload

# Puedes reiniciar el servicio mediante:
/etc/init.d/proftpd restart
## ó
## service proftpd restart

# Con esta nueva configuración puedes acceder mediante FTPES
# Para ello crearemos una Plantilla en FileZilla:
## 1) Ver script FileZilla_Perfiles.sh en el video:
## FileZilla. Crear sitios (perfiles) --> http://www.youtube.com/watch?v=nBm-2rgkf5Y
## 2) Arrancar FileZilla como usuario del sistema alumno.
  su -c filezilla alumno

# Para desinstalar proftpd
# apt-get remove proftpd
Se ejecuta el script mediante el comando:
sh Instalacion_ProFTPD_TLS_FileZilla_FTPES.sh
```

Antes de finalizar la ejecución se escribe, como consecuencia del comando `proftpd-gencert` para la generación del certificado:

```
Country Name: ES
Locality Name: Madrid
Organization Name: Empresa
Common Name: ftp.empresa.ftpes.local
```

Una vez acabada la ejecución del script aparece el cliente ftp gráfico **FileZilla**. En la interface arriba a la izquierda aparece el primer icono **Abrir el Gestor de Sitios**. Se hace clic en el mismo y aparece un panel con dos secciones: la de la izquierda donde aparecen los Sitios configurados y la de la derecha donde aparecen las opciones configuradas de cada Sitio Seleccionado:

En la sección de la izquierda aparecen dos sitios configurados, estando seleccionado el sitio **FTPES-PLANTILLA** y mostrándose a la derecha la configuración del mismo:

- ✓ En la caja de texto Servidor: `localhost`.
- ✓ En `Server Type`: **FTPES** – FTP sobre TLS/SSL explícito.
- ✓ En `Logon Type`: Preguntar la contraseña.
- ✓ En la caja de texto Usuario: `alumno`.

y se pulsa el botón **Aceptar**, desapareciendo el panel de configuración.

Se vuelve a pulsar en el icono **Abrir el Gestor de Sitios** y aparece el nuevo sitio (perfil) configurado **FTPES-PLANTILLA**. Ahora se pulsa el botón **Conectar** en la sección de la derecha del panel y aparece una caja de texto preguntando la contraseña del usuario alumno, se escribe, acepta y aparece el certificado digital de conexión a la espera de aceptarlo, se acepta y se establece la conexión con el servidor `localhost`.

A continuación se accede al Gestor de Sitios y se borra la plantilla **FTPES-PLANTILLA** haciendo clic en el botón **Borrar** y aceptando la confirmación de borrado, luego se accede de nuevo al Gestor de Sitios para pulsar en el botón **Nuevo Sitio**, apareciendo una caja de texto donde se escribe el nombre del sitio a configurar (nombre del perfil a guardar). Se escribe **FTPES-PLANTILLA** y se activa la sección de la derecha.

En la sección de la derecha, al tener seleccionado **FTPES-PLANTILLA**, se escriben de nuevo los parámetros anteriores:

- ✓ En la caja de texto Servidor: **localhost**
- ✓ En **Server Type**: **FTPES – FTP sobre TLS/SSL explícito**
- ✓ En **Logon Type**: **Preguntar la contraseña**
- ✓ En la caja de texto Usuario: **alumno**

y se pulsa el botón **Aceptar**, desapareciendo el panel de configuración.

Se hace clic en el icono **Desconectar del servidor actualmente visible** y se vuelve a pulsar en el icono **Abrir el Gestor de Sitios** y aparece el nuevo sitio(perfil) configurado **FTPES-PLANTILLA**. Ahora se pulsa el botón **Conectar** en la sección de la derecha del panel y se establece la conexión con el servidor **localhost**.

Anexo I - PAM

¿Qué es PAM?

La idea original de los Pluggable Authentication Modules, en adelante PAM, fue de Sun y sus especificaciones se encuentran recogidas en RFC 86.0. Sin embargo, muchos otros sistemas adoptaron esta solución y cuentan desde hace tiempo con sus propias implementaciones.

En este sentido, GNU/Linux no es una excepción y, gracias a Red Hat, disfruta ya desde hace años de la funcionalidad que ofrece Linux-PAM (*se hará referencia “PAM” y “Linux-PAM” indistintamente*)

Pero, ¿qué es PAM exactamente? Tal y como puede leerse en la FAQ (<http://www.kernel.org/pub/linux/libs/pam/FAQ>) oficial del proyecto, PAM es, básicamente, un mecanismo flexible para la autenticación de usuarios. Y quizás esta característica, la flexibilidad, sea su aportación más importante.

A lo largo de los años, desde los primeros sistemas UNIX, los mecanismos de autenticación han ido evolucionando y han aparecido nuevas opciones: desde mejoras del clásico `/etc/passwd` —como la shadow— hasta dispositivos hardware orientados a la autenticación. Y, claro está, cada vez que aparecía y se popularizaba un nuevo método, los desarrolladores debían modificar sus programas para darles soporte.

PAM permite el desarrollo de programas independientes del mecanismo de autenticación a utilizar. Así es posible que un programa que aproveche las facilidades ofrecidas por PAM sea capaz de utilizar desde el sencillo `/etc/passwd` hasta dispositivos hardware —como lectores de huella digital—, pasando por servidores LDAP (*Lightweight Directory Access Protocol*) o sistemas de gestión de bases de datos. Y, por supuesto, todo esto sin cambiar ni una sola línea de código.

Pero PAM va más allá todavía, permitiendo al administrador del sistema construir políticas diferentes de autenticación para cada servicio.

En resumen, podrían sintetizarse las ventajas más importantes de PAM en los siguientes puntos:

- ✓ Ofrece un esquema de autenticación común y centralizado.
- ✓ Permite a los desarrolladores abstraerse de las labores de autenticación.
- ✓ Facilita el mantenimiento de las aplicaciones.
- ✓ Ofrece flexibilidad y control tanto para el desarrollador como para el administrador de sistema.

Grupos de gestión

A pesar de lo que se ha explicado anteriormente, la misión de PAM no es, únicamente, comprobar que un usuario es quien dice ser —autenticación—. Su alcance es mucho mayor y pueden dividirse sus tareas en cuatro grupos independientes de gestión, cada uno de los cuáles se encarga de un aspecto diferente de los servicios restringidos.

account (cuenta) En este grupo se engloban tareas que no están relacionadas directamente con la autenticación. Algunos ejemplos son permitir/denegar el acceso en función de la hora, los recursos disponibles o, incluso, la localización. Ofrece verificación de cuentas de usuario. Por ejemplo, se encarga de determinar si el usuario tiene o no acceso al servicio, si su contraseña ha caducado, etc.

authentication (autenticación) Tareas encaminadas a comprobar que, efectivamente, el usuario es realmente quien dice ser. A menudo, cuando se habla de PAM, sólo se tiene en cuenta esta tarea,

Gracias a esta función, PAM se libera de tener que preocuparse de cómo enviar/recibir información del cliente —cuadros de diálogo, intercambio en un terminal, protocolos de red, etc.—.

Configuración

Enfoques de la organización de la configuración

La configuración de Linux-PAM puede organizarse siguiendo dos esquemas diferentes:

- ✓ Poner toda la configuración en el fichero `/etc/pam.conf` —la más antigua—.
- ✓ Colocar la configuración de cada servicio en ficheros separados bajo el directorio `/etc/pam.d/`. Este tipo de organizaciones ha ido ganando adeptos, y proyectos como Apache también hacen uso de un esquema de este tipo.
- ✓ Una combinación de las dos anteriores: por ejemplo, leer primero `/etc/pam.d/` y luego `/etc/pam.conf` (como en el caso de Red Hat Linux).

Afortunadamente, en todos los casos se utiliza la misma sintaxis, con la salvedad de que en el caso de `/etc/pam.conf` hay que indicar el servicio al que pertenece cada directiva (con `/etc/pam.d/` esta información está implícita en el nombre del fichero de configuración).

En aras de la simplicidad, en adelante se supondrá que se usa el enfoque combinado, si bien este es un detalle que, conceptualmente, carece de importancia.

```
$ ls /etc/pam.d/
```

chage	chsh	groupadd	kde-np	other	shadow	su	system-auth	xdm
chfn	cups	kde	login	passwd	sshd	sudo	useradd	xserver

Organización de la configuración en ficheros separados

Reglas

Los ficheros de configuración de PAM están compuestos por una serie de reglas a aplicar, una por línea (aunque pueden dividirse en varias usando el carácter `\`). Se ignoran todas las líneas que comiencen por el carácter `#`.

Todas estas reglas tienen la siguiente forma:

```
servicio* tipo control ruta [argumentos]
```

A continuación, se va a profundizar en el significado y la sintaxis de cada uno de estos campos. Hay que mencionar que, exceptuando los casos de ruta y argumentos (depende del módulo), en el resto de los campos no se hacen distinciones entre mayúsculas y minúsculas.

Pero antes, y para que lector se haga una idea de la estructura de este fichero, un pequeño ejemplo. Esta configuración pertenece al servicio login de Red Hat Linux `/etc/pam.d/login`.

```
##PAM-1.0
auth      required pam_securetty.so
auth      required pam_stack.so service=system-auth
auth      required pam_nologin.so
account   required pam_stack.so service=system-auth
password  required pam_stack.so service=system-auth
session   required pam_stack.so service=system-auth
session   optional pam_console.so
```

Servicio

Este primer campo indica el nombre de la aplicación/servicio correspondiente, como por ejemplo `login`, `su` o `smtp`. Tal y como se explicó anteriormente, se utiliza sólo en `/etc/pam.conf`, ya que en el caso de la configuración mediante ficheros individuales el nombre servicio se encuentra implícito en el del fichero.

Existe un nombre de servicio reservado, `other`, que se utiliza para especificar reglas por defecto. Así, en el caso de que, por ejemplo, no hubiese reglas para `smtp`, se le aplicarían las asociadas al “servicio” `other`.

Tipo

Como se vio anteriormente, PAM puede dividir su actividad en cuatro tareas de gestión. Y tipo expresa, precisamente, el área al que se destina esta regla. Puede adoptar uno de estos valores: `auth`, `account`, `session` o `password`. Como ya se había mencionado con anterioridad, los módulos correspondientes a un mismo área forman una pila.

Control

Este campo indica a PAM qué hacer en caso de éxito/fallo del módulo de la regla en cuestión, ejecutándose en serie todos los del mismo tipo. El orden en la pila hay que tenerlo muy en cuenta a la hora de determinar el valor adecuado para este campo.

Actualmente, pueden usarse dos sintaxis: la más simple consiste en una sola palabra clave, mientras que la otra (*más precisa y compleja*) implica el uso de corchetes y parejas valor-acción. Para facilitar la comprensión, en primer lugar se va a estudiar la sintaxis más simple para, a continuación, explorar su alternativa compleja.

Usando la sintaxis sencilla, este campo puede tomar únicamente cuatro valores diferentes:

- ✓ `required` Indica que es necesario que el módulo tenga éxito para que la pila también lo tenga. Si se produce un fallo, no se notifica hasta que se procesa el resto de la pila.
- ✓ `requisite` En esencia, es igual que el anterior, con la diferencia de que en caso de fallo, el control se devuelve inmediatamente a la aplicación.
- ✓ `sufficient` El éxito en este módulo, si no se ha producido un fallo en los procesados anteriormente en la pila, es “suficiente”. Llegados a este punto, el procesamiento se detiene (*ignorando incluso posibles required posteriores*). Un fallo no siempre resulta definitivo para la pila.
- ✓ `optional` Por lo general, PAM ignora los módulos marcados con este indicador. Su valor será tenido en cuenta sólo en caso de que no se haya llegado a ningún valor concreto de éxito o fracaso (*por ejemplo, PAM IGNORE*).

La sintaxis alternativa de configuración ofrece mayor flexibilidad y funcionalidad, pero introduce cierta complejidad que en muchos casos no es deseable. Se basa en asociar parejas valor-acción. Así, pueden asociarse acciones con los valores devueltos por los módulos.

```
[valor1=acción1 valor2=acción2 valor3=acción3 ...]
```

Los posibles valores de valor_i se encuentran documentados [aquí](#).

En lo que a las partes derechas se refiere, acción_i puede ser un número entero positivo n o cualquiera de estos valores:

`ignore` El valor que devuelve este módulo no se tiene en cuenta.

`bad` Si se trata del primer módulo en fallar, el valor que devuelva la pila será el que devuelva este módulo.

`die` Equivalente al anterior, pero en este caso se devuelve el control a la aplicación de inmediato.

`ok` Si hasta el momento, el estado de la pila conduce a un éxito (`PAM SUCCESS`), el código que devuelve será sobrescrito por el de este módulo. En caso contrario, deja el estado tal y como se lo encuentra.

`done` Funciona del mismo modo que ok, con la salvedad de que, llegado a este punto, devolverá el control a la aplicación.

`reset` Se olvida el estado de la pila hasta el momento y se sigue con el siguiente módulo de la pila.

El valor entero positivo n que antes se mencionó, indica que deben saltarse los siguientes n módulos. Esto permite al administrador tener, en cierto modo, control sobre el flujo de ejecución del proceso.

Para ilustrar la relación entre una y otra sintaxis de configuración, a continuación se expresan los posibles valores de la sintaxis sencilla en función de expresiones de la sintaxis compleja.

required	equivale a	<code>[success=ok newauthok_reqd=ok ignore=ignore default=bad]</code> .
requisite	equivale a	<code>[success=ok newauthok_reqd=ok ignore=ignore default=die]</code> .
sufficient	equivale a	<code>[success=done new authok reqd=done default=ignore]</code> .
optional	equivale a	<code>[success=ok new authtok reqd=ok default=ignore]</code> .

Ruta

Este campo contiene la ruta del módulo que se va a utilizar: si empieza por el carácter `'/'` se indica una ruta absoluta. En otro caso, será relativa a `/lib/security/`.

Argumentos

Se trata de argumentos que pueden ser pasados al módulo para su operación. Generalmente, los argumentos son específicos para cada módulo y deberían estar documentados. Si se pasara un argumento no válido, el módulo lo ignoraría, aunque debería usar `syslog` para informar del error.

En cuanto a la sintaxis, hay que señalar que si se quieren introducir espacios en blanco en un argumento, éste deberá ir encerrado entre corchetes. Si lo que se pretende hacer es pasar un corchete como parte del parámetro, habrá hacer uso del carácter de escape `'\'`.

```
squid auth required pam_mysql.so user=passwd_query passwd=mada \
db=eminence [query=select user name from internet service where \
username_='%' and password=PASSWORD('%p') and service='web_proxy']
```

En el ejemplo anterior puede observarse como se combinan distintos elementos para formar una regla bastante compleja. Nótese que los valores `%u` y `%p` son sustituidos por el nombre de usuario y la contraseña respectivamente.

Algunos módulos disponibles

Hasta el momento se han estudiado los fundamentos, la arquitectura y la configuración de PAM. Sin embargo, poco se ha dicho acerca de los módulos que ofrece. Esta sección está dedicada a describir, sin entrar en demasiado detalle, algunos de los módulos que vienen con la distribución oficial de PAM en su versión 0.77.

Esto no pretende ser una guía de referencia exhaustiva, por lo que para detalles más concretos sobre estos módulos se remite al lector a la documentación del proyecto. Simplemente, se expondrá una breve descripción de cada módulo y se mostrarán cuáles son los principales argumentos que admiten.

pam console.so

Grupo `session y auth`.

Este módulo permite el cambio de los permisos y de los propietarios de los ficheros indicados en `/etc/security/console.perms` cuando un usuario accede al sistema mediante una consola física y no hay ningún otro usuario registrado en el sistema. Cuando el usuario abandona la sesión, los permisos y propietarios originales se restauran.

En el caso de que varios usuarios trabajasen al mismo tiempo en la consola física, los ficheros sólo se le asignarían al primero en registrarse en el sistema, aunque esto no suele ocurrir.

Los parámetros admitidos por este módulo son:

`allow nonroot tty` Bloquea la consola y cambia los permisos incluso si el propietario del terminal no es el superusuario.

`permsfile=fichero` Permite especificar un fichero alternativo al `/etc/security/console.perms` del cual leer la base de datos de permisos.

`fstab=fichero` Permite indicar un fichero alternativo al `fstab`. El fichero `fstab` almacena información relativa a los dispositivos que se pueden montar en el sistema, indicando donde deben de montarse y el sistema de ficheros en el que se encuentra.

pam cracklib.so

Grupo `password`

Se trata de un módulo preventivo que se encarga de notificar al usuario de la debilidad de su clave cuando ve que puede “romperse” usando un diccionario. Este módulo entra en funcionamiento cuando se establece o modifica la clave de un usuario.

Básicamente este módulo hace pasar la clave por la biblioteca `cracklib`. En caso de que pase como clave segura, intentará las siguientes comprobaciones, comparando la nueva clave con la antigua:

Palíndromo Comprueba que la nueva contraseña no sea la vieja al revés.

Cambio de mayúsculas Realiza varias combinaciones cambiando mayúsculas por minúsculas y viceversa.

Similar Comprueba que las claves se diferencian en más de `difok` caracteres.

Simple Verifica el tamaño de la clave.

Rotada Comprueba que la clave nueva no es una rotación de la antigua.

Ya usada Se asegura de que la clave no se ha usado con anterioridad. Las contraseñas ya usadas se almacenan en `/etc/security/opasswd`.

Así mismo este módulo permite un gran número de parámetros, ahora se mostraran los más importantes:

`debug` Muestra información mas detallada a través de `syslog`. Esta opción no imprimirá las claves en el fichero de log.

`retry=N` Permite especificar el número de veces que se volverá a pedir la clave en caso de que no sea segura. Por defecto es **1**.

`difok=N` Indica el número de caracteres que deben ser diferentes entre la clave anterior y la nueva.

`minlen=N` Indica el número mínimo de caracteres que debe tener una clave.

`use authok` Evita que el usuario introduzca la nueva contraseña tomándola del módulo que se encuentra por delante en la pila.

pam deny.so

Grupo `account`, `authentication`, `password` y `session`.

El objetivo de este módulo es producir un fallo siempre. Si este módulo es el único que se encuentra en la pila, se considerará que ésta ha fallado.

pam env.so

Grupo `authentication`

Permite establecer las variables de entorno por defecto o sustituir los valores de las variables ya establecidas cuando un usuario se registra en el sistema. El fichero donde se definen dichas variables se encuentra en `/etc/security/pam_env.conf`.

Las cláusulas de este fichero son de la siguiente manera:

```
VARIABLE [DEFAULT=[valor]] [OVERRIDE=[valor]]
```

donde la cláusula `default` especificará el valor a tomar por defecto y `override` el valor por el que será sustituido el contenido de la variable.

Este módulo admite cuatro parámetros:

`debug` Muestra información mas detallada sobre el módulo en syslog.

`conf=archivo` Especifica el fichero de configuración alternativo.

`envfile=archivo` Especifica el fichero alternativo a `/etc/environment` que contiene las variables de entorno establecidas para el sistema.

`readenv=0/1` Especifica si se lee o no el fichero con las variables de entorno. Por defecto sí se lee.

pam limits.so

Grupo `session`

Controla los límites impuestos en el fichero `/etc/security/limits.conf` a los recursos disponibles. Las limitaciones se pueden aplicar a un usuario, a un grupo o a todos los usuarios del sistema. Los recursos que se pueden limitar desde este módulo van desde el máximo tiempo de CPU asignable, hasta el número máximo de ficheros que puede tener bloqueados simultáneamente.

Este módulo permite los siguiente parámetros:

`debug` Muestra información añadida a través de syslog.

`conf=archivo` Permite especificar un fichero de configuración alternativo.

`change uid` Permite cambiar el `uid` real para los usuarios que se han establecido límites. Esta opción se usa principalmente en sistemas en los que al poner un límite de 0 procesos el usuario no puede ni abrir la sesión.

`utmp early` Corrige el problema generado por algunas aplicaciones que intentan crear `utmp` para el usuario antes de que este entre en el sistema.

La sintaxis del fichero de configuración es la siguiente:

```
<dominio> <tipo> <elemento> <valor>
```

donde `<dominio>` es el nombre de usuario, nombre de grupo, el comodín `*` o el comodín `%`. `<tipo>` indicara si se trata de un limite duro (`hard`) débil (`soft`) o los dos tipos simultáneamente (`-`). Finalmente, el parámetro `<elemento>` se puede sustituirse por varios valores que indican el recurso que se esta limitando.

pam nologin.so

Grupo `account` y `authentication`

Este módulo sólo deja entrar a los usuarios del sistema si el fichero `/etc/nologin` no existe. En caso contrario, sólo el superusuario puede ingresar en el sistema. Al resto de los usuarios se les mostrará el contenido de dicho archivo.

`Pam nologin` permite dos parámetros:

`successok` Devuelve PAM SUCCESS en vez de PAM IGNORE en caso de no encontrar el fichero `/etc/nologin`.

`file=archivo` Permite indicar un fichero alternativo a `/etc/nologin`.

pam permit.so

Grupo `account`, `authentication`, `password` y `session`.

`Pam permit` funciona justamente al contrario del módulo `pam deny.so`. Para cada llamada al módulo este devuelve un acierto, `PAM SUCCESS`. Por esta razón este módulo es muy inseguro y debe de ser usado con precaución extrema.

pam rootok.so

Grupo authentication

`Pam rootok.so` devuelve un acierto siempre que el identificador real del usuario sea 0 (el usuario `root`). Con ello se consigue que el usuario `root` no necesite introducir la clave para acceder a los servicios asociados a este módulo. Al usar `pam rootok` hay que tener mucho cuidado ya que plantea un grave problema de seguridad. Si asociáramos `pam rootok` a un proceso que se arranca con el sistema, como por ejemplo `login`, el `uid` real será 0 y por lo tanto acertará siempre, independientemente de que intentemos registrarnos con un usuario con `uid` distinto de 0.

pam securetty.so

Grupo authentication

Sirve para limitar las consolas en las que se puede autenticar el usuario `root`. El fichero `/etc/securetty` contiene una lista de consolas seguras.

pam stack.so

Grupo account, authentication, password y session.

Este módulo permite la asociación en pila de varios módulos PAM devolviendo un acierto, en caso de que toda la pila devuelva un acierto. En caso de que uno de los módulos de la pila devuelva un error, `pam stack` devolverá el código de error devuelto por el módulo que ha fallado.

pam wheel.so

Grupo authentication Limita la autenticación como `root` a los usuarios del grupo `wheel`.

Los parámetros admitidos por `pam wheel` son los siguientes:

debug Muestra información añadida a través del `syslog`.

use uid Modifica el comportamiento del módulo usando el `uid` del proceso y no el nombre de usuario de `getlogin`.

trust Con esta opción se consigue que los usuarios que pertenecen al grupo `wheel` cambiarse por el usuario `root` sin usar la clave. Debe de usarse con mucho cuidado.

deny Este parámetro hace que el módulo se comporte al revés denegando a los usuarios del grupo `wheel` la posibilidad de convertirse en `root`.

group=XXXX Permite especificar los grupos a los que se les permite la autenticación.

pam xauth.so

Grupo session

Se encarga de redireccionar las “cookies” de autenticación de un usuario a otro en el sistema X-Window. Esto se usa para que, cuando un usuario se haga pasar por otro mediante el uso del comando `su` o alguno de sus derivados, pueda seguir lanzando aplicaciones como el nuevo usuario sin necesidad de salir y volver a entrar en el entorno X-Window.

Ejemplos de configuración

Ahora que ya se han estudiado tanto los fundamentos de configuración como el cometido de los módulos más destacados de Linux-PAM, parece buena idea fijar la atención en una serie de ejemplos reales, con el fin de ilustrar los conceptos vistos con anterioridad.

Para ello se comentará a continuación la configuración que da Red Hat Linux a algunos de sus servicios. Concretamente, `login`, `passwd`, `su` y `other`.

Sin embargo, en primer lugar se va a tener en cuenta un servicio “artificial”, `system-auth`. Este engloba políticas comunes a muchos otros servicios que lo incluyen en su configuración haciendo uso del módulo `pam_stack`. La ventaja de este esquema es que, en caso de querer cambiar el comportamiento común de varios servicios, tan sólo es necesario modificar `system-auth`.

```
##PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
### auth ###
# Inicializa las variables de entorno definidas en /etc/security/pam env.conf.
auth required /lib/security/$ISA/pam env.so
# Autentica al usuario al estilo 'unix' tradicional (/etc/passwd). Se detiene
# aquí si tiene éxito.
# nullok: permite contraseñas en blanco.
# likeauth: pam_sm_setcred() devuelve lo mismo que pam_sm_authenticate()
auth sufficient /lib/security/$ISA/pam_unix.so likeauth nullok
# Si se llega hasta aquí sin éxito, se falla.
auth required /lib/security/$ISA/pam_deny.so
### account ###
# Basándose en /etc/shadow, determina el estado de la cuenta de usuario (si ha
# expirado, debe cambiar la contraseña, etc.).
account required /lib/security/$ISA/pam_unix.so
### password ###
# Comprueba que la clave no es vulnerable a un ataque simple basado en
# diccionario.
# retry: 3 intentos para cambiar.
# type: modifica el mensaje. Para type=XXX, sería 'New XXX password: '.
password required /lib/security/$ISA/pam_cracklib.so retry=3 type=
# Actualiza la contraseña.
# nullok: permite cambiar una contraseña en blanco. Sin este parámetro,
# una clave en blanco se interpreta como señal de que la cuenta se
# encuentra desactivada.
# use authtok: toma como nueva contraseña la obtenida por el módulo anterior
# (cracklib en este caso).
# md5: usa la función MD5 para cifrar la contraseña.
password sufficient /lib/security/$ISA/pam_unix.so nullok use_authtok md5 shadow
# Si se llega hasta aquí sin éxito, se falla.
password required /lib/security/$ISA/pam_deny.so
### session ###
# Limita los recursos para la sesión del usuario en base al contenido del
# archivo /etc/security/limits.conf.
session required /lib/security/$ISA/pam_limits.so
# Registra el acceso al servicio en syslog, tanto al principio como al final
# de la sesión.
session required /lib/security/$ISA/pam_unix.so
```

Sólo un apunte más, los comentarios de los ficheros que se presentan en esta sección han sido añadido por los autores de este documento, por lo que es probable que introduzcan algunas imprecisiones.

login

El programa `login` ofrece a los usuarios la posibilidad de abrir una sesión en el sistema.

Este es uno de los puntos de acceso más sensibles y que, por lo general, ofrece un mayor riesgo para la integridad del sistema. Así pues, la configuración de Linux-PAM en este (como en todos los servicios, a decir verdad) debe estar especialmente cuidada.

Por lo general, `login` pide un nombre de usuario y una contraseña. Después de llevar a cabo una serie de comprobaciones e inicializaciones, al usuario se le ofrece un intérprete de comandos (`shell`).

La configuración de este servicio en Red Hat Linux hace uso intensivo del servicio “artificial” `system-auth`, siendo idénticas las pilas `account` y `password`. Las diferencias radican, principalmente, en que al superusuario sólo se le permite acceder desde terminales físicas (`tty`), se deshabilita la entrada si

existe el fichero `/etc/nologin` y se realizan cambios de permisos/propietario según lo indicado en `/etc/security/console.perms`.

```
##PAM-1.0
### auth ###
# Deshabilita el 'login' para el root exceptuando los tty's.
auth required pam_securetty.so
# Procesa los módulos 'auth' del servicio 'system-auth'.
auth required pam_stack.so service=system-auth
# Deshabilita la entrada de cualquier usuario que no sea el 'root' cuando el
# fichero /etc/nologin existe.
auth required pam_nologin.so
### account ###
# Procesa los módulos 'account' del servicio 'system-auth'.
account required pam_stack.so service=system-auth
### password ###
# Procesa los módulos 'password' del servicio 'system-auth'.
password required pam_stack.so service=system-auth
### session ###
# Procesa los módulos 'session' del servicio 'system-auth'.
session required pam_stack.so service=system-auth
# Cambia permisos y propietarios de ciertos ficheros según se indica en
# /etc/security/console.perms si el acceso se realiza por medio de una
# consola 'física'. Los permisos/propietarios originales se restauran al
# salir.
session optional pam_console.so
```

passwd

`passwd` es la utilidad que permite modificar la contraseña de las cuentas asociadas a los usuarios y a los grupos. Actuando como un usuario normal del sistema solamente podrá modificar la propia contraseña, mientras que el superusuario podrá modificar la contraseña de cualquier cuenta. Así mismo, solamente el administrador de un grupo podrá cambiar la contraseña del grupo.

Además de permitir el cambio de la contraseña, `passwd` permite modificar otros parámetros asociados a una cuenta de usuario como son el nombre, el shell de inicio o la fecha de caducidad de la clave.

```
##PAM-1.0
### auth ###
# Procesa los módulos 'authentication' del servicio 'system-auth'
auth required pam_stack.so service=system-auth
### account ###
# Procesa los módulos 'account' del servicio 'system-auth'
account required pam_stack.so service=system-auth
### password ###
# Procesa los módulos 'password' del servicio 'system-auth'
password required pam_stack.so service=system-auth
```

su

`su` es una aplicación que permite convertirse en otro usuario una vez que tenemos una sesión ya abierta. Cuando se invoca sin parámetros el `su` actúa como si se quisiera convertir al superusuario, `root`. En caso contrario su intentara convertirse en el usuario pasado por parámetro.

A continuación se mostrara el fichero de configuración del pam para la utilidad `su` suministrado en RedHat.

```
##PAM-1.0
### auth ###
# Permite que el usuario root use la utilidad su sin suministrar
# ninguna contraseña.
auth sufficient /lib/security/$ISA/pam_rootok.so
# Descomentar la siguiente línea para indicar que los usuarios
# pertenecientes al grupo wheel sean usuarios en los que se confía y
# puedan realizar un 'su' sin suministrar contraseña.
#auth sufficient /lib/security/$ISA/pam_wheel.so trust use_uid
# Descomentar la siguiente línea para indicar que los usuarios
# pertenecientes al grupo wheel son los únicos usuarios a los que se
```

```
# les permite usar la utilidad 'su'
#auth required /lib/security/$ISA/pam_wheel.so use_uid
# Procesa los módulos 'authentication' del servicio 'system-auth'.
auth required /lib/security/$ISA/pam_stack.so service=system-auth
### account ###
# Procesa los módulos 'account' del servicio 'system-auth'.
account required /lib/security/$ISA/pam_stack.so service=system-auth
### password ###
# Procesa los módulos 'password' del servicio 'system-auth'.
password required /lib/security/$ISA/pam_stack.so service=system-auth
### session ###
# Procesa los módulos 'session' del servicio 'system-auth'.
session required /lib/security/$ISA/pam_stack.so service=system-auth
# Carga el modulo xauth como opcional, permitiendo que el usuario al
# que se ha cambiado mediante el su pueda seguir lanzando aplicaciones X-Window
# en la sesión abierta por el usuario original.
session optional /lib/security/$ISA/pam_xauth.so
```

other

Tal y como se explicó con anterioridad, cuando PAM no encuentra un fichero de configuración para un servicio en particular, aplica las reglas del servicio especial `other`. Por tanto, puede decirse que se trata de la política por defecto.

Por motivos de seguridad, es aconsejable que este servicio deniegue cualquier acceso. Así, si Linux-PAM no encuentra configuración para un servicio, simplemente denegará el acceso, minimizando posibles amenazas.

Con esta premisa, la configuración resulta tremendamente simple, tal y como puede apreciarse en las siguientes líneas.

```
##PAM-1.0
# Por defecto, se deniega el acceso a cualquiera
auth required /lib/security/$ISA/pam_deny.so
account required /lib/security/$ISA/pam_deny.so
password required /lib/security/$ISA/pam_deny.so
session required /lib/security/$ISA/pam_deny.so
```

Valores devueltos por los módulos de PAM

Cada módulo de PAM devuelve un valor al ser invocado. Este indica si se ha producido algún fallo o si, por el contrario, todo ha ido bien.

PAM permite especificar comportamientos para cada uno de estos posibles valores en el fichero de [configuración](#). Para ello, se utiliza la sintaxis de pares `valor=acción`, colocando en la parte izquierda de la asignación la etiqueta del valor (*cualquiera de los valores descritos a continuación sin el prefijo PAM y escrito en minúscula; por ejemplo, `auth err`*).

authentication

PAM AUTH ERR El usuario no se autenticó.

PAM AUTHINFO UNAVAIL El módulo no fue capaz de acceder a la información de autenticación. Esto se debe generalmente a un fallo hardware o de acceso a la red.

PAM CRED ERR Este valor se devuelve cuando el módulo no pudo establecer las credenciales del usuario.

PAM CRED EXPIRED Indica que las credenciales del usuario han caducado.

PAM CRED INSUFFICIENT Por alguna razón, la aplicación no tiene suficientes credenciales para autenticar al usuario.

PAM CRED UNAVAIL El módulo no pudo obtener las credenciales de los usuarios.

PAM MAXTRIES Los módulos devuelven este valor cuando han alcanzado el número máximo de reintentos de autenticar al usuario.

PAM USER UNKNOWN El nombre de usuario es desconocido para el sistema de 'authentication'.

PAM SUCCESS Este valor se devuelve cuando el módulo ha tenido éxito.

account

PAM ACCT EXPIRED La cuenta del usuario ha caducado y ya no puede volver a acceder con dicha cuenta.

PAM AUTH ERR El usuario no se autenticó.

PAM AUTHTOKEN REQD El token de autenticación ha caducado. Antes de volver a llamar a esta función se debería pedir un nuevo token.

PAM SUCCESS Este valor se devuelve cuando el módulo ha tenido éxito.

PAM USER UNKNOWN El nombre de usuario es desconocido para el sistema de 'account'.

password

PAM AUTHOK DISABLE AGING El token de autenticación ha sido desactivado.

PAM AUTHOK ERR El módulo fue incapaz de obtener un nuevo token de autenticación.

PAM AUTHOK RECOVERY ERR No se consiguió leer el anterior token de autenticación.

PAM AUTHOK LOCK BUSY El token de autenticación estaba bloqueado cuando se intentó cambiar.

PAM PERM DENIED Permiso denegado.

PAM SUCCESS Este valor se devuelve cuando el módulo ha tenido éxito.

PAM TRY AGAIN Las comprobaciones anteriores fallaron. Se pide que se vuelva a hacer la misma llamada.

PAM USER UNKNOWN El nombre de usuario es desconocido para el sistema de 'password'.

session

PAM SESSION ERR Se ha producido un fallo al intentar abrir o cerrar la sesión.

PAM SUCCESS Este valor se devuelve cuando el módulo ha tenido éxito.

Anexo II - proftpd.conf

```
#
# /etc/proftpd/proftpd.conf -- This is a basic ProFTPD configuration file.
# To really apply changes reload proftpd after modifications.
#

# Includes DSO modules
Include /etc/proftpd/modules.conf

# Set off to disable IPv6 support which is annoying on IPv4 only boxes.
UseIPv6                on
# If set on you can experience a longer connection delay in many cases.
IdentLookups            off

ServerName              "Debian"
ServerType              standalone
DeferWelcome            off

MultilineRFC2228        on
DefaultServer           on
ShowSymlinks            on

TimeoutNoTransfer        600
TimeoutStalled          600
TimeoutIdle             1200

DisplayLogin            welcome.msg
DisplayChdir            .message true
ListOptions             "-l"

DenyFilter              \ *.*

# Use this to jail all users in their homes
# DefaultRoot           ~

# Users require a valid shell listed in /etc/shells to login.
# Use this directive to release that constrain.
# RequireValidShell     off

# Port 21 is the standard FTP port.
Port                    21

# In some cases you have to specify passive ports range to by-pass
# firewall limitations. Ephemeral ports can be used for that, but
# feel free to use a more narrow range.
# PassivePorts          49152 65534

# If your host was NATted, this option is useful in order to
# allow passive tranfers to work. You have to use your public
# address and opening the passive ports used on your firewall as well.
# MasqueradeAddress     1.2.3.4

# This is useful for masquerading address with dynamic IPs:
# refresh any configured MasqueradeAddress directives every 8 hours
<IfModule mod_dynmasq.c>
# DynMasqRefresh 28800
</IfModule>

# To prevent DoS attacks, set the maximum number of child processes
# to 30. If you need to allow more than 30 concurrent connections
# at once, simply increase this value. Note that this ONLY works
# in standalone mode, in inetd mode you should use an inetd server
# that allows you to limit maximum number of processes per service
# (such as xinetd)
MaxInstances            30

# Set the user and group that the server normally runs at.
User                    proftpd
Group                   nogroup

# Umask 022 is a good standard umask to prevent new files and dirs
# (second parm) from being group and world writable.
Umask                   022 022
# Normally, we want files to be overwriteable.
AllowOverwrite          on
```

```
# Uncomment this if you are using NIS or LDAP via NSS to retrieve passwords:
# PersistentPasswd          off

# This is required to use both PAM-based authentication and local passwords
# AuthOrder                 mod_auth_pam.c* mod_auth_unix.c

# Be warned: use of this directive impacts CPU average load!
# Uncomment this if you like to see progress and transfer rate with ftpwho
# in downloads. That is not needed for uploads rates.
#
# UseSendFile               off

TransferLog /var/log/proftpd/xferlog
SystemLog   /var/log/proftpd/proftpd.log

<IfModule mod_quotatab.c>
QuotaEngine off
</IfModule>

<IfModule mod_ratio.c>
Ratios off
</IfModule>

# Delay engine reduces impact of the so-called Timing Attack described in
# http://security.lss.hr/index.php?page=details&ID=LSS-2004-10-02
# It is on by default.
<IfModule mod_delay.c>
DelayEngine on
</IfModule>

<IfModule mod_ctrls.c>
ControlsEngine      off
ControlsMaxClients  2
ControlsLog         /var/log/proftpd/controls.log
ControlsInterval    5
ControlsSocket       /var/run/proftpd/proftpd.sock
</IfModule>

<IfModule mod_ctrls_admin.c>
AdminControlsEngine off
</IfModule>

#
# Alternative authentication frameworks
#
#Include /etc/proftpd/ldap.conf
#Include /etc/proftpd/sql.conf

#
# This is used for FTPS connections
#
#Include /etc/proftpd/tls.conf

#
# Useful to keep VirtualHost/VirtualRoot directives separated
#
#Include /etc/proftpd/virtuals.conf

# A basic anonymous configuration, no upload directories.

# <Anonymous ~ftp>
#   User          ftp
#   Group          nogroup
#   # We want clients to be able to login with "anonymous" as well as "ftp"
#   UserAlias      anonymous ftp
#   # Cosmetic changes, all files belongs to ftp user
#   DirFakeUser    on ftp
#   DirFakeGroup   on ftp
#
#   RequireValidShell      off
#
#   # Limit the maximum number of anonymous logins
#   MaxClients             10
#
#   # We want 'welcome.msg' displayed at login, and '.message' displayed
```

```
# # in each newly chdired directory.
# DisplayLogin          welcome.msg
# DisplayChdir          .message
#
# # Limit WRITE everywhere in the anonymous chroot
# <Directory *>
#   <Limit WRITE>
#     DenyAll
#   </Limit>
# </Directory>
#
# # Uncomment this if you're brave.
# # <Directory incoming>
# #   # Umask 022 is a good standard umask to prevent new files and dirs
# #   # (second parm) from being group and world writable.
# #   Umask          022 022
# #   <Limit READ WRITE>
# #     DenyAll
# #   </Limit>
# #   <Limit STOR>
# #     AllowAll
# #   </Limit>
# # </Directory>
#
# </Anonymous>
```

Anexo III - tls.conf

```
#####Fichero
/etc/proftpd/tls.conf#####
#
# Proftpd sample configuration for FTPS connections.
#
# Note that FTPS impose some limitations in NAT traversing.
# See http://www.castaglia.org/proftpd/doc/contrib/ProFTPD-mini-HOWTO-TLS.html
# for more information.
#

<IfModule mod_tls.c>
  TLSEngine                      on
  TLSLog                        /var/log/proftpd/tls.log
  TLSProtocol                   SSLv23
  #
  # Server SSL certificate. You can generate a self-signed certificate using
  # a command like:
  #
  # openssl req -x509 -newkey rsa:1024 \
  #             -keyout /etc/ssl/private/proftpd.key -out /etc/ssl/certs/proftpd.crt \
  #             -nodes -days 365
  #
  # The proftpd.key file must be readable by root only. The other file can be
  # readable by anyone.
  #
  # chmod 0600 /etc/ssl/private/proftpd.key
  # chmod 0644 /etc/ssl/certs/proftpd.crt
  #
  TLSRSACertificateFile          /etc/ssl/certs/proftpd.crt
  TLSRSACertificateKeyFile       /etc/ssl/proftpd.key
  #
  # CA the server trusts
  #TLSCACertificateFile          /etc/ssl/certs/CA.pem
  # or avoid CA cert
  TLSOptions                     NoCertRequest
  #
  # Authenticate clients that want to use FTP over TLS?
  #
  TLSVerifyClient                off
  #
  # Are clients required to use FTP over TLS when talking to this server?
  #
  TLSRequired                    on
  #
  # Allow SSL/TLS renegotiations when the client requests them, but
  # do not force the renegotiations. Some clients do not support
  # SSL/TLS renegotiations; when mod_tls forces a renegotiation, these
  # clients will close the data connection, or there will be a timeout
  # on an idle data connection.
  #
  TLSRenegotiate                 required off
</IfModule>

#####Fin
/etc/proftpd/tls.conf#####
```

Anexo IV - tls2.conf

```
#####Fichero
/etc/proftpd/tls.conf#####
#
# Proftpd sample configuration for FTPS connections.
#
# Note that FTPS impose some limitations in NAT traversing.
# See http://www.castaglia.org/proftpd/doc/contrib/ProFTPD-mini-HOWTO-TLS.html
# for more information.
#

<IfModule mod_tls.c>
<global>
TLSEngine                                on
TLSLog                                  /var/log/proftpd/tls.log
</global>

TLSProtocol                             SSLv23

<global>
#
# Server SSL certificate. You can generate a self-signed certificate using
# a command like:
#
# openssl req -x509 -newkey rsa:1024 \
#             -keyout /etc/ssl/private/proftpd.key -out /etc/ssl/certs/proftpd.crt \
#             -nodes -days 365
#
# The proftpd.key file must be readable by root only. The other file can be
# readable by anyone.
#
# chmod 0600 /etc/ssl/private/proftpd.key
# chmod 0644 /etc/ssl/certs/proftpd.crt
#
TLRSACertificateFile                     /etc/ssl/certs/proftpd.crt
TLRSACertificateKeyFile                  /etc/ssl/proftpd.key
#
# CA the server trusts
# TLSCACertificateFile                     /etc/ssl/certs/CA.pem
# or avoid CA cert
TLSOptions                               NoCertRequest
#
# Authenticate clients that want to use FTP over TLS?
#
TLSVerifyClient                           off
#
# Are clients required to use FTP over TLS when talking to this server?
#
TLSRequired                               on
#
# Allow SSL/TLS renegotiations when the client requests them, but
# do not force the renegotiations. Some clients do not support
# SSL/TLS renegotiations; when mod_tls forces a renegotiation, these
# clients will close the data connection, or there will be a timeout
# on an idle data connection.
#
TLSPRenegotiate                           required off
</global>
</IfModule>

#####Fin
/etc/proftpd/tls.conf#####
```

TEMA 5

Contenido

1.- Servidores de nombres de dominio.	2
1.1.- Sistema de nombres de dominio.	2
1.1.1.- ¿Cómo es un nombre de dominio?	4
1.1.2.- Jerarquía de nombres de dominio.	5
1.2.- Ventajas del DNS.	6
1.3.- Funcionamiento del DNS.	8
1.4.- DNS Dinámico.....	9
1.5.- Tipos de servidores DNS.	10
1.6.- Servidores raíz.	11
1.7.- Tipos de registros DNS.....	12
1.8.- Funcionamiento del cliente DNS.	14
1.8.1.- Consultas recursivas.....	15
1.8.2.- Consultas iterativas.....	16
1.8.3.- Consultas inversas.....	17
1.9.- Cómo funcionan los DNS preferidos y alternativos.	18
1.10.- Comandos (I).	19
Ejemplos de resolución directa: Resolución de nombre a IP.....	20
1.10.1.- Comandos (II).	21
1.11.- Instalación del servidor DNS BIND.	22
1.11.1.- Archivos de configuración del servidor DNS.....	22
1.11.2.- Arranque y parada del servidor DNS.....	24
1.11.3.- Configuración como caché DNS.	25
1.11.4.- Configuración como DNS maestro.	26
1.11.5.- Configuración como DNS esclavo.....	27
2.- Servicio de directorio.	29
2.1.- ¿Para qué usar un servicio de directorio? 30	
2.2.- Directorio vs DNS.	31
2.3.- Organización del directorio LDAP.	31
2.4.- Integración del servicio de directorio con otros servicios.	33
2.5.- El formato de intercambio de datos LDIF. 33	
2.6.- Instalación de OpenLDAP.	34
2.6.1.- Configuración de OpenLDAP.	35
2.6.2.- Arranque y parada del servidor LDAP.	36
2.6.3.- Administrando un servidor LDAP:	37
2.6.4.- Configuración de los clientes. Instalación de librerías de autenticación.	39
2.6.5.- Probar la autenticación con pamtest.....	40
ANEXO I - Servidores raíz DNS.....	41
ANEXO II - Comprobar funcionamiento servidor DNS BIND	43
ANEXO III - Ejemplo despliegue aplicación web OpenCart	44
ANEXO IV - Instalación y configuración de OpenLDAP.....	46
Instalación de OpenLDAP	46
Configuración inicial de OpenLDAP.....	46
Arranque y parada manual del servidor LDAP	47
Administración de OpenLDAP.....	47
Introducción	47
Paso 1: Cargar plantillas.....	47
Paso 2: Archivo de configuración del esquema básico	47
Paso 3: Creación de unidades organizativas para almacenar cuentas unix	48
ANEXO V - Explorador de directorios LDAP....	51
Instalar phpldapadmin	51
JXplorer - Explorador LDAP en java.....	51
Conexión con el servidor LDAP	52
Creación de usuarios y grupos con jxplorer	53
ANEXO VI - Administración de usuarios y grupos con LDAP	55
Administración mediante scripts	55
Administración con webmin.....	56
Configuración inicial del módulo de Usuarios y grupos LDAP.....	56
Administración de usuarios y grupos de LDAP con webmin	57
Creación masiva de usuarios.....	58
El módulo de servidor LDAP.....	59
LDAP Account Manager	59

Servicios de red implicados en el despliegue de una aplicación web

Caso práctico

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

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

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

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

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

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

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

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

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

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

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

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

—Vale, adiós —dijo **María**—.

—Hasta luego —dijo **Juan**—.

1.- Servidores de nombres de dominio.

Caso práctico

Para poder llevar a buen fin el proyecto, **María** se puso manos a la obra y determinó el siguiente escenario de trabajo para la realización de las dos últimas tareas del proyecto:

- ✓ Sistema Operativo Servidor: Debian GNU/Linux 6.0.
- ✓ Servidor Web: Apache (apache2).
- ✓ Servidor DNS Primario (Maestro): BIND (BIND9).
- ✓ Servidor DNS Secundario (Esclavo): BIND (BIND9).
- ✓ Servidor LDAP: OpenLDAP.
- ✓ Configuración de Red:
 - ➔ Servidor Web: 192.168.200.250.
 - ➔ Cliente de pruebas (desde donde se lanza el navegador): 192.168.200.100.
 - ➔ Servidor DNS Maestro: 192.168.200.250.
 - ➔ Servidor DNS Esclavo: 192.168.200.249.
 - ➔ Servidor OpenLDAP: 192.168.200.248.

María, debe garantizar el correcto funcionamiento de la resolución DNS, por lo tanto, como con otros proyectos, prevé la redundancia del servicio mediante dos servidores DNS, uno actuando de primario y otro de secundario, y debido a sus características se ha decantado por el servidor DNS BIND.

Para la autenticación de usuarios, el cliente tras explicarle las alternativas se ha decantado por LDAP, por lo cual **María** configurará el servicio mediante OpenLDAP debido a sus características.

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

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

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

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

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

1.1.- Sistema de nombres de dominio.

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

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

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

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

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

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

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

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

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

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

Según lo expuesto, y si en tu configuración de red del sistema operativo solamente posees un servidor DNS, entonces: ¿cuál sería el proceso para encontrar la IP de la dirección web:

<http://www.debian.org/distrib/netinst?>

El proceso sería el siguiente:

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

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

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

1.1.1.- ¿Cómo es un nombre de dominio?

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



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

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

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

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

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

A nivel gramatical los dominios deben cumplir una serie de requisitos. Por ejemplo:

- ✓ Sólo pueden estar **compuestos** de **letras** (alfabeto inglés), **números** y **guiones** ("-").
- ✓ **No** pueden **empezar** o **terminar** por **guiones**.
- ✓ Tienen que tener **menos de 63 caracteres** sin incluir la extensión, y más de uno o dos dependiendo del dominio de primer nivel.

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

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

1.1.2.- Jerarquía de nombres de dominio.

El espacio de nombres de dominio (*el universo de todos los nombres de dominio*) está organizado de forma jerárquica. El nivel más alto en la jerarquía es el dominio raíz, que se representa como un punto (".") y el siguiente nivel en la jerarquía se llama dominio de nivel superior (**TLD**). Sólo hay un dominio raíz, pero hay 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 **FQDN**, que es la concatenación de: TLD, dominio de segundo nivel, dominio de tercer nivel, etc., tal como:

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

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

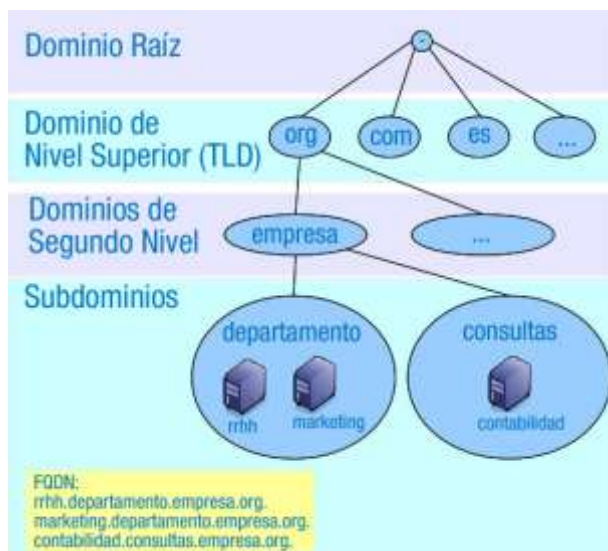
En la siguiente figura puedes ver una parte del espacio de nombres. La raíz del árbol, que se identifica con un punto sencillo, es lo que se denomina dominio raíz y es el origen de todos los dominios. Para indicar que un nombre es FQDN, a veces se termina su escritura en un punto, aunque por lo general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

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

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

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

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



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

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

1.2.- Ventajas del DNS.

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

Albert Einstein

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

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

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

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

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

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

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

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

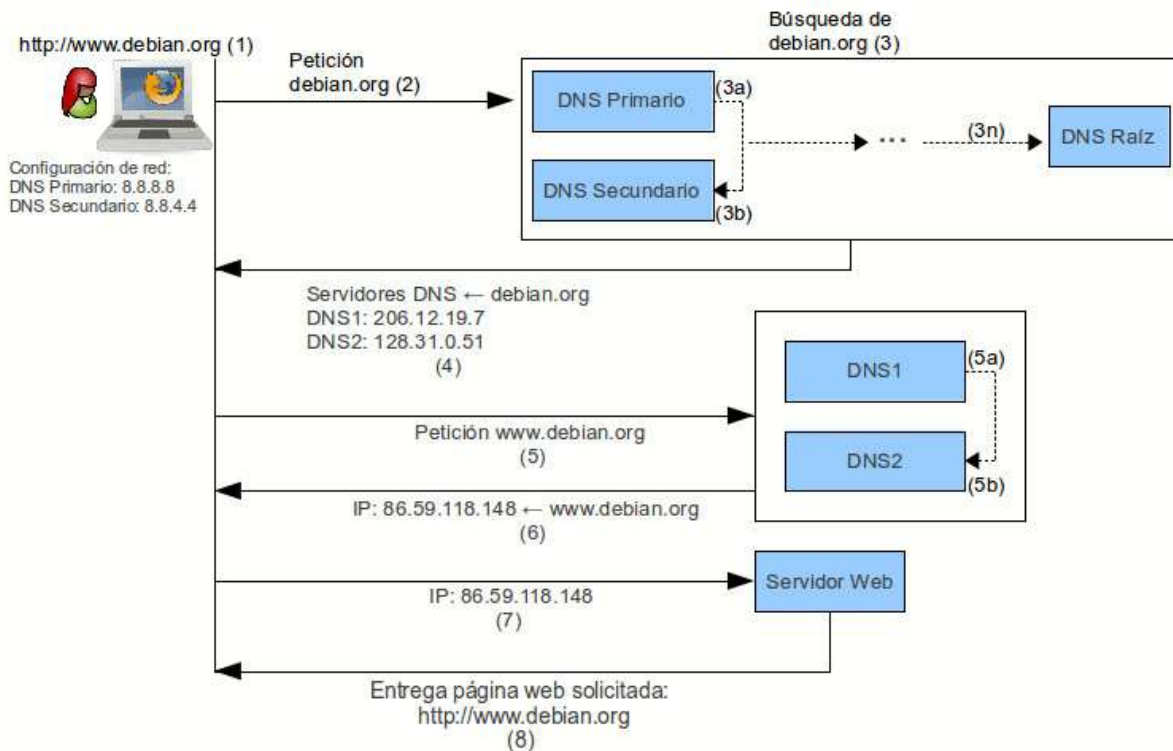
```
ping 192.168.200.100
ping cliente.local
```

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

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

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

1.3.- Funcionamiento del DNS.



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

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

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

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

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

2. Existe un servidor DNS caché en tu red o en tu host: por lo tanto, si un segundo cliente, que tiene configurado este servidor DNS, vuelve a realizar la misma petición, como este servidor tiene la respuesta almacenada en su memoria caché, responderá inmediatamente sin tener que cursar la petición a ningún 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 <https://www.dyndns.com/account/> , http://www.no-ip.com/services/managed_dns/free_dynamic_dns.html y <http://freedns.afraid.org/> . De hecho, hoy en día, los routers que los ISP suelen montar ya poseen la opción de configuración por DNS dinámica.

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

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

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

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

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

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

Cada campo, de cada entrada, puede ir separado por espacios o por tabulados.

Estas entradas solamente serán efectivas en el equipo en el que se modifique el archivo `/etc/hosts`. Así, debes modificar el archivo `/etc/hosts` en cada equipo que quieres que se resuelvan esas entradas.

1.5.- Tipos de servidores DNS.

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



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

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

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

distribuyendo el trabajo entre distintos servidores situados estratégicamente (por zonas geográficas, por ejemplo).

Todos los servidores DNS guardan en la caché las consultas que resolvieron.

Una transferencia de zona puede darse en cualquiera de los casos siguientes:

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

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

donde:

`retransfer` → indica que la acción a realizar es una transferencia.

`proyecto-empresa.local` → es el nombre de la zona que quieres transferir.

1.6.- Servidores raíz.

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



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

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

En el siguiente enlace puedes encontrar más información sobre ICANN.

<http://www.icann.org/>

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

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

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

tienen una de las 13 direcciones IP pueden encontrarse en docenas de ubicaciones físicas distintas). Todos estos servidores almacenan una copia del mismo archivo que actúa como índice principal de las agendas de direcciones de Internet. Enumeran una dirección para cada dominio de nivel principal (.com, .es, etc.) en la que puede encontrarse 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 ejército de EE.UU. Aproximadamente la mitad depende de organizaciones públicas estadounidenses.

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

Anexo I - Los servidores raíz DNS.

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

Información publicada por los servidores raíz DNS.

1.7.- Tipos de registros DNS.

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

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

Formato de los registros de recursos DNS	
Campo	Descripción
Propietario	Indica el nombre de dominio DNS que posee un registro de recursos. Este nombre es el mismo que el del nodo del árbol de la consola donde se encuentra un registro de recursos.
Tiempo de vida (TTL)	Para la mayor parte de los registros de recursos, este campo es opcional. Indica el espacio de tiempo utilizado por otros servidores DNS para determinar cuánto tarda la información en caché en caducar un registro y descartarlo. Por ejemplo, la mayor parte de los registros de recursos que crea el servicio del servidor DNS heredan el TTL mínimo (predeterminado) de 1 hora desde el registro de recurso de inicio de autoridad (SOA) que evita que otros servidores DNS almacenen en caché durante demasiado tiempo. En un registro de recursos individual, puede especificar un TTL específico para el registro que suplantará 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 <i>obligatorio</i> .
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 <i>obligatorio</i> .
Datos específicos del registro	Un campo de longitud variable y <i>obligatorio</i> con información que describe el recurso. El formato de esta información varía según el tipo y clase del registro de recursos.

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

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

Tipos de registros DNS	
Registro	Descripción, sintaxis y ejemplo
A	Descripción: Address (<i>Dirección</i>). Este registro se usa para <u>traducir nombres de hosts a direcciones IP versión 4</u>. Sintaxis: <i>propietario clase ttl A IP_version4</i>. Ejemplo: <code>host1.ejemplo.com IN A 127.0.0.1.</code>
AAAA	Descripción: Address (<i>Dirección</i>). Este registro se usa para <u>traducir nombres de hosts a direcciones IP versión 6</u>. Sintaxis: <i>propietario clase ttl AAAA IP_version6</i>. Ejemplo: <code>host1ipv6.ejemplo.com. IN AAAA 1234:0:1:2:3:4:567:89ab.</code>
CNAME	Descripción: Canonical Name (<i>Nombre Canónico</i>). Se usa para <u>crear nombres de hosts adicionales, o alias</u>. 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 (<i>virtualhosts</i>). Sintaxis: <i>propietario ttl clase CNAME nombreCanónico</i>. Ejemplo: <code>nombrealias.ejemplo.com CNAME nombreverdadero.ejemplo.com.</code> Como se ha comentado anteriormente <code>nombreverdadero.ejemplo.com</code> previamente debe estar definido como registro tipo A.
NS	Descripción: Name Server (<i>Servidor de Nombres</i>). Indica <u>qué servidores de nombres tienen total autoridad sobre un dominio</u> concreto. Cada dominio se puede asociar a una cantidad cualquiera de servidores de nombres. Sintaxis: <i>propietario ttl IN NS nombreServidorNombreDominio</i>. Ejemplo: <code>ejemplo.com. IN NS nombreservidor1.ejemplo.com.</code>
MX	Descripción: Mail eXchange (<i>Registro de Intercambio de Correo</i>). <u>Asocia un nombre de dominio a una lista de servidores de intercambio de correo</u> para ese dominio. Sintaxis: <i>propietario ttl clase MX preferencia hostIntercambiadorDeCorreo</i>. Ejemplo: <code>ejemplo.com. MX 10 servidorcorreo1.ejemplo.com.</code> 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 (<i>Indicador</i>). <u>Traduce direcciones IP en nombres de dominio</u>. También conocido como 'registro inverso', ya que funciona a la inversa del registro "A". Sintaxis: <i>propietario ttl clase PTR nombreDominioDestino</i>. Ejemplo: <code>1.0.0.10.in-addr.arpa. PTR host.ejemplo.com.</code>
SOA	Descripción: Start Of Authority (<i>Autoridad de la zona</i>). Proporciona <u>información sobre el servidor DNS primario</u> de la zona. Sintaxis: <i>propietario clase SOA servidorNombres personaResponsable (numeroSerie intervaloActualización intervaloReintento caducidad tiempoDeVidaMínimo)</i>. 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"`.

En el siguiente enlace encontrarás más información sobre el registro SPF.

<http://www.openspf.org/>

1.8.- Funcionamiento del cliente DNS.

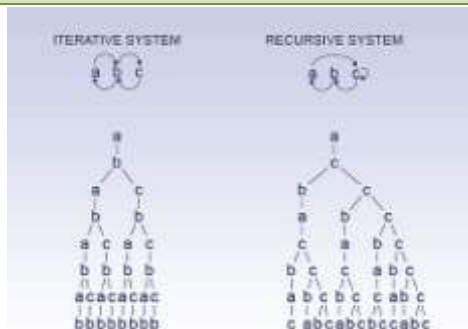
"Los sabios son los que buscan la sabiduría; los necios piensan ya haberla encontrado."

Napoleón Bonaparte

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

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

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



Las consultas DNS se resuelven de diferentes formas:

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

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

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

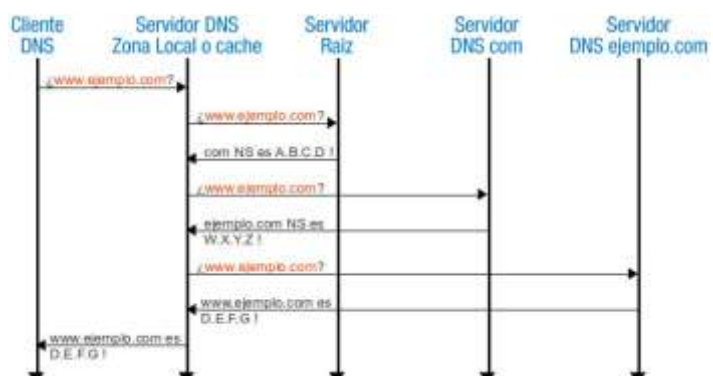
1.8.1.- Consultas recursivas.

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

"René Descartes"

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

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



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

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

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

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

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

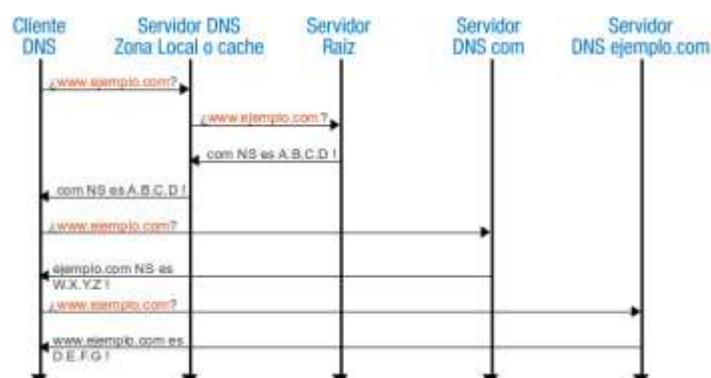
1.8.2.- Consultas iterativas.

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

"Confucio"

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

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



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

Una solicitud iterativa de un cliente informa al servidor DNS de que el cliente espera la mejor respuesta que el servidor 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.

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

"Miguel de Unamuno

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

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

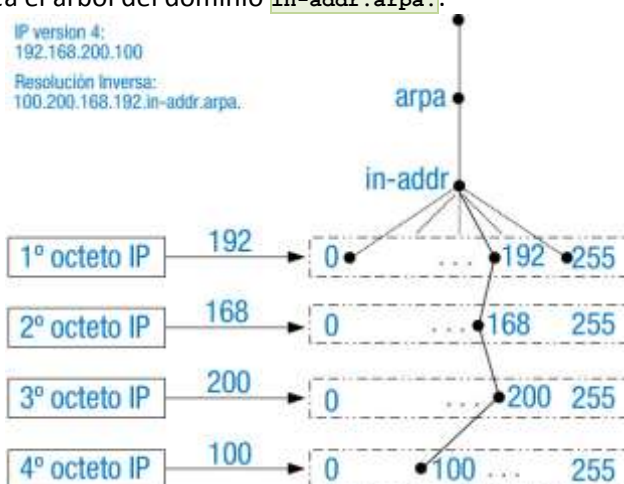
Entonces, para resolver este problema, en el estándar DNS se definió y se reservó un dominio especial para las IP versión 4, el dominio `in-addr.arpa`, en el espacio de nombres DNS de Internet con el fin de proporcionar una forma práctica y confiable para realizar las consultas inversas. Al crear el

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

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

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

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

1.9.- Cómo funcionan los DNS preferidos y alternativos.

El servidor DNS preferido es aquel con el que el cliente prueba en primer lugar. También es el servidor en el que el cliente 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`.

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

1.10.- Comandos (I).

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

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

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

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

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

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

;; ANSWER SECTION:
ftp.rediris.es.                4888    IN      CNAME   zeppo.rediris.es.
zeppo.rediris.es.             5364    IN      A       130.206.1.5

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

alumno@servidor-fp->
```

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

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

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

Ejemplos de resolución directa: Resolución de nombre a IP.

1. Comando `nslookup`:

Para consultar la dirección IP del ordenador `ftp.rediris.es`, basta con ejecutar:

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

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

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

2. Comando `host`:

Para consultar la dirección IP del ordenador `ftp.rediris.es`, basta con ejecutar:

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

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

3. Comando `dig`:

Para consultar la dirección IP del ordenador `ftp.rediris.es`, basta con ejecutar:

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

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

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

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

;; Query time: 76 msec
```

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

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

1.10.1.- Comandos (II).

Ejemplos de resolución inversa: Resolución de IP a nombre.

1. Comando **nslookup**:

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

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

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

Authoritative answers can be found from:
```

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

2. Comando **host**:

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

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

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

3. Comando **dig**:

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

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

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

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

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

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

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

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

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

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

Authoritative answers can be found from:

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

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

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

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

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

alumno@servidor-fp:~$
```

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

1.11.- Instalación del servidor DNS BIND.

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

✓ Hostname: `debian-servidor-fp`.

✓ IP: `192.168.200.250`.

1. Actualiza los repositorios del sistema operativo.

```
root@debian-servidor-fp:~# apt-get update
```

NOTA: es necesario para el buen funcionamiento del comando que tengas configurado correctamente la conexión a Internet.

2. Actualiza el sistema operativo.

```
root@debian-servidor-fp:~# apt-get upgrade
```

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

```
root@debian-servidor-fp:~# apt-get install bind9 bind9utils
```

NOTA: La instalación crea el usuario `bind` que ejecuta el servicio dns denominado `named`.

4. Verifica que el servidor **bind9** está activo.

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

5. Verifica en qué puertos (Número utilizado en las comunicaciones cliente/servidor, en transmisiones TCP o UDP, comprendido entre 1 y 65535, que indica por dónde tiene lugar la conexión con un servidor. Están estandarizados, esto es, un servidor suele estar activo siempre por definición en un puerto determinado, pero éste puede que sea modificado en la configuración del servidor. Por ejemplo, un servidor web espera en el puerto TCP 80) TCP y UDP está activo el servidor **bind9**, para ello comprueba el servicio **named**:

```
root@debian-servidor-fp:~# netstat -natp | grep named
tcp 0 0 192.168.200.250:53 0.0.0.0:* LISTEN 1442/named
tcp 0 0 127.0.0.1:53 0.0.0.0:* LISTEN 1442/named
tcp 0 0 127.0.0.1:953 0.0.0.0:* LISTEN 1442/named
tcp6 0 0 :::53 :::* LISTEN 1442/named
tcp6 0 0 ::1:953 :::* LISTEN 1442/named
root@debian-servidor-fp:~# netstat -naup | grep named
udp 0 0 192.168.200.250:53 0.0.0.0:* 1442/named
udp 0 0 127.0.0.1:53 0.0.0.0:* 1442/named
udp6 0 0 :::53 :::* 1442/named
```

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

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

1.11.1.- Archivos de configuración del servidor DNS.

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

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

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

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

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

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

Donde,

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

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

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

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

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

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

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

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

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

donde,

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

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

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

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

donde,

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

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

`filename` → El nombre del archivo de zona.

Ejemplos de ejecución:

1. Verificar archivo de configuración

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

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

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

1.11.2.- Arranque y parada del servidor DNS.

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

✓ **Comando `service`:**

1. Comprobar las opciones del comando:

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

Donde,

`start` → opción que permite arrancar el servicio.

`stop` → opción que permite apagar el servicio.

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

`restart` → opción que permite reiniciar el servicio.

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

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

2. Arrancar el servidor DNS:

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

3. Parar el servidor DNS:

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

4. Comprobar el estado activo/inactivo del servicio:

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

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

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

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

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

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

✓ **Comando `/etc/init.d/bind9`:**

1. Comprobar las opciones del comando:

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

2. Arrancar el servidor DNS:

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

3. Parar el servidor DNS:

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

4. Comprobar el estado activo/inactivo del servicio:

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

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

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

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

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

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

1.11.3.- Configuración como caché DNS.

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

Proverbio español

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

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

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

Para activar la caché debes realizar el siguiente procedimiento:

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

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

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

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

    //forwarders {
    0.0.0.0;
    };

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

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

```
nameserver 127.0.0.1
```

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

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

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

ANEXO II - Comprobar funcionamiento servidor DNS BIND

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

1. Borrado de la caché del cliente DNS:

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

2. Borrado de la caché del servidor DNS BIND:

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

1.11.4.- Configuración como DNS maestro.

"El maestro sabe lo que hace."

Proverbio español

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

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

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

En este ejemplo, el servidor sirve el dominio "ejemplo.com" como master, y la zona se guarda en el fichero `/var/lib/bind/master/db.ejemplo.com.hosts`.

Habrà una entrada de este tipo por cada zona servida.

Normalmente los ficheros de zona están situados en la ruta `/var/lib/bind`. Entonces, para una mayor comprensión y entendimiento, y para facilidad de uso en posteriores momentos, estaría bien que crearas los directorios master y slave dentro de esa ruta. Así, los ficheros con zonas maestras se pueden encontrar en `/var/lib/bind/master/db.*.hosts` y los ficheros con zonas esclavas se pueden encontrar en `/var/lib/bind/slave/db.*.hosts`.

2. Configuras el fichero `/var/lib/bind/master/db.ejemplo.com.hosts` para agregar los registros RR a la zona, por ejemplo:

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

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

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

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

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

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

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

;; Query time: 3 msec
```

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

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

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

1.11.5.- Configuración como DNS esclavo.

"Hace mal quien lo secundario hace principal."

Proverbio español

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

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

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

En este ejemplo, el servidor sirve el dominio "`ejemplo.com`" como slave, y la zona se guarda en el fichero `/var/lib/bind/slave/db.ejemplo.com.hosts`.

Habrà una entrada de este tipo por cada zona servida.

Normalmente los ficheros de zona están situados en la ruta `/var/lib/bind`. Entonces, para una mayor comprensión y entendimiento, y para facilidad de uso en posteriores momentos, estaría bien que crearas los directorios master y slave dentro de esa ruta. Así, los ficheros con zonas maestras se pueden encontrar en `/var/lib/bind/master/db.*.hosts` y los ficheros con zonas esclavas se pueden encontrar en `/var/lib/bind/slave/db.*.hosts`.

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

1. Para indicar qué servidores tienen permitido la transferencia de los ficheros de zona, mediante la directiva `allow-transfer`. Por ejemplo:

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

En este listado deberán estar incluidos todos los servidores slave que tengan configurado a éste como servidor master, y adicionalmente alguna IP que debiera tenerlo permitido por alguna razón.

2. Mediante la directiva `notify-yes` se consigue enviar automáticamente una notificación de cambio de zona del maestro, cuando ésta se produce, a los servidores DNS especificados en la zona mediante el registro de recurso NS.

Adicionalmente, se puede enviar una notificación de cambio de zona a servidores esclavos que no aparecen en la misma, mediante la directiva `also-notify`:

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

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

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

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

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

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

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

2.- Servicio de directorio.

Caso práctico

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

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

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

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

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

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

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

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

4. Dificilmente configurable:

1. ¿Cómo hacer en la guía telefónica para realizar una búsqueda solamente sobre un segundo apellido, de una zona urbana y con teléfonos que poseen dos números que tú determines?
2. ¿Cómo hacer en la programación televisiva para realizar una búsqueda sobre cadenas por satélite para Europa que emitan en franjas horarias determinadas deportes y, a poder ser, torneos o ligas profesionales?

Bien, parece ser que manejar tanta cantidad de información para ser ofrecida en ese tipo de búsquedas la hace no impresa e incluso inmanejable. Por contra, los directorios electrónicos pueden establecer la información que recibe una persona en función de sus necesidades.

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

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

Bjarne Stroustrup

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

1. Encontrar información:

Los directorios electrónicos a diferencia de los clásicos permiten acceder a la información contenida en los mismos de múltiples formas. Así, comparando con la guía telefónica tradicional, un directorio electrónico permite realizar búsquedas, no solamente por orden alfabético, sino también por: apellido: dirección, teléfono... ¿Cómo realizarías una búsqueda por teléfono en una guía telefónica tradicional?

Es más, podrías sumar campos de búsqueda, como por ejemplo: dirección y apellido.

2. Gestionar información:

En los directorios electrónicos pueden existir varios usuarios que en tiempo real estén realizando modificaciones, como agregar/editar/eliminar distintos usuarios con sus correspondientes campos. Además, esta información ya estaría visible para todas aquellas aplicaciones que accedan a la misma. Centralizar así los datos en un directorio evita tener que sincronizar varios directorios, con el consiguiente riesgo que esto provoca, pues: ¿qué pasaría si la sincronización no tuvo lugar y una aplicación accede a los datos? Pues sí, obtendría los datos no actualizados, o error en los mismos.

Un caso muy común es el de los servidores Web con autenticación: si solamente dispones de un servidor web la solución es sencilla, puesto que solamente se necesitaría actualizar una base de datos de usuarios, pero ¿y si dispones de más de un servidor web que debe acceder a la misma base de datos? Entonces, la cosa se complica, puesto que debes sincronizar a los distintos servidores. Es más, y si esa base de datos la quisiéramos aprovechar para ofrecer otro servicio distinto del de los servidores web? Pues, todo el trabajo no sería aprovechable, y por lo tanto sería mejor desde un principio adaptar este sistema a los servicios de directorios.

3. Control de seguridad:

Los servicios de directorios no simplemente permiten delimitar el acceso a los usuarios, sino que también proporcionan una solución al problema de gestión de certificados digitales. Así, permiten:

1. **Su creación:** Incorporar a los certificados los datos contenidos en el directorio.
2. **Su distribución:** Tener accesibles mediante un protocolo estándar los certificados.
3. **Su destrucción:** Revocar los certificados de forma sencilla simplemente borrando el certificado del directorio.
4. **Su ubicación:** Los usuarios pueden acceder a través del directorio a los certificados de los restantes usuarios, de forma muy sencilla y fácil de integrar con las aplicaciones.

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

2.2.- Directorio vs DNS.

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

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

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

Antes de intentar configurar una aplicación web con autenticación LDAP deberías probar la instalación, configuración y autenticación por medio de una base de datos SQL.

En el siguiente anexo encontrarás información detallada sobre el procedimiento de instalación, configuración y autenticación por medio de una base de datos SQL de la aplicación web OpenCart.

ANEXO III - Ejemplo despliegue aplicación web OpenCart

2.3.- Organización del directorio LDAP.

El servicio de directorio puede estar centralizado o distribuido:

- ✓ **Centralizado:** En este caso un único servidor ofrece todo el servicio de directorio respondiendo a todas las consultas de los clientes.
- ✓ **Distribuido:** Si el directorio está distribuido, varios servidores proporcionan el servicio de directorio. Cuando está distribuido, los datos pueden estar fraccionados y/o replicados:
 - ➔ Cuando está fraccionada, cada servidor de directorio almacena un subconjunto único y no solapado de la información, es decir, una entrada es almacenada en un solo servidor.
 - ➔ Cuando la información está replicada, una entrada puede estar almacenada en varios servidores.

Generalmente cuando el servicio de directorio es distribuido, parte de la información está fraccionada y parte está replicada.

En 1988, la CCITT (ahora ITU-T) creó el estándar X.500 (<http://x500standard.com/>) sobre servicios de directorio, el cual organiza las entradas en el directorio de manera jerárquica, capaz de almacenar gran cantidad de datos, con grandes capacidades de búsqueda y fácilmente escalable. X.500 especifica que la comunicación entre el cliente y el servidor de directorio debe emplear el protocolo DAP, pero DAP es un protocolo a nivel de aplicación, por lo que, tanto al cliente como el servidor debían implementar completamente la torre de protocolos OSI.

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

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

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

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

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

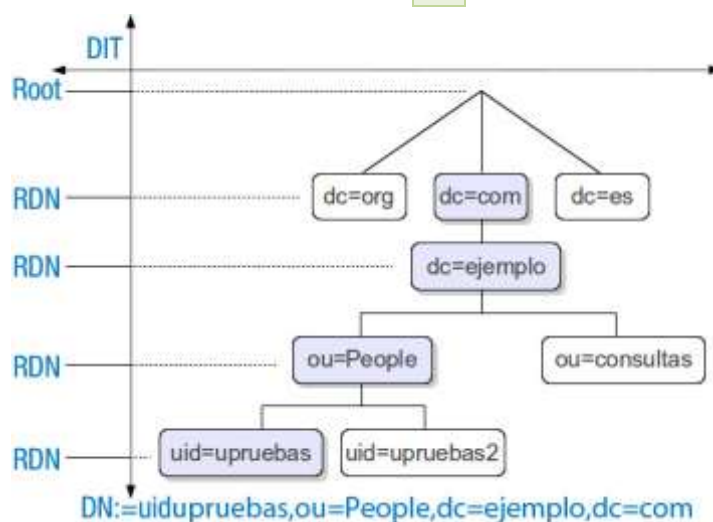
Por lo general, existen dos tipos de objetos:

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

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

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

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



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

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

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

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

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

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



Puedes utilizar el servicio de directorio como repositorio en el cual almacenar la información que varios servidores deben compartir, por ejemplo: la configuración, información sobre el control de acceso, etc.

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

Otra utilidad que puede resultar interesante es la de emplear el servicio de directorio para indexar la documentación almacenada en el servidor Web, con la precisión que otras herramientas no pueden generar.

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

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

2.5.- El formato de intercambio de datos LDIF.

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

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

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

- ✓ El **DN** que debe figurar en la primera línea de la entrada y que se compone de la cadena **dn:** seguida del nombre distinguido (**DN**) de la entrada.
- ✓ La segunda parte son los atributos de la entrada. Cada atributo se compone de un nombre de atributo, seguido del carácter dos puntos ':' y el valor del atributo. Si hay atributos multivaluados deben ponerse seguidos.

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

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

<nombre_atributo>: <valor>.

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

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

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

2.6.- Instalación de OpenLDAP.

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



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

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

1. Actualiza los repositorios del sistema operativo.

```
root@debian-servidor-fp:~# apt-get update
```

NOTA: es necesario para el buen funcionamiento del comando que tengas configurado correctamente la conexión a Internet.

2. Actualiza el sistema operativo.

```
root@debian-servidor-fp:~# apt-get upgrade
```

3. Instala los paquetes necesarios para el funcionamiento de OpenLDAP. La instalación te pedirá una contraseña, como puedes ver a

```
root@debian-servidor-fp:~# apt-get install slapd ldap-utils
Contraseña del administrador: admin
Verificación de la contraseña: admin
```

4. Verifica que el servidor OpenLDAP está activo, por defecto, en el puerto **TCP 389**.

```
root@debian-servidor-fp:~# netstat -natp | grep 389
tcp 0 0 0.0.0.0:389 0.0.0.0:* LISTEN 1955/slapd
tcp6 0 0 :::389 :::* LISTEN 1955/slapd
```

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

ANEXO IV - INSTALACIÓN Y CONFIGURACIÓN DEL SERVIDOR OPENLDAP

2.6.1.- Configuración de OpenLDAP.



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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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



2.6.2.- Arranque y parada del servidor LDAP.

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

✓ Comando `service`:

1. Comprobar las opciones del comando:

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

Donde:

start → opción que permite arrancar el servicio.

stop → opción que permite apagar el servicio.

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

restart → opción que permite reiniciar el servicio.

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

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

2. Arrancar el servidor OpenLDAP:

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

3. Parar el servidor OpenLDAP:

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

4. Comprobar el estado activo/inactivo del servicio:

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

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

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

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

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

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

✓ Comando `/etc/init.d/slapd`:

1. Comprobar las opciones del comando:

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

2. Arrancar el servidor OpenLDAP:

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

3. Parar el servidor OpenLDAP:

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

4. Comprobar el estado activo/inactivo del servicio:

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

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

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

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

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

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

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

2.6.3.- Administrando un servidor LDAP:

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



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

1. Añadir entradas: comando `ldapadd`.

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

`estructura_basica.ldif`.

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

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

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

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

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

`usuario.ldif`.

```
# Usuario
```

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

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

2. Modificar entradas: comando `ldapmodify`.

- Modificar la contraseña del usuario anterior '`pruebas`' mediante la ejecución del archivo `cambiar_usuario.ldif`.

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

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

3. Buscar entradas: comando `ldapsearch`.

- Buscar todos los usuarios cuyo nombre contenga los caracteres '`pru`':

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

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

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

4. Eliminar entradas: comando `ldapdelete`.

- Eliminar el usuario `upruebas`:

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

Los comandos anteriores poseen la opción `-h` con la cual se puede indicar el host (nombre de dominio o IP) que identifica al servidor LDAP. Por ejemplo: `ldapsearch -h 192.168.200.250 -x -b dc=proyecto-empresa,dc=local "(objectclass=*)"` conectaría con el servidor LDAP en la IP `192.168.200.250` para buscar el DIT del dominio `proyecto-empresa.local`.

Existe un paquete de nombre `ldapscripts` que contiene una serie de scripts para administrar de forma sencilla los usuarios y grupos almacenados en el servidor LDAP. Puedes encontrar plantillas de ejemplo, formato LDIF, situadas en `/usr/share/doc/ldapscripts/examples/` cuando se instala el paquete `ldapscripts`.

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

ANEXO V - Exploradores de directorios LDAP.

ANEXO VI - Administración de usuarios y grupos con LDAP

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

Como ya hemos comentado anteriormente, una de las utilidades más importantes de un servidor LDAP es la de servidor de autenticación. Autenticarse 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 autentique 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, autenticando los usuarios que inicien sesión validándose contra el servidor LDAP.

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

Para ello realiza el siguiente procedimiento:

1. Instala y configura los paquetes

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

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

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

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

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

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

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

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

5. Por último, prueba que la configuración del cliente es correcta:

1. Mediante el comando `getent passwd`, que proporciona todos los usuarios del sistema operativo, en este caso los de `Unix authentication` y `LDAP Authentication`.

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

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

2.6.5.- Probar la autenticación con pamtest.



Ahora que la autenticación 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 autenticación y el segundo es el nombre del usuario que se va a autenticar sobre dicho servicio. Veamos unos ejemplos:

1. Intentar autenticar 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 autenticar 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 autenticar 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 autenticar 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 autenticar 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 autenticar 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 autenticar a los usuarios a través de LDAP.

ANEXO I - Servidores raíz DNS

```

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

```

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

ANEXO II - Comprobar funcionamiento servidor DNS BIND

Procedimiento para comprobar el funcionamiento del servidor DNS BIND como servidor caché:

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

1. Ejecutas el comando:

```
dig www.debian.org
```

El cual te muestra una salida similar a la siguiente:

```
; <<>> DiG 9.7.3 <<>> www.debian.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16236
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL: 0
;; QUESTION SECTION:
;www.debian.org. IN A
;; ANSWER SECTION:
www.debian.org. 300 IN A 86.59.118.148
www.debian.org. 300 IN A 82.195.75.97
;; AUTHORITY SECTION:
www.debian.org. 28800 IN NS geo1.debian.org.
www.debian.org. 28800 IN NS geo3.debian.org.
www.debian.org. 28800 IN NS geo2.debian.org.
;; Query time: 401 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Sep 12 08:20:34 2011
;; MSG SIZE rcvd: 121
```

donde, **401 msec** significa el tiempo de resolución consumido de la petición DNS en milisegundos.

2. Ejecutas de nuevo el comando anterior: `dig www.debian.org` obteniendo una salida similar a la siguiente:

```
; <<>> DiG 9.7.3 <<>> www.debian.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10876
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL: 0
;; QUESTION SECTION:
;www.debian.org. IN A
;; ANSWER SECTION:
www.debian.org. 295 IN A 82.195.75.97
www.debian.org. 295 IN A 86.59.118.148
;; AUTHORITY SECTION:
www.debian.org. 28795 IN NS geo3.debian.org.
www.debian.org. 28795 IN NS geo1.debian.org.
www.debian.org. 28795 IN NS geo2.debian.org.
;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Sep 12 08:20:39 2011
;; MSG SIZE rcvd: 121
```

donde, **4 msec** significa el tiempo de resolución consumido de la petición DNS en milisegundos.

3. Ahora, deberías obtener una salida con un número inferior indicando un menor tiempo de resolución en la petición DNS. En este procedimiento puedes observar que el tiempo disminuye de **401 msec** en el paso 1 a **4 msec** en el paso 2, lo que indica que el tiempo de resolución consumido por la petición fue menor, puesto que la misma fue resuelta por la caché DNS, es decir, la primera consulta realizada en el paso 1 fue escalada a otro/s servidor/es DNS, empezando la búsqueda a través de los servidores raíz que se encuentran en el archivo `/etc/bind/db.root`, mientras que la segunda consulta, realizada en el paso 2 fue realizada en el propio servidor DNS y no fue escalada a otro/s servidor/es DNS.

ANEXO III - Ejemplo despliegue aplicación web OpenCart

"El movimiento se demuestra andando."

Diógenes de Sínope

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

1. Descarga y descomprime la aplicación:

- ✓ En la página de descarga de OpenCart(<http://www.opencart.com/index.php?route=download/download>) puedes ver los requisitos para la instalación de Opencart: Web Server (preferiblemente Apache) , PHP (al menos la 5.2), MySQL , Curl , Fsock

- ✓ Descarga el último paquete estable de Opencart de la página web de descarga en `/tmp/pruebas`

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

- ✓ Descomprime el paquete

```
cd /tmp/pruebas
apt-get install unzip
unzip opencart_v1.4.9.5.zip
```

2. Lee el fichero de instalación `install.txt`.

3. Crea el virtualhost para OpenCart:

- ✓ Copia la carpeta `upload` en el servidor web. Para ello genera en `/etc/apache2/sites-available/` un virtualhost de nombre `tienda-virtual` como el siguiente:

```
<VirtualHost 192.168.200.250:80>
    DocumentRoot /var/www/tienda-virtual
    ServerName www.tienda-virtual.empresa-proyecto.com
    ErrorLog /var/log/apache2/error tienda-virtual.log
    CustomLog /var/log/apache2/access tienda-virtual.log "%h %l %u %t \"%r\" %>s %b\n\"%{Referer}i\" %I %O"
</VirtualHost>
```

- ✓ Ahora mueve la carpeta `upload` con el nombre `tienda-virtual` en `/var/www/tienda-virtual`
- ✓ Activa el sitio nuevo `tienda-virtual: a2ensite tienda-virtual`
- ✓ Recarga la configuración de Apache: `/etc/init.d/apache2 reload`
- ✓ Verifica que los siguientes ficheros y carpetas tengan permisos de escritura en `/var/www/tienda-virtual/`:

```
chmod 0755 6 0777 para: image/, image/cache/, image/data/, system/cache/, system/logs/,
download/, config.php, admin/config.php
```

4. Crea la base de datos para OpenCart y el usuario con permisos en la misma:

Asegúrate que posees una base de datos mysql para OpenCart y un usuario distinto de root con permisos en la misma:

- ✓ Primero, debes crear una nueva base de datos para tu sitio OpenCart:

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

donde:

`root` es el usuario administrador de MySQL y por lo tanto tiene los privilegios para crear una base de datos.

`db_opencart` es el nombre de la base de datos de opencart que acabas de crear.

MySQL te pide la contraseña del usuario root y luego crea los archivos iniciales de la base de datos.

- ✓ Segundo, creas el usuario con privilegios en la base de datos (*de nuevo se requiere la contraseña de root*).

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

donde:

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

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

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

- ✓ Tercero, para activar los nuevos cambios ejecuta:

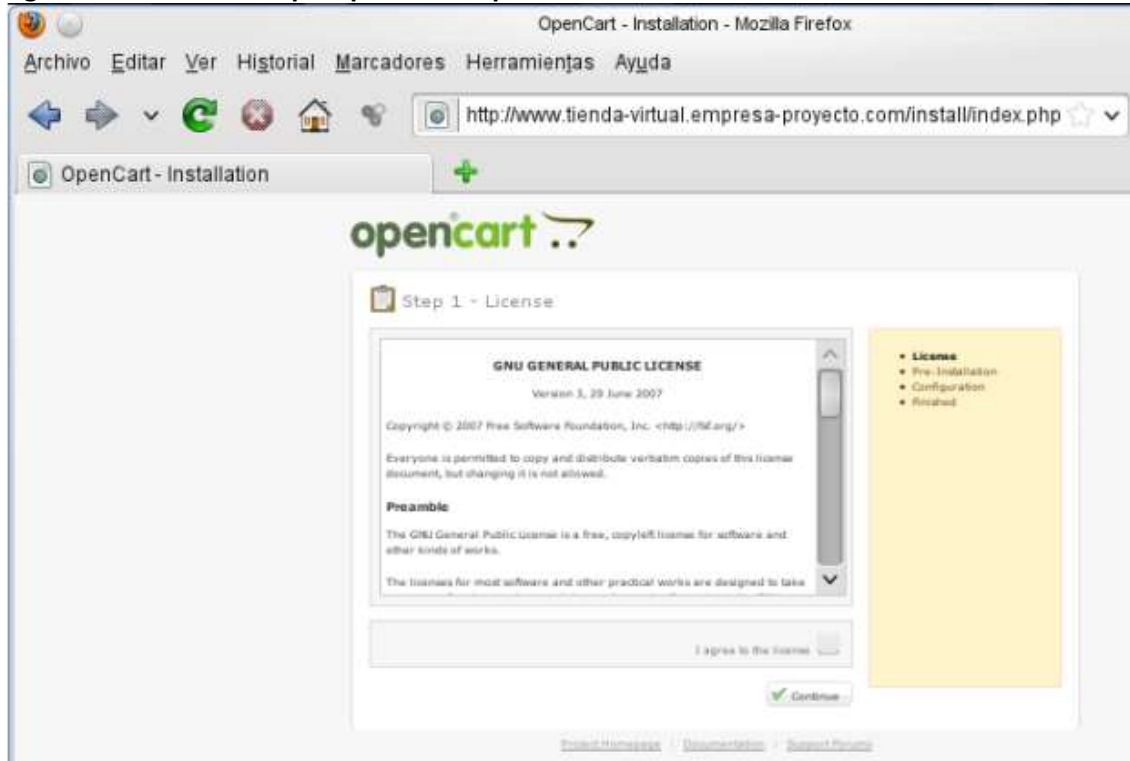
```
/usr/bin/mysql -h127.0.0.1 -uroot -p -e "flush privileges;"
```

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

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

<http://www.tienda-virtual.empresa-proyecto.com/>

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



7. **Una vez acabada la instalación borra la carpeta `install`.**

8. **Puedes ya visitar tu tienda online en:** <http://www.tienda-virtual.empresa-proyecto.com/> **y tu panel de administración en:** <http://www.tienda-virtual.empresa-proyecto.com/admin/>

ANEXO IV - Instalación y configuración de OpenLDAP

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

Instalación de OpenLDAP

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

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

Configuración inicial de OpenLDAP

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

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

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

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



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

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



Con esto habremos concluido la configuración inicial del servidor LDAP.

Arranque y parada manual del servidor LDAP

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

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

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

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

Administración de OpenLDAP

Introducción

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

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

Paso 1: Cargar plantillas

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

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

Paso 2: Archivo de configuración del esquema básico

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

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

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

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

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

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

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

Paso 3: Creación de unidades organizativas para almacenar cuentas unix

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

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

dn: cn=admin,dc=ieslapaloma,dc=com
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword:: e2NyeXB0fXdSVDNLMEpKS1QydmM=

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

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

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

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

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

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

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

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

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

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

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

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

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

ANEXO V - Explorador de directorios LDAP

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

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

Instalar phpldapadmin

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



JXplorer - Explorador LDAP en java

Por su calidad superior, utilizaremos **JXplorer** para administrar el directorio LDAP.

Previo a instalar **jxplorer**, es necesario instalar la máquina virtual java de Sun, para lo cual utilizaremos `apt-get`, pero antes debemos activar los repositorios `-partner-` de Ubuntu

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

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

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

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

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

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

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

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

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

Veremos la pantalla principal de JXplorer:



Conexión con el servidor LDAP

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

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

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

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

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

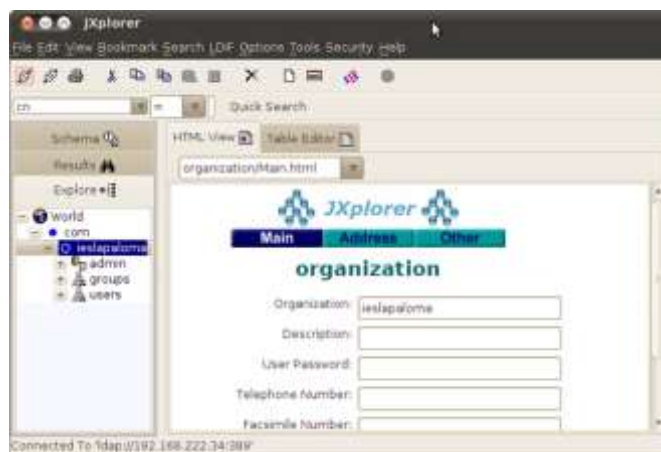
El nombre de usuario administrador por defecto suele ser `admin` y a menudo hay que proporcionar nombre y base del directorio: `cn=admin, dc=ieslapaloma, dc=com`

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



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

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



Creación de usuarios y grupos con jxplorer

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

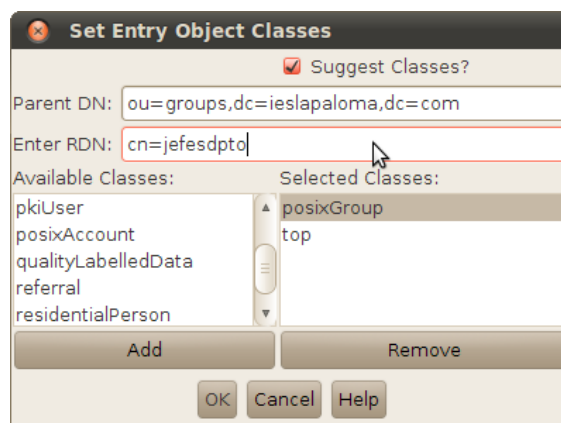
Crearemos el siguiente grupo:

- ✓ jefesdpto (gid=1003)

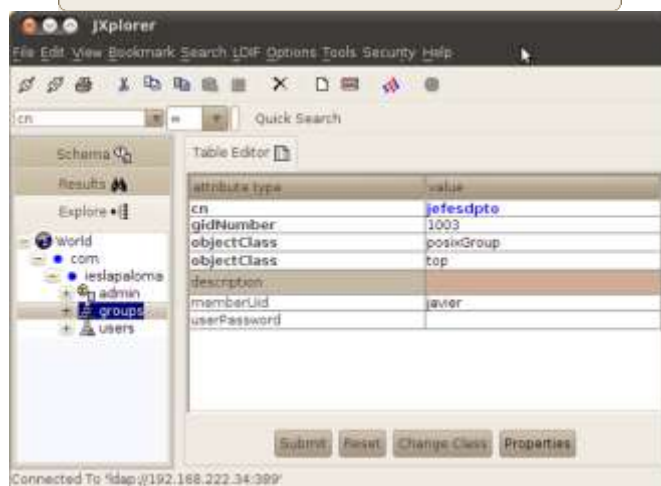
Además, crearemos un usuario nuevo:

- ✓ carlos (uid=1006, jefesdpto)

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



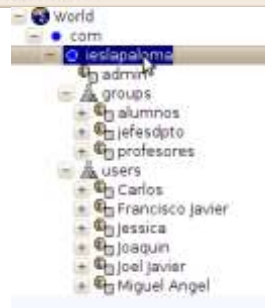
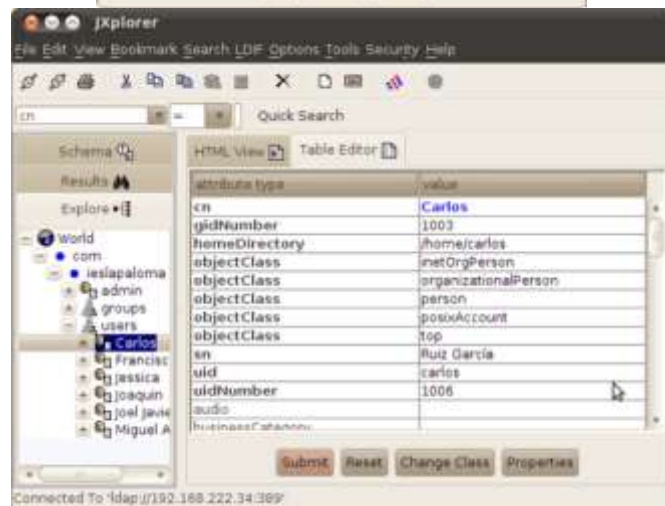
Al pulsar **OK** nos aparecerá la siguiente figura, en la cual observamos los atributos clásicos de un grupo posix. Debemos rellenar al menos el campo 'gidNumber'. También podemos introducir miembros al grupo. En el parámetro 'memberUid' añadimos 'javier'. Luego, haciendo clic con el derecho en 'javier'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



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



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



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

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

Trabajar con herramientas gráficas como jxplorer o phpldapadmin resulta interesante cuando hay que realizar consultas o pequeñas modificaciones...

...pero cuando se trata de crear usuarios de forma masiva, lo mejor es utilizar archivos ldif y el comando ldapadd para cargarlos al servidor

ANEXO VI - Administración de usuarios y grupos con LDAP

Administración mediante scripts

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

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

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

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

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

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

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

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

NOTAS:

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

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

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

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

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

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

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

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

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

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

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

Administración de usuarios y grupos de LDAP con webmin

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

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

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

Creación masiva de usuarios

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

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

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

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

Podemos ver a continuación un ejemplo donde se carga el fichero `usuarios.txt` con este contenido, y el resultado de su ejecución:

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

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

Nombre de usuario	ID de usuario	Grupo	Nombre Real	Directorio local	Shell
felipe	10000	prof	Felipe Carballo	/home/felipe	/bin/bash
alberto	10000	prof	Alberto Miguez	/home/alberto	/bin/bash
xan	10000	prof	Xan Pereira	/home/xan	/bin/bash

Lista de usuarios de LDAP después de cargado el fichero

El módulo de servidor LDAP

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

Configuración

Para el módulo LDAP Server

Opciones configurables para LDAP Server

LDAP server options

LDAP server hostname ☒ This system

LDAP server port ☒ Detect automatically

Login for LDAP server ☐ Detect automatically ☒ cn=admin,dc=iescalquera,dc=

Password for LDAP server ☐ Detect automatically ☒ admin

Use encryption with LDAP server? ☒ Detect automatically ☐ Yes ☐ Yes TLS ☐ No

Full path to OpenLDAP server program

OpenLDAP server configuration file or directory

OpenLDAP schema directory

User OpenLDAP server runs as

OpenLDAP server boot script name ☐ Same as module name ☒

OpenLDAP database directory ☒ Not known

User interface settings

Maximum number of sub-objects to display ☐ Unlimited ☒

LDAP server commands

Command to start LDAP server ☐ Just run slapd ☒

Command to stop LDAP server ☐ Just kill process ☒

Command to apply configuration ☐ Just stop and re-start ☒

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

Vista del contenido de la rama base de LDAP

[illegible]

Vista de las propiedades del usuario Alberto

LDAP Account Manager

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

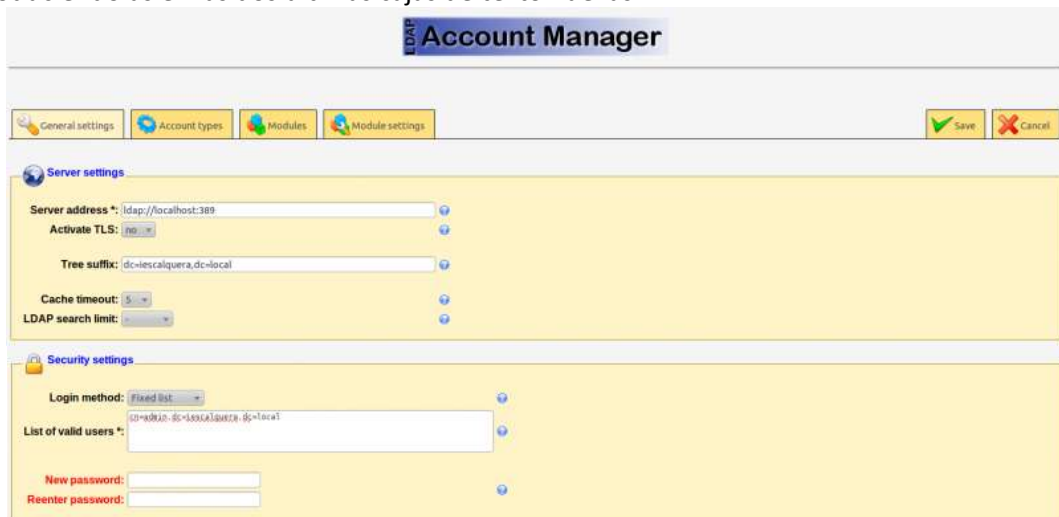
```
sudo apt-get install ldap-account-manager
```

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



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

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



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



Picamos en el botón **Save** para guardar los cambios. Todos estos parámetros introducidos se almacenan en el fichero de configuración de lam (`/usr/share/ldap-account-manager/config/lam.conf`).

Ahora ya podemos entrar una herramienta introduciendo la contraseña de administrador de LDAP (`admin`):

Inicio de sesión



Nos pregunta si queremos crear las ramas para almacenar las máquinas y los dominios en el directorio, ya que detecta que no existe aún.



Vista del árbol de LDAP



Vista de los usuarios



Vista de los grupos



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

TEMA 6

Contenido

1.- Documentación de aplicaciones web.....	1
2.- PhpDocumentor.....	3
2.1.- Funcionamiento de phpDocumentor.....	4
2.2.- Instalación de phpDocumentor.....	5
2.3.- Configuración de phpDocumentor.....	6
3.- JavaDoc.....	8
3.1.- Instalación de Javadoc.....	9
3.2.- Documentando con JavaDoc.....	10
3.3.- Creación y uso de plantillas de código.....	11
4.- Sistemas de control de versiones.....	12
4.1.- Conceptos básicos de sistemas de control de versiones.....	12
4.2.- Procedimiento de uso habitual de un sistema de control de versiones.....	13
4.3.- Sistemas de control de versiones centralizados y distribuidos.....	14
4.4.- Git como sistema de control de versiones.....	15
4.5.- Funcionamiento de Git.....	18
4.6.- Instalación de Git.....	19
4.7.- Configuración de Git (I).....	20
4.7.1.- Configuración de Git (II).....	21
4.8.- Trabajando con Git (I).....	22
4.8.1.- Trabajando con Git (II).....	23
4.9.- Seguridad documentación en Git.....	24

Documentación y control de versiones.

Caso práctico

En la empresa **BK programación** están centrándose especialmente en el desarrollo de aplicaciones web, debido a que es el entorno más solicitado por los clientes; a medida que el número de proyectos que se incorporan aumenta surge la necesidad de automatizar y gestionar, mediante algún tipo de herramientas software, determinados aspectos relacionados con el desarrollo de aplicaciones web, como es el caso de la documentación de las aplicaciones y el establecimiento de un sistema de control de versiones.

La documentación de un proyecto de software es tan importante como su código. Una buena documentación nos facilita, en gran medida, el mantenimiento futuro de la aplicación. Si además estamos trabajando en equipo es muy útil saber lo que hacen las partes que desarrollan otras personas, sobre todo si tenemos que utilizarlas en nuestra parte.

Juan ha empezado a utilizar diversas herramientas que permiten generar documentación de forma automática a partir del código fuente. Javadoc es la herramienta estándar en Java. Para PHP una de las herramientas más utilizadas es phpDocumentor.

Un sistema de control de versiones se encarga de controlar los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Los sistemas de control de versiones facilitan la administración de las distintas versiones de cada producto desarrollado, así como las posibles especializaciones realizadas (por ejemplo, para algún cliente específico).

El control de versiones se realiza, principalmente, en la industria informática para controlar las distintas versiones del código fuente. Sin embargo, los mismos conceptos son aplicables a otros ámbitos como documentos, imágenes, sitios web, etcétera.

María se ha encargado de documentarse acerca de los programas de software libre existentes para sistemas de control de versiones con la finalidad de implantar alguno de ellos en la empresa, entre los que ha destacado Git y Subversion.

1.- Documentación de aplicaciones web.

Caso práctico

Todos los proyectos software deben disponer de una documentación consistente, de forma que en cualquier etapa de su ciclo de vida, empezando en su fase de análisis y diseño, pasando por su fase de codificación o implementación (etapa donde, efectivamente, se programa el sistema. La implementación es la programación de un determinado algoritmo en un lenguaje de programación específico. Por ejemplo, un algoritmo en pseudocódigo se implementa en forma de código de un lenguaje de programación) en un lenguaje de programación determinado e, incluso, en la fase de explotación, debe haber una documentación robusta con la finalidad de suministrar información relevante acerca del código, funcionalidades de la aplicación o, incluso, limitaciones de la misma, para que la aplicación pueda ser explotada en su total amplitud y para que su mantenimiento resulte cómodo.

En **BK programación**, Juan está estudiando diversas herramientas que ayuden a automatizar el proceso de documentación de las aplicaciones, entre las que destaca **phpDocumentor** y **JavaDoc**.

En primer lugar sería necesario responder a la siguiente cuestión, ¿qué conviene documentar en una aplicación? En principio tres aspectos fundamentales de la aplicación:

1. **La interfaz:** Qué hace (*no como lo hace*) una función o un método (*subrutina asociada exclusivamente a una clase*) de una clase (*construcción que se emplea como modelo para crear objetos de un tipo determinado. La clase describe el comportamiento y estado que todos los objetos comparten*), qué parámetros hay que pasar y qué devuelve. Esta información es tremendamente útil para las personas que utilizan funciones o clases diseñadas por otros.
2. **La implementación:** Indicar cómo está implementada cada función, cómo se lleva a cabo cada paso, por qué se utiliza determinada variable, qué algoritmo se utiliza, qué hacen los métodos privados de una clase. Toda esta información resulta interesante a quienes tengan que depurar o actualizar bloques de código de la aplicación.
3. **La toma de decisiones:** Por qué se ha implementado de determinada forma y no de otra la aplicación, por ejemplo, para analizar el rendimiento de la aplicación y optimización de recursos.

Esto resulta interesante a nivel de implementación para los desarrolladores y a nivel funcional para los responsables del desarrollo.

Normalmente la información sobre la implementación no necesita salir del código pero, por el contrario, la información de la interfaz conviene pasarla a un documento independiente del código fuente (manual de uso). La persona que necesite utilizar una determinada librería de clases o funciones tendrá toda la información necesaria: qué hace cada elemento y cómo se utiliza. No necesita acceder al código fuente.

El problema con este tipo de documentación es que cada vez que se modifica algo en el código (actualizaciones, corrección de errores, etc...) hay que reflejarlo también en el manual de uso, lo que implica doble trabajo. Lo ideal, por tanto, sería poder automatizar de alguna forma este proceso.

Existen algunas herramientas que permiten generar documentación de forma automática a partir del código fuente. **Javadoc** es la herramienta estándar en Java. Para PHP una de las herramientas más utilizadas es **phpDocumentor**.

Los entornos de programación modernos, por ejemplo, son capaces de obtener la información de los comentarios, de forma que la muestran en el "autocompletado" de código, que se convierte en una herramienta estupenda, y aun imprescindible en lenguajes como PHP, que no necesita que se declare el tipo del argumento de una función, por poner un caso. Entornos como NetBeans o Eclipse, aprovechan los comentarios de nuestro código fuente para mostrar información muy útil, sobre todo para terceras personas.

Hay que tener en cuenta que todas estas herramientas que venimos viendo, NetBeans, Eclipse, phpDocumentor, esperan el mismo tipo de comentarios, basado en el estándar establecido por Javadoc, de modo que haremos el trabajo una sola vez y podremos aprovecharnos del mismo en varios entornos y con varias herramientas. Más aún, toda persona que se acerque a nuestro proyecto podrá aprovechar la documentación, incluso más que nosotros.

"La programación es una carrera entre ingenieros de software luchando para construir programas cada vez más grandes, mejores y a prueba de idiotas, y el universo intentando producir cada vez más grandes y mejores idiotas. Por ahora, gana el universo. "

Rich Cook

2.- PhpDocumentor.

Caso práctico

Una de las herramientas que la empresa **BK programación** ha decidido someter a estudio, con la finalidad de ayudar a documentar el software desarrollado en PHP, y así decidir su implantación, es **phpDocumentor**. Para ello Juan ha empezado a documentarse sobre dicha herramienta.

Existen algunas herramientas que permiten generar documentación de forma automática a partir del código fuente, **Javadoc** es la herramienta estándar para Java, para PHP una de las herramientas más utilizadas es **phpDocumentor**.

Como ya comentamos antes, la documentación de un proyecto de software es tan importante como su código. Una buena documentación nos facilita, en gran medida, el mantenimiento futuro de la aplicación. Si además estamos trabajando en equipo es muy útil saber lo que hacen las partes que desarrollan otras personas, sobre todo si tenemos que utilizarlas en nuestra parte.

Para ayudarnos existe la aplicación **phpDocumentor**, que nos permite generar automáticamente una buena documentación de nuestro código, de una forma parecida a cómo lo hace **JavaDoc**. Mediante comentarios y unas etiquetas especiales podemos definir de forma sencilla qué hace cada clase, cada método y cada función de nuestro código. Para saber más sobre esta aplicación se puede acceder a su página web, desde donde se puede descargar la aplicación (es *software* libre) y acceder a la documentación de ésta, de todas maneras intentaremos ampliar aquí los conocimientos sobre esta herramienta.

<http://www.phpdoc.org/>

PhpDocumentor permite generar la documentación de varias formas y en varios formatos.

- ✓ Desde línea de comandos (php CLI - Command Line Interpreter).
- ✓ Desde interfaz web (incluida en la distribución).
- ✓ Desde código. Como phpDocumentor está desarrollado en PHP, podemos incluir su funcionalidad dentro de scripts propios.

En todo caso, es necesario especificar los siguientes parámetros:

1. El directorio en el que se encuentra nuestro código. PhpDocumentor se encargará luego de recorrer los subdirectorios de forma automática.
2. Opcionalmente los paquetes (`@package`) que deseamos documentar, lista de ficheros incluidos y/o excluidos y otras opciones interesantes para personalizar la documentación.
3. El directorio en el que se generará la documentación.
4. Si la documentación va a ser pública (sólo interfaz) o interna (en este caso aparecerán los bloques `private` y los comentarios `@internal`).
5. El formato de salida de la documentación.

Formatos de salida

1. HTML a través de un buen número de plantillas predefinidas (podemos definir nuestra propia plantilla de salida).
2. PDF.
3. XML (DocBook). Muy interesante puesto que a partir de este dialecto podemos transformar (XSLT) a cualquier otro utilizando nuestras propias reglas y hojas de estilo (*CSS viene del inglés Cascading Style Sheets, del que toma sus siglas. CSS es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML*).

Una alternativa a phpDocumentor es Doxygen que puede también documentar código **PHP**, la principal diferencia es que **Doxygen** es un programa, mientras phpDocumentor es una colección de código en PHP. Es decir, genera la documentación con el mismo PHP usado para ejecutar el propio

código PHP, es por ello que se necesita tener también PHP instalado, sin embargo no se necesita instalar un servidor web.

Una buena documentación facilita el mantenimiento de la aplicación. Existen herramientas que generan documentación de forma automática a partir del código fuente de las aplicaciones; entre las más utilizadas están **Javadoc**, que es la herramienta estándar en Java, y para PHP una de las herramientas más utilizadas es **phpDocumentor**.

2.1.- Funcionamiento de phpDocumentor.

En **phpDocumentor** la documentación se distribuye en bloques "*DocBlock*". Estos bloques siempre se colocan justo antes del elemento al que documentan y su formato es:

```
/**
 * Descripción breve (una línea)
 *
 * Descripción extensa. Todas las líneas que
 * sean necesarias
 * Todas las líneas comienzan con *
 <- Esta línea es ignorada
 *
 * Este DocBlock documenta la función suma()
 */
function suma()
{
    ...
}
```

Los elementos que pueden ser documentados son:

```
define
function
class
class vars
include/require/include once/require once
global variables
```

También se puede incluir documentación global a nivel de fichero y clase mediante la marca **@package**.

Dentro de cada **DocBlock** se pueden incluir marcas que serán interpretadas por phpDocumentor con un significado especial, dichas marcas pueden ser las siguientes:

- ✓ **@access**: Si **@access** es 'private' no se genera documentación para el elemento (a menos que se indique explícitamente). Muy interesante si sólo se desea generar documentación sobre la interfaz (métodos públicos) pero no sobre la implementación (métodos privados).
- ✓ **@author**: Autor del código.
- ✓ **@copyright**: Información sobre derechos.
- ✓ **@deprecated**: Para indicar que el elemento no debería utilizarse, ya que en futuras versiones podría no estar disponible.
- ✓ **@example**: Permite especificar la ruta hasta un fichero con código PHP. phpDocumentor se encarga de mostrar el código resaltado (syntax-highlighted).
- ✓ **@ignore**: Evita que phpDocumentor documente un determinado elemento.
- ✓ **@internal**: Para incluir información que no debería aparecer en la documentación pública, pero sí puede estar disponible como documentación interna para desarrolladores.
- ✓ **@link**: Para incluir un enlace (http://...) a un determinado recurso.
- ✓ **@see**: Se utiliza para crear enlaces internos (enlaces a la documentación de un elemento).
- ✓ **@since**: Permite indicar que el elemento está disponible desde una determinada versión del paquete o distribución.
- ✓ **@version**: Versión actual del elemento.

Existen otras marcas que solamente se pueden utilizar en bloques de determinados elementos:

- ✓ `@global`: Para especificar el uso de variables globales dentro de una función.
- ✓ `@param`: Para documentar parámetros que recibe una función.
- ✓ `@return`: Valor devuelto por una función.

2.2.- Instalación de phpDocumentor.

Para proceder a la instalación de **phpDocumentor** vamos a partir de una máquina en la que tenemos instalado la distribución **Debian 6.0.1Squeeze**.

Como requisito previo probaremos si **php** y **apache** están funcionando correctamente; lo podemos establecer con las siguientes pruebas:

1. Para probar si Apache sirve peticiones, abrimos un navegador e introducimos la siguiente URL `http://localhost` y debería aparecer una página con el mensaje "It works!"
2. Probar que funciona PHP, lo podemos hacer del siguiente modo, ejecutamos desde línea de comandos: `echo "" | php` y deberíamos ver como resultado "10"
3. Probar que Apache ejecuta código PHP, para ello en la carpeta donde Apache busca las páginas web, es decir en donde se encuentra la página `index.html` (la encargada de mostrar el mensaje "It works"), en nuestro caso la carpeta `/var/www`, creamos un archivo al que vamos a llamar "phpinfo.php" con el siguiente contenido :

```
<?php
phpinfo();
?>
```



Posteriormente tecleamos en el navegador la siguiente URL `http://localhost/phpinfo.php` y deberíamos encontrar información similar a la de la imagen, en donde vemos parte de las características de Apache y PHP de nuestro equipo.

Una vez que probamos que las pruebas anteriores han confirmado el correcto funcionamiento de Apache y PHP, comenzamos la instalación de **phpDocumentor**. Lo primero será instalar el paquete **php-pear**, que es un entorno de desarrollo y sistema de distribución para componentes de código PHP. Para instalarlo podemos hacerlo mediante el gestor de paquetes **apt**:

```
#apt-get install php-pear
```

Por defecto Apache busca en el directorio `/var/www` para el contenido web, con lo cual antes de la instalación de phpDocumentor deberíamos decirle a **pear** que es aquí donde queremos que **phpDocumentor** debe trabajar, se puede configurar este u otro directorio, siempre y cuando sea accesible por el servidor web:

```
#pear config-set data_dir /var/www
```

ahora podemos instalar phpDocumentor y también sus dependencias:

```
#pear install --alldeps PhpDocumentor
```

o también directamente podemos descargar el paquete mediante:

```
#wget sourceforge.net/projects/phpdocu/files/PhpDoc/phpDocumentor-1.4.3/PhpDocumentor-1.4.3.tgz
```

y luego descomprimirlo.

Una vez terminado el proceso de instalación necesitamos crear un directorio de salida para **phpDocumentor**, y cambiar el propietario de dicho directorio a **www-data**, de forma que pueda trabajar en esta carpeta sin ninguna limitación; por ejemplo podemos realizar lo siguiente:

```
#mkdir /var/www/docs
#chown www-data /var/www/docs/
```

Si desde un navegador tecleamos `http://localhost/PhpDocumentor/` deberíamos comprobar que la instalación ha sido correcta y tenemos **phpDocumentor** listo para funcionar.

Indica si las siguientes afirmaciones son verdaderas o falsas:

- ☒ **phpDocumentor es una colección de código en PHP.**
- ☒ **Para que funcione phpDocumentor es necesario tener instalado PHP.**
- ☐ phpDocumentor únicamente va a funcionar en los servidores web.
- ☐ Apache no es necesario para trabajar con phpDocumentor.

2.3.- Configuración de phpDocumentor.

Una vez hemos instalado **phpDocumentor** se puede trabajar con él de dos modos para generar automáticamente la documentación de nuestros proyectos PHP; se puede trabajar desde línea de comandos, mediante:

```
#phpdoc -o [formato_de_la_documentacion_generada] -d [carpeta_donde_estan_los_proyectos_php] -t [carpeta_donde_se_almacenan_los_archivos_de_documentacion]
```

Por ejemplo:

```
#phpdoc -o HTML:frames:phpedit -d /var/www/ -t /var/www/docs/
```

En este caso estamos indicando que se genere la documentación en formato HTML (también es posible en formato PDF, CHM) de los proyectos de la carpeta `/var/www` y se almacene dicha documentación en la carpeta `/var/www/docs/`; aunque existen un gran número de parámetros para adaptar el formato de la documentación que **phpdoc** genere al formato que más nos interese, para ello podemos obtener información ejecutando:

```
#phpdoc -h
```

Por otra parte también es posible trabajar desde el entorno web que **phpDocumentor** ofrece; para lo que es recomendable establecer la configuración necesaria de donde phpDocumentor debe recoger los proyectos y a donde enviar la documentación generada; para ello podemos establecer la configuración del siguiente modo:



En la ruta donde hemos instalado phpDocumentor accedemos a la ruta: `PhpDocumentor/user` y duplicamos el archivo `default.ini` poniéndole un nombre distinto; por ejemplo: `"mi_proyecto.ini"`.

- ✓ Editamos el archivo que acabamos de crear y modificamos lo siguiente:
 - ➔ Para establecer la ruta donde guardar la documentación generada cambiamos la línea con el contenido: `target = /you-MUST/change-me/to-fit/your-environment` estableciendo la ruta que nos interese, por ejemplo: `target = /var/www/Documentacion_de_mi_proyecto`.
 - ➔ Para establecer la ruta donde se encuentran los archivos del proyecto cambiamos la línea con el contenido: `directory = /you-MUST/also-change-me/to-fit/your-environment` estableciendo la ruta que nos interese, por ejemplo: `directory = /var/www/proyectos` y guardamos los cambios realizados.
- ✓ Ya para crear la documentación de nuestros proyectos entraremos en la aplicación de PhpDocumentor desde el navegador tecleando `http://localhost/PhpDocumentor/` y en el Menú elegimos la opción `Config`, a continuación, en la lista debemos elegir la configuración y seleccionar `"mi_proyecto.ini"`, para luego presionar `Go`, en esos momentos comenzará a crearse la documentación en el directorio elegido.

Para acceder a la documentación generada deberíamos seleccionar el documento `"index.html"` de la carpeta donde se almacenan los documentos generados y veríamos la página principal de la documentación.

PhpDocumentor puede crear su propia documentación. Los ficheros fuente de esta documentación están en una subcarpeta "`tutorials`". Sin embargo, se debe especificar el directorio raíz de phpDocumentor como directorio de entrada a phpdoc para que se compilen estos tutoriales (phpDocumentor no procesa documentación en "`tutorials`" que no esté vinculada a algún código fuente, es decir, no es posible compilar sólo tutoriales). En este caso como proyecto se debe especificar "`phpDocumentor`".

Esta web sirve como manual de referencia, guía de usuario, tutoriales prácticos, etc. sobre phpDocumentor.

<http://www.phpdoc.org/>

3.- JavaDoc.

Caso práctico

De igual forma que en **BK programación** se ha decidido someter a estudio la aplicación **phpDocumentor** para así poder decidir su implantación, lo mismo ha ocurrido con **JavaDoc**. De esta manera se está investigando el funcionamiento, configuración, limitaciones, etc. de JavaDoc, que nos permitirá desarrollar documentación de forma automatizada de los programas Java que en la empresa se desarrollen.



Se pueden encontrar en la red diversas reglas para la generación de documentación de los programas en Java, cada una de ellas con unas características específicas, aunque todas persiguen el mismo objetivo que es documentar los programas Java para que sean más legibles.

Javadoc es una utilidad de Sun Microsystems empleado para generar APIs (Application Programming Interface) en formato HTML de un archivo de código fuente Java. **Javadoc** es el estándar de la industria para documentar clases de Java, la mayoría de los IDEs los generan automáticamente. Esto facilita la tarea de los desarrolladores, ya que, con sólo seguir una serie de reglas a la hora de generar los comentarios en su código, podrán obtener una buena documentación simplemente usando esta herramienta.

La finalidad de **Javadoc** es intentar evitar que la documentación se quede rápidamente obsoleta, cuando el programa continúa su desarrollo y no se dispone del tiempo suficiente para mantener la documentación al día. Para ello, se pide a los programadores de Java que escriban la documentación básica (clases, métodos, etc.) en el propio código fuente (en comentarios en el propio código), con la esperanza de que esos comentarios sí se mantengan actualizados cuando se cambie el código. La herramienta Javadoc extrae dichos comentarios y genera con ellos un juego de documentación en formato HTML.

Básicamente **Javadoc** es un programa, que recoge los comentarios que se colocan en el código con marcas especiales y construye un archivo HTML con clases, métodos y la documentación que corresponde. Este HTML tiene el formato de toda la documentación estándar de Java provista por Sun.

La documentación a ser utilizada por Javadoc se escribe en comentarios que comienzan con `/**` y que terminan con `*/`, comenzando cada línea del comentario por `*` a su vez, dentro de estos comentarios se puede escribir código HTML y operadores para que interprete **Javadoc** (generalmente precedidos por `@`).

Javadoc localiza las etiquetas incrustadas en los comentarios de un código Java. Estas etiquetas permiten generar una API completa a partir del código fuente con los comentarios. Las etiquetas comienzan con el símbolo `@` y son sensibles a mayúsculas-minúsculas. Una etiqueta se sitúa siempre al principio de una línea, o sólo precedida por espacio(s) y asterisco(s) para que la herramienta Javadoc la interprete como tal. Si no se hace así las interpretará como texto normal.

Hay dos tipos de etiquetas:

- ✓ **Etiquetas de bloque:** sólo se pueden utilizar en la sección de etiquetas que sigue a la descripción principal. Son de la forma: `@etiqueta`
- ✓ **Etiquetas inline:** se pueden utilizar tanto en la descripción principal como en la sección de etiquetas. Son de la forma: `{@tag}`, es decir, se escriben entre los símbolos de llaves.

Javadoc es una utilidad de Sun Microsystems para generar APIs (Application programming Interface) en formato HTML de un archivo de código fuente Java.

La herramienta Javadoc extrae los comentarios del código fuente de los programas Java y genera con ellos un juego de documentación en formato html.

La documentación a ser utilizada por Javadoc se escribe en comentarios que comienzan con `/**` y que terminan con `*/`

Las etiquetas se ubican dentro de los comentarios, comienzan con el símbolo `@` y son sensibles a mayúsculas-minúsculas.

3.1.- Instalación de Javadoc.

Para proceder a la instalación de **Javadoc** vamos a partir de una máquina en la que tenemos instalado la distribución **Debian 6.0.1Squeeze**, igual que hemos hecho en el caso del phpDocumentor.

Previamente a la instalación de **Javadoc**, tendremos en cuenta que estamos realizando la programación Java desde una herramienta IDE como puede ser **Eclipse** o **NetBeans**; sin duda son los dos entornos de desarrollo integrados que más han crecido en los últimos tiempos. Su comunidad de desarrolladores sirve como pilar para su crecimiento y evolución constante. El avance de las nuevas tecnologías, nuevos lenguajes y metodologías en el desarrollo del software hacen que lo nuevo quede viejo en poco tiempo. Esto presiona a los programadores a trabajar de manera más intensa agregando nuevas funcionalidades y perfeccionando sus productos en una competencia por ser el mejor IDE.

Para realizar la instalación de **Eclipse** únicamente ejecutamos desde un terminal:

```
# apt-get install eclipse
```

En el caso de querer realizar la instalación de **NetBeans**, accedemos a la página de NetBeans para realizar la descarga, previamente debemos seleccionar idioma del IDE y plataforma, en este caso español y Linux(x86/x64) respectivamente:

NetBeans.

<http://netbeans.org/downloads/index.html>

Una vez descargado el paquete procedemos del siguiente modo:

- ✓ Creamos una carpeta para la instalación, tal como `/usr/NetBeans69`, para ello:

```
# mkdir /usr/NetBeans69
```

- ✓ Damos permisos de propietario al usuario:

```
# chown -R alumno /usr/NetBeans69
```

- ✓ Copiamos el archivo bajado (`NetBeans-6.9-ml-linux.sh`) a la carpeta creada:

- ✓ Asignamos permisos de ejecución:

```
# chmod a+x /usr/NetBeans69/NetBeans-6.9-ml-linux.sh
```

- ✓ Instalamos el binario:

```
# sh NetBeans-6.9-ml-linux.sh
```



Y una vez aceptada la licencia aparece un "wizard" donde vamos seleccionando la configuración de nuestra instalación.

Tanto Eclipse como NetBeans disponen entre sus opciones la de generar javadoc y mediante diversas ventanas que ofrecen se pueden seleccionar las opciones para **javadoc**. Pero no sólo eso, sino también ofrecen el completado de código javadoc.

Por ejemplo, en el caso de Eclipse, si disponemos del siguiente código (típico ejemplo "*hola mundo*"):

```
public class holamundo {  
    /**  
     * @param args  
     */  
}
```

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub  
    System.out.println("Hola mundo!");  
}
```

Ahora si accedemos a la opción de menú **"Project"** observamos una opción en la que podemos seleccionar **"Generate Javadoc..."** donde nos permite seleccionar el proyecto del que generar la documentación y también la ruta de la carpeta donde generarla; una vez establecidos dichos parámetros se genera la documentación, que podremos consultar accediendo desde un navegador a los documentos html generados; existe un `"index.html"` desde el que podremos iniciar la navegación e ir accediendo a la documentación generada.

La mayor parte de los entornos de desarrollo incluyen un botón para llamar a **javadoc** así como opciones de configuración; no obstante, siempre se puede ir al directorio donde se instaló el JDK y ejecutar javadoc directamente sobre el código fuente Java.

```
# javadoc ejemplo.java
```

3.2.- Documentando con Javadoc.

Los comentarios **JavaDoc** están destinados a describir, principalmente, clases y métodos. Como están pensados para que otro programador los lea y utilice la clase (o método) correspondiente, se decidió fijar, al menos parcialmente, un formato común, de forma que los comentarios escritos por un programador resultaran legibles por otro. Para ello los comentarios Javadoc deben incluir unos indicadores especiales, que comienzan siempre por '@' y se suelen colocar al comienzo de línea. Veamos cómo se introducen los comentarios para Javadoc en la siguiente clase de ejemplo:

```
/**  
 * Una clase para empezar a programar en Java  
 * el típico ejemplo de HolaMundo  
 * @version 1.2.0, 24/07/11  
 * @author Paco Programador Java  
 */  
public class holamundo {  
    /**  
     * Muestra el mensaje de Hola Mundo  
     */  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        System.out.println("Hola mundo!");  
    }  
}
```



Como se ve, y esto es usual en **JavaDoc**, la descripción de la clase o del método no va precedida de ningún indicador. Se usan indicadores para el número de versión (`@version`), el autor (`@author`) y otros. Es importante observar que los indicadores no son obligatorios; por ejemplo, en un método sin parámetros no se incluye obviamente el indicador `@param`. También puede darse que un comentario incluya un indicador más de una vez, por ejemplo varios indicadores `@param` porque el método tiene varios parámetros. Resumiendo, los indicadores más usuales:

- ✓ `@author nombreDelAutor descripción.`
Indica quién escribió el código al que se refiere el comentario. Si son varias personas se escriben los nombres separados por comas o se repite el indicador, según se prefiera. Es normal incluir este indicador en el comentario de la clase y no repetirlo para cada método, a no ser que algún método haya sido escrito por otra persona.
- ✓ `@version númeroVersión descripción.`
Si se quiere indicar la versión. Normalmente se usa para clases, pero en ocasiones también para métodos.
- ✓ `@param nombreParámetro descripción.`
Para describir un parámetro de un método.
- ✓ `@return descripción.`

Describe el valor de salida de un método.

- ✓ `@see nombre descripción.`

Cuando el trozo de código comentado se encuentra relacionada con otra clase o método, cuyo nombre se indica en nombre.

- ✓ `@throws nombreClaseExcepción descripción.`

Cuando un método puede lanzar una excepción ("romperse" si se da alguna circunstancia) se indica así.

- ✓ `@deprecated descripción.`

Indica que el método (es más raro encontrarlos para una clase) ya no se usa y se ha sustituido por otro.

3.3.- Creación y uso de plantillas de código.

Si se usan ciertas convenciones en el código fuente Java (como comenzar un comentario con `/**` y terminarlo con `*/`), **javadoc** puede fácilmente generar páginas HTML con el contenido de esos comentarios, que pueden visualizarse en cualquier navegador. La documentación del API de Java ha sido creada de este modo. Esto hace que el trabajo de documentar el código de nuevas clases Java sea trivial.

Las plantillas:

- ✓ Son sugerencias de código asociadas a palabras clave.
- ✓ Se hallan definidas en **Preferences > Java > Editor > Templates** en Eclipse y en **Herramientas - Opciones - Editor - Plantillas de código** en NetBeans.
- ✓ Es aconsejable examinar todas, ya que pueden ahorrar mucho trabajo.
- ✓ Muchas de ellas utilizan nombres similares a las construcciones Java que encapsulan (`try`, `for`, `while`, `if`,...).
- ✓ Podemos definir y crear nuestras propias plantillas.
- ✓ Además existen plantillas **Javadoc** predefinidas.



Una plantilla se compone de:

- ✓ un nombre,
- ✓ una descripción,
- ✓ un contexto en función del lenguaje (en java, si estamos en el código, en el javadoc,...) y
- ✓ un pattern, que es el código de la plantilla. Dentro del código de la plantilla podemos usar texto fijo o una serie de variables predefinidas, por ejemplo:
 - ➔ `${cursor}`: posición en la que se establecerá el cursor de texto tras desplegar el código de la plantilla.
 - ➔ `${enclosing_type}`: tipo de la clase en la que nos encontramos.
 - ➔ `${enclosing_method}`: nombre del método en el que nos encontramos.
 - ➔ `${year}`: año en curso.
 - ➔ `${time}`: hora en curso.

Estas plantillas se mostrarán como sugerencias en el código tras comenzar a escribir su nombre y pulsar **CTRL+ espacio**. Lo más interesante es que nosotros podemos crearnos nuestras propias plantillas, además de modificar las existentes. Para ello no tenemos más que añadir una nueva desde la opción de **"Templates"**, asignarle un nombre, descripción y elegir el código que queremos que se muestre al seleccionar la misma.

Si nos encontramos trabajando en una empresa de desarrollo de software, como puede ser el caso de **BK programación**; en ese entorno de trabajo, ¿qué importancia y utilidad crees que tendrían las plantillas de código? ¿crees que la reutilización de código es útil, fiable y correcto? ¿cómo crees que afectan los comentarios del código fuente a la reutilización de código?

4.- Sistemas de control de versiones.

Caso práctico

En **BK programación**, debido al incremento de trabajo que están teniendo últimamente, han pensado en ampliar la plantilla. Su intención es ofrecer un contrato de trabajo a **Carlos**, pues sus conocimientos de diseño web serán de utilidad para trabajar como programador en la empresa.

Ada se está dando cuenta que, a medida que la empresa crece, el número de proyectos aumenta al igual que lo va a hacer la plantilla de personal. Por eso cree fundamental instalar un sistema de control de versiones para facilitar la integración del código fuente y demás documentos, de cada uno de los programadores, a los respectivos proyectos en desarrollo; de este modo la integración conjunta del trabajo individual de cada uno de los empleados será más sencilla, controlada y existirán nuevas posibilidades en cuanto a la disposición del código y documentación de los proyectos.

Cuando realizamos un proyecto software es bastante habitual que vayamos haciendo pruebas, modificando nuestros fuentes continuamente, añadiendo funcionalidades, etc. Muchas veces, antes de abordar un cambio importante que requiera tocar mucho código, nos puede interesar guardarnos una versión de los fuentes que tenemos en ese momento, de forma que guardamos una versión que sabemos que funciona y abordamos, por separado, los cambios.

Si no usamos ningún tipo de herramienta que nos ayude a hacer esto, lo más utilizado es directamente hacer una copia de los fuentes en un directorio separado. Luego empezamos a tocar. Pero esta no es la mejor forma. Hay herramientas, los sistemas de control de versiones, que nos ayudan a guardar las distintas versiones de los fuentes.

Con un sistema de control de versiones hay un directorio, controlado por esta herramienta, donde se van guardando los fuentes de nuestro proyecto con todas sus versiones. Usando esta herramienta, nosotros sacamos una copia de los fuentes en un directorio de trabajo, ahí hacemos todos los cambios que queramos y, cuando funcionen, le decimos al sistema de control de versiones que nos guarde la nueva versión. El sistema de control de versiones suele pedirnos que metamos un comentario cada vez que queremos guardar fuentes nuevos o modificados.

También, con esta herramienta, podemos obtener fácilmente cualquiera de las versiones de nuestros fuentes, ver los comentarios que pusimos en su momento e, incluso, comparar distintas versiones de un mismo fuente para ver qué líneas hemos modificado.

Aunque los sistemas de control de versiones se hacen imprescindibles en proyectos de cierta envergadura y con varios desarrolladores, de forma que puedan mantener un sitio común con las versiones de los fuentes a través de un sistema de control de versiones, también puede ser útil para un único desarrollador en su casa, de forma que siempre tendrá todas las versiones de su programa controladas.

Los sistemas de control de versiones son programas que permiten gestionar un repositorio de archivos y sus distintas versiones; utilizan una arquitectura cliente-servidor en donde el servidor guarda la(s) versión(es) actual(es) del proyecto y su historia. Sirven para mantener distintas versiones de un fichero, normalmente código fuente, documentación o ficheros de configuración.

"Hay que unirse, no para estar juntos, sino para hacer algo juntos. "

Juan Donoso Cortés (1808-1853); Ensayista español.

4.1.- Conceptos básicos de sistemas de control de versiones.

Existen una serie de conceptos necesarios para comprender el funcionamiento de los sistemas de control de versiones, entre los cuales destacamos los siguientes:

- ✓ **Revisión:** Es una visión estática en el tiempo del estado de un grupo de archivos y directorios. Posee una etiqueta que la identifica. Suele tener asociado metadatos ("datos sobre datos". o "informaciones sobre datos") como pueden ser:
 - ➔ Identidad de quién hizo las modificaciones.
 - ➔ Fecha y hora en la cual se almacenaron los cambios.
 - ➔ Razón para los cambios.
 - ➔ De qué revisión y/o rama se deriva la revisión.
 - ➔ Palabras o términos clave asociados a la revisión.
- ✓ **Copia de trabajo:** También llamado "*Árbol de trabajo*", es el conjunto de directorios y archivos controlados por el sistema de control de versiones, y que se encuentran en edición activa. Está asociado a una rama de trabajo concreta.
- ✓ **Rama de trabajo(o desarrollo):** En el más sencillo de los casos, una rama es un conjunto ordenado de revisiones. La revisión más reciente se denomina principal (main) o cabeza. Las ramas se pueden separar y juntar según como sea necesario, formando un grafo de revisión.
- ✓ **Repositorio:** Lugar en donde se almacenan las revisiones. Físicamente puede ser un archivo, colección de archivos, base de datos, etc.; y puede estar almacenado en local o en remoto (servidor).
- ✓ **Conflicto:** Ocurre cuando varias personas han hecho cambios contradictorios en un mismo documento (o grupo de documentos); los sistemas de control de versiones solamente alertan de la existencia del conflicto. El proceso de solucionar un conflicto se denomina **resolución**.
- ✓ **Cambio:** Modificación en un archivo bajo control de revisiones. Cuando se unen los cambios en un archivo (o varios), generando una revisión unificada, se dice que se ha hecho una **combinación** o integración.
- ✓ **Parche:** Lista de cambios generada al comparar revisiones, y que puede usarse para reproducir automáticamente las modificaciones hechas en el código.

Con el empleo de los sistemas de control de versiones se consigue mantener un repositorio con la información actualizada. La forma habitual de trabajar consiste en mantener una copia en local y modificarla. Después actualizarla en el repositorio. Como ventaja tenemos que no es necesario el acceso continuo al repositorio.

Algunos sistemas de control de versiones permiten trabajar directamente contra el repositorio; en este caso tenemos como ventaja un aumento de la transparencia, a pesar de que como desventaja existe el bloqueo de ficheros.



Supongamos que estamos 3 programadores aportando código al mismo proyecto dentro de la empresa **BK programación**; para la integración del código se emplea un sistema de control de versiones; ¿en qué caso se pueden producir conflictos?, ¿cómo soluciona el sistema de control de versiones el conflicto?

El conflicto se va a producir cuando más de un programador intente integrar cambios en el mismo código. La herramienta de control de versiones no soluciona el conflicto, sólo informa de su existencia.

4.2.- Procedimiento de uso habitual de un sistema de control de versiones.

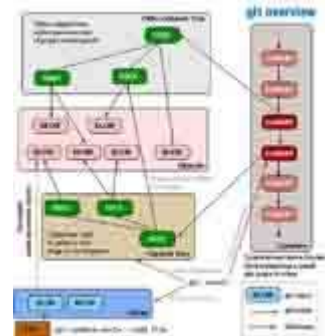
El funcionamiento general de un sistema de control de versiones sigue el siguiente ciclo de operaciones:

- ✓ Descarga de ficheros inicial (Checkout):
 - ➔ El primer paso es bajarse los ficheros del repositorio.
 - ➔ El "checkout" sólo se hace la primera vez que se usan esos ficheros.
- ✓ Ciclo de trabajo habitual:

- ➔ Modificación de los ficheros, para aplicar los cambios oportunos como resultado de la aportación de cada una de las personas encargadas de manipular el código de los ficheros.
- ➔ Actualización de ficheros en local (Update): Los ficheros son modificados en local y, posteriormente, se sincronizan con los ficheros existentes en el repositorio.
- ➔ Resolución de conflictos (si los hay): Como resultado de la operación anterior es cuando el sistema de control de versiones detectará si existen conflictos, en cuyo caso informa de ello y siendo los usuarios implicados en la manipulación del código afectado por el conflicto, los encargados de solucionarlo.
- ➔ Actualización de ficheros en repositorio (Commit): Consiste en la modificación de los ficheros en el repositorio; el sistema de control de versiones comprueba que las versiones que se suben estén actualizadas.

Como conclusión, vemos que los sistemas de control de versiones permiten las siguientes funciones:

- ✓ Varios clientes pueden sacar copias del proyecto al mismo tiempo.
- ✓ Realizar cambios a los ficheros manteniendo un histórico de los cambios:
 - ➔ Deshacer los cambios hechos en un momento dado.
 - ➔ Recuperar versiones pasadas.
 - ➔ Ver históricos de cambios y comentarios.
- ✓ Los clientes pueden también comparar diferentes versiones de archivos.
- ✓ Unir cambios realizados por diferentes usuarios sobre los mismos ficheros.
- ✓ Sacar una "foto" histórica del proyecto tal como se encontraba en un momento determinado.
- ✓ Actualizar una copia local con la última versión que se encuentra en el servidor. Esto elimina la necesidad de repetir las descargas del proyecto completo.
- ✓ Mantener distintas ramas de un proyecto.



Debido a la amplitud de operaciones que el sistema de control de versiones permite aplicar sobre los proyectos que administra, muchos de los sistemas de control de versiones ofrecen establecer algún método de autorización, es decir, la posibilidad por la cual a ciertas personas se les permite, o no, realizar cambios en áreas específicas del repositorio.

Algunos proyectos utilizan un sistema basado en el honor: cuando a una persona se le permite la posibilidad de realizar cambios, aunque sea a una pequeña área del repositorio, lo que reciben es una contraseña que le permite realizar cambios en cualquier otro sitio del repositorio y sólo se les pide que mantenga sus cambios en su área. Hay que recordar que no existe ningún peligro aquí; de todas formas, en un proyecto activo, todos los cambios son revisados. Si alguien hace un cambio donde no debía, alguien más se dará cuenta y dirá algo. Es muy sencillo, si un cambio debe ser rectificado todo está bajo el control de versiones de todas formas, así que sólo hay que volver atrás.

Entre las funciones que permite realizar un sistema de control de versiones cabe destacar:

- ✓ Sacar copias del proyecto al mismo tiempo.
- ✓ Realizar cambios a los ficheros manteniendo un histórico de los cambios.
- ✓ Los clientes pueden también comparar diferentes versiones de archivos.
- ✓ Unir cambios realizados por diferentes usuarios sobre los mismos ficheros.
- ✓ Sacar una "foto" histórica del proyecto tal como se encontraba en un momento determinado.
- ✓ Actualizar una copia local con la última versión que se encuentra en el servidor.
- ✓ Mantener distintas ramas de un proyecto.

4.3.- Sistemas de control de versiones centralizados y distribuidos.

El método de control de versiones usado por mucha gente es copiar los archivos a otro directorio controlando la fecha y hora en que lo hicieron. Este enfoque es muy común porque es muy simple, pero también tremendamente propenso a errores. Es fácil olvidar en qué directorio nos encontramos, y guardar accidentalmente en el archivo equivocado o sobrescribir archivos que no queríamos. Para hacer frente a este problema, los programadores desarrollaron hace tiempo **sistemas de control de versiones locales** que contenían una simple base de datos en la que se llevaba registro de todos los cambios realizados sobre los archivos.

Una de las herramientas de control de versiones más popular fue un sistema llamado rcs. Esta herramienta funciona básicamente guardando conjuntos de parches (es decir, las diferencias entre archivos) de una versión a otra en un formato especial en disco; puede entonces recrear cómo era un archivo en cualquier momento sumando los distintos parches.



El siguiente gran problema que se encuentra la gente es que necesitan colaborar con desarrolladores en otros sistemas. Para solventar este problema, se desarrollaron los **sistemas de control de versiones centralizados** (Centralized Version Control Systems o CVCSs en inglés). Estos sistemas, como CVS, Subversion y Perforce, tienen un único servidor que contiene todos los archivos versionados, y varios clientes que descargan los archivos de ese lugar central. Durante muchos años, éste ha sido el estándar para el control de versiones.

Esta configuración ofrece muchas ventajas, especialmente frente a VCSs locales. Por ejemplo, todo el mundo sabe hasta cierto punto en qué está trabajando el resto de gente en el proyecto. Los administradores tienen control detallado de qué puede hacer cada uno; y es mucho más fácil administrar un CVCS que tener que lidiar con bases de datos locales en cada cliente.

Sin embargo, esta configuración también tiene serias desventajas. La más obvia es el punto único de fallo que representa el servidor centralizado. Si ese servidor se cae durante una hora, entonces durante esa hora nadie puede colaborar o guardar cambios versionados de aquello en que están trabajando. Si el disco duro en el que se encuentra la base de datos central se corrompe, y no se han llevado copias de seguridad adecuadamente, se pierde absolutamente todo, toda la historia del proyecto salvo aquellas instantáneas que la gente pueda tener en sus máquinas locales.

Es aquí donde entran los **sistemas de control de versiones distribuidos** (Distributed Version Control Systems o DVCSs en inglés). En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no sólo descargan la última instantánea de los archivos sino que replican completamente el repositorio. Así, si un servidor muere, y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios de los clientes puede copiarse en el servidor para restaurarlo. Cada vez que se descarga una instantánea, en realidad se hace una copia de seguridad completa de todos los datos.

Es más, muchos de estos sistemas se las arreglan bastante bien teniendo varios repositorios con los que trabajar, por lo que se puede colaborar con distintos grupos de gente de maneras distintas simultáneamente dentro del mismo proyecto. Esto permite establecer varios tipos de flujos de trabajo que no son posibles en sistemas centralizados, como pueden ser los modelos jerárquicos.

4.4.- Git como sistema de control de versiones.

El núcleo de Linux es un proyecto de software de código abierto con un alcance bastante grande. Durante la mayor parte del mantenimiento del núcleo de Linux (1991-2002), los cambios en el software se pasaron en forma de parches y archivos. En 2002, el proyecto del núcleo de Linux empezó a usar un DVCS propietario llamado BitKeeper.

En 2005, la relación entre la comunidad que desarrollaba el núcleo de Linux y la compañía que desarrollaba BitKeeper se vino abajo, y la herramienta dejó de ser ofrecida gratuitamente. Esto impulsó a la comunidad de desarrollo de Linux (y en particular a Linus Torvalds, el creador de Linux) a desarrollar su propia herramienta basada en algunas de las lecciones que aprendieron durante el uso de BitKeeper. Algunos de los objetivos del nuevo sistema fueron los siguientes:

- ✓ Velocidad.
- ✓ Diseño sencillo.
- ✓ Fuerte apoyo al desarrollo no lineal (miles de ramas paralelas).
- ✓ Completamente distribuido.
- ✓ Capaz de manejar grandes proyectos como el núcleo de Linux de manera eficiente (velocidad y tamaño de los datos).

Git es un sistema rápido de control de versiones, está escrito en C y se ha hecho popular sobre todo a raíz de ser el elegido para el kernel delinux.

Desde su nacimiento en 2005, **Git** ha evolucionado y madurado para ser fácil de usar y, aún así, conservar estas cualidades iniciales. Es tremendamente rápido, muy eficiente con grandes proyectos, y tiene un increíble sistema de ramificación (branching) para desarrollo no lineal.

La principal diferencia entre **Git** y cualquier otro VCS (Subversion y compañía incluidos) es cómo Git modela sus datos. Conceptualmente, la mayoría de los demás sistemas almacenan la información como una lista de cambios en los archivos. Estos sistemas (CVS, Subversion, Perforce, Bazaar, etc.) modelan la información que almacenan como un conjunto de archivos y las modificaciones hechas sobre cada uno de ellos a lo largo del tiempo.

Git no modela ni almacena sus datos de este modo, modela sus datos más como un conjunto de instantáneas de un mini sistema de archivos. Cada vez que confirmas un cambio, o guardas el estado de tu proyecto en Git, él básicamente hace una "foto" del aspecto de todos tus archivos en ese momento, y guarda una referencia a esa instantánea. Para ser eficiente, si los archivos no se han modificado, Git no almacena el archivo de nuevo, sólo un enlace al archivo anterior idéntico que ya tiene almacenado.



Casi cualquier operación es local, la mayoría de las operaciones en **Git** sólo necesitan archivos y recursos locales para operar; por ejemplo, para navegar por la historia del proyecto, no se necesita salir al servidor para obtener la historia y mostrarla, simplemente se lee directamente de la base de datos local. Esto significa que se ve la historia del proyecto casi al instante. Si es necesario ver los cambios introducidos entre la versión actual de un archivo y ese archivo hace un mes, **Git** puede buscar el archivo hace un mes y hacer un cálculo de diferencias localmente, en lugar de tener que pedirle a un servidor remoto que lo haga, u obtener una versión antigua del archivo del servidor remoto y hacerlo de manera local.

Aparte de todo lo anterior, **Git** posee integridad debido a que todo es verificado mediante una suma de comprobación antes de ser almacenado, y es identificado a partir de ese momento mediante dicha suma. Esto significa que es imposible cambiar los contenidos de cualquier archivo o directorio sin que Git lo detecte. Como consecuencia de ello es imposible perder información durante su transmisión o sufrir corrupción de archivos sin que **Git** sea capaz de detectarlo.

En la siguiente tabla se resume el concepto de las herramientas de control de versiones, entre ellas GIT.

Sistema de Control de Versiones	
¿QUÉ SON?	Se llama control de versiones a la <u>gestión de los diversos cambios que se realizan</u> sobre los elementos <u>de algún producto o una configuración del mismo</u> En informática los Sistemas de Control de Versiones se encargan de controlar las distintas versiones del código fuente sobre un proyecto de desarrollo de software determinado
TIPOS DE SISTEMAS DE CONTROL DE VERSIONES	✓ Centralizados: ➔ Único repositorio ➔ Almacena todo el código del proyecto ➔ Lo administra un único usuario o grupo de ellos ➔ Ejemplos: CVS, Subversion ✓ Distribuidos: ➔ Cada usuario tiene su repositorio ➔ Los distintos repositorios pueden intercambiar y mezclar revisiones ➔ Ejemplos: Git, Mercurial
CONCEPTOS RELACIONADOS	✓ Repositorio: Lugar en el que se almacenan los datos actualizados e históricos del proyecto ✓ Revisión: Es una visión estática en el tiempo del estado de un grupo de archivos y directorios ✓ Rama (branch): Un módulo o proyecto puede ser bifurcado en un determinado momento (ramificado), y a partir de ahí las dos copias del proyecto o módulo pueden sufrir distintas transformaciones de modo independiente ✓ Integración, unión o merge: Une 2 conjuntos de cambios sobre uno o varios ficheros en una revisión unificada
ORIGEN Y DESARROLLO DE GIT	✓ Durante el desarrollo del <u>núcleo de Linux se empezó utilizando</u> un Sistema de Control de Versiones Distribuido llamado <u>Bitkeeper</u>. ✓ En <u>2005 Bitkeeper dejó de ser gratuito</u> y Linus Torvalds, junto con su equipo, decidieron desarrollar su propia herramienta de control de versiones, de donde <u>surge GIT</u> ✓ <u>Git es un sistema rápido de control de versiones, escrito en C y se ha hecho popular sobre todo a raíz de ser elegido para el kernel de Linux.</u> ✓ <u>Git es rápido, muy eficiente</u> con grandes proyectos, y tiene un <u>increíble sistema de ramificación</u>. <u>Posee integridad</u> debido a que todo es verificado mediante una suma de comprobación antes de ser almacenado, y es identificado a partir de ese momento mediante dicha suma.
IMPORTANCIA DE GIT	La autoridad y poder que ha adquirido Git se observa simplemente con ver diversos proyectos que lo utilizan, entre ellos: ✓ Android, Debian, Fedora, Eclipse, CakePHP, GNOME, OpenSUSE, PostgreSQL, Ruby on Rails, Samba, VLS A parte de lo anterior GitHub: ✓ Es un servicio de hospedaje web para proyectos que utilizan el sistema de control de versiones Git. GitHub ofrece tanto planes comerciales como planes gratuitos para proyectos de código abierto

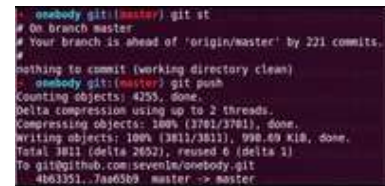
4.5.- Funcionamiento de Git.

Git tiene tres estados principales en los que se pueden encontrar los archivos: confirmado (`committed`), modificado (`modified`) y preparado (`staged`).

- ✓ **Confirmado** significa que los datos están almacenados de manera segura en la base de datos local.
- ✓ **Modificado** estado en el que se ha modificado el archivo pero todavía no se ha confirmado a la base de datos.
- ✓ **Preparado** significa que se ha marcado un archivo modificado en su versión actual para que vaya en la próxima confirmación.

Esto nos lleva a las tres secciones principales de un proyecto de **Git**:

- ✓ El **directorio de Git** (`Git directory`): Almacena los metadatos y la base de datos de objetos para tu proyecto. Es la parte más importante de Git, y es lo que se copia cuando se clona un repositorio desde otro ordenador.
- ✓ El **directorio de trabajo** (`working directory`): Es una copia de una versión del proyecto. Estos archivos se sacan de la base de datos comprimida en el directorio de Git, y se colocan en disco para que se pueda usar o modificar.
- ✓ El **área de preparación** (`staging area`): es un sencillo archivo, generalmente contenido en tu directorio de Git, que almacena información acerca de lo que va a ir en la próxima confirmación. A veces se denomina **índice**, pero se está convirtiendo en estándar el referirse a ello como el **área de preparación**.



```
~$ git:(master) git st
# On branch master
# Your branch is ahead of 'origin/master' by 221 commits.
nothing to commit (working directory clean)
~$ git:(master) git push
Counting objects: 4255, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (3781/3781), done.
Writing objects: 100% (3811/3811), 998.09 KiB, done.
Total 3811 (delta 2652), reused 0 (delta 1)
To git@github.com:seven1n/seven1n.git
4863351..7ad65b9 master -> master
```

El flujo de trabajo básico en Git consiste en:

1. Modificar una serie de archivos en el directorio de trabajo.
2. Preparar los archivos, añadiendo instantáneas de ellos al área de preparación.
3. Confirmar los cambios, lo que toma los archivos tal y como están en el área de preparación, y almacena esa instantánea de manera permanente en el directorio de Git.

Si una versión concreta de un archivo está en el directorio de Git, se considera **confirmada** (`committed`). Si ha sufrido cambios desde que se obtuvo del repositorio, pero ha sido añadida al área de preparación, está **preparada** (`staged`). Y si ha sufrido cambios desde que se obtuvo del repositorio, pero no se ha preparado, está **modificada** (`modified`).

Observa el siguiente vídeo:

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

Se trata de un impresionante vídeo que forma parte de una serie de vídeos de **Git** y **GitHub** realizados por un excelente profesional que es Jesús Conde, que lo demuestra en la calidad de sus vídeo-tutoriales. Este vídeo demuestra cómo realizar el trabajo básico con **Git**, mediante sus comandos fundamentales que permitirán:

- ✓ Configurar e inicializar un repositorio.
- git init - git clone
- ✓ Empezar y detener el rastreo de archivos.
- ✓ Guardar cambios en `stage` y `commit`.
- ✓ Ignorar ciertos archivos y patrones.
- ✓ Deshacer errores.
- ✓ Navegar por el historial del proyecto.
- ✓ Hacer Push y Pull en repositorios remotos.

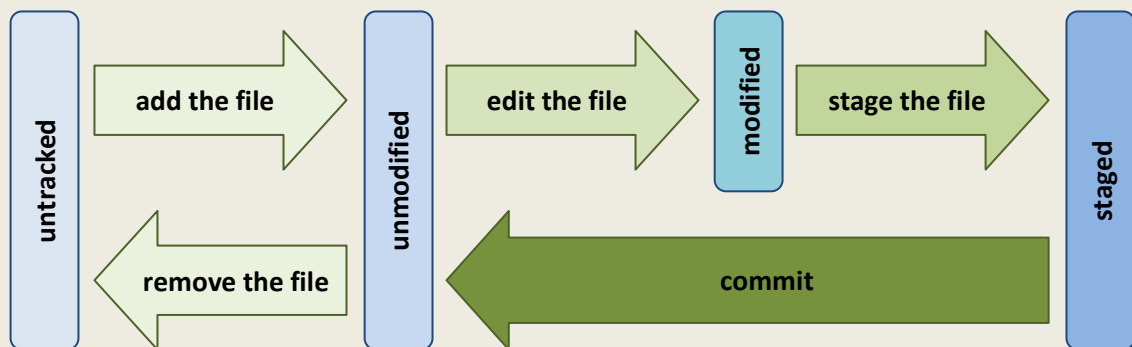
Empieza mostrando como obtener un repositorio Git, empleando 2 métodos:

1. Proyecto creado en un directorio existente e importarlo dentro de Git; comando `git init`
 - a. Crear carpeta para proyecto.
 - b. Dentro del directorio creado ejecutar `git init` automáticamente se crea un directorio `.git` con una serie de archivos y carpetas encargadas de llevar a cabo el control de versiones del proyecto.
 - c. Para añadir archivos al proyecto se emplea el comando `git add` seguido del archivo a añadir y luego ejecutamos el comando `git commit`. Tener en cuenta lo que ocurre si no se ha hecho el `git init` cómo se demuestra en el vídeo.
2. Clonar un repositorio existente desde un servidor; comando `git clone` con lo que se obtiene una copia exacta del proyecto del servidor.
 - a. La sintaxis es: `git clone "protocolo://url del proyecto a clonar"` ; tener en cuenta la carpeta en la que estamos situados.

Los archivos en estado tracked (**rastreado**) en este caso los archivos pueden estar en estado:

- ✓ **Unmodified** en este caso los archivos se encuentran tal cual como están en el repositorio.
- ✓ **Modified** en este caso hemos realizado alguna modificación de los archivos que hemos bajado del repositorio.
- ✓ **Staged** cuando hemos realizado el `git add` sobre el archivo para que se añada al repositorio en el próximo `commit`.

Los archivos en estado **untracked (no rastreado)**, archivos nuevos en la carpeta del proyecto sobre los que no se ha realizado ninguna operación "git" para añadirlos al repositorio. Los estados anteriores se explican siguiendo un ciclo de vida para los archivos, mostrando cómo van cambiando.



El comando `git status` muestra cómo se encuentran los archivos de nuestro proyecto en relación al repositorio del servidor. Si existe la necesidad de excluir archivos del directorio del proyecto para que estos no sean rastreados por GIT, se emplea el archivo `.gitignore` que se demuestra en el vídeo cómo hacerlo empleando los patrones:

<http://www.jedit.org/users-guide/globs.html>

4.6.- Instalación de Git.

Procederemos a instalar Git en una máquina con Debian 6.0.1Squeeze.

Suele ser útil instalar Git desde código fuente, porque obtendremos la versión más reciente. Cada versión de Git tiende a incluir útiles mejoras en la interfaz de usuario, por lo que utilizar la última versión es a menudo el camino más adecuado, realizando para ello la compilación de software desde código fuente.

Para instalar Git, es necesario disponer de las siguientes librerías de las que Git depende: *curl*, *zlib*, *openssl*, *expat* y *libiconv*; para instalarlas podemos ejecutar:

```
#apt-get install libcurl4-gnutls-dev libexpat1-dev gettext libz-dev
```

Una vez instaladas las librerías anteriores procederemos a realizar la descarga de Git desde su página, pudiendo emplear para ello:

```
#wget http://kernel.org/pub/software/scm/git/git-1.7.6.tar.bz2
```

siendo la versión 1.7.6 la más reciente en este momento.

También podríamos dirigirnos a su web y realizar manualmente la descarga:

<http://git-scm.com>

Una vez haya finalizado la descarga procederemos a la compilación e instalación del paquete, podemos seguir los siguientes pasos:

```
#tar -xjvf git-1.7.6.tar.bz2
#cd git-1.7.6
#apt-get build-dep git-core
#apt-get install libssl-dev
#make prefix=/usr/local all doc
#make prefix=/usr/local install install-doc
```

Una vez hecho esto, también es posible obtener Git a través del propio Git para futuras actualizaciones, empleando para ello el siguiente comando de manera que descargaría automáticamente el código fuente desde su repositorio:

```
# git clone git://git.kernel.org/pub/scm/git/git.git
```

Si queremos instalar Git en Linux a través de un instalador binario, en general puedes hacerlo a través de la herramienta básica de gestión de paquetes que trae Debian en nuestro caso, aunque también podríamos realizar la instalación mediante el comando:

```
# apt-get install git-core
```

Sea una forma u otra la que hayamos decidido para realizar la instalación de Git, para comprobar si se ha realizado correctamente comprobamos con el siguiente comando:

```
# git --version
```

que en nuestro caso debería devolver "**git version 1.7.6**".

4.7.- Configuración de Git (I).

Las opciones de configuración reconocidas por **Git** pueden distribuirse en dos grandes categorías: las del lado cliente y las del lado servidor. La mayoría de las opciones que permiten configurar las preferencias personales de trabajo están en el lado cliente. Aunque hay multitud de ellas, aquí vamos a ver solamente unas pocas, nos centraremos en las más comúnmente utilizadas y en las que afectan significativamente a la forma personal de trabajar. Para consultar una lista completa con todas las opciones contempladas en la versión instalada de Git, se puede emplear el siguiente comando:

```
$ git config --help
```

Git trae una herramienta llamada **git config** que permite obtener y establecer variables de configuración que controlan el aspecto y funcionamiento de **Git**.

Lo primero que se debe hacer cuando se instala **Git** es establecer el nombre de usuario y dirección de correo electrónico. Esto es importante porque las confirmaciones de cambios (commits) en Git usan esta información, y es introducida de manera inmutable en los commits que el usuario va a enviar:

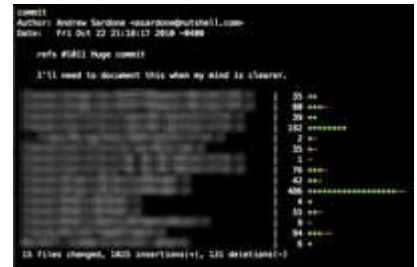
```
$ git config --global user.name "alumno"
$ git config --global user.email alumno@example.com
```



Solamente se necesita hacer esto una vez si se especifica la opción `--global`, ya que Git siempre usará esta información para todo lo que se haga en ese sistema. En el caso de querer sobrescribir esta información con otro nombre o dirección de correo para proyectos específicos, puedes ejecutar el mismo comando sin la opción `--global` cuando estemos en el proyecto concreto.

Una vez que la identidad está configurada, podemos elegir el editor de texto por defecto que se utilizará cuando Git necesite que introduzcamos un mensaje. Si no se indica nada, Git usa el editor por defecto del sistema que, generalmente, es **Vi** o **Vim**. En el caso de querer usar otro editor de texto, como **emacs**, podemos hacer lo siguiente:

```
$ git config --global core.editor emacs
```



Otra opción útil que puede ser interesante configurar es la herramienta de diferencias por defecto, usada para resolver conflictos de unión (merge). Supongamos que quisiéramos usar **vimdiff**:

```
$ git config --global merge.tool vimdiff
```

Git acepta *kdiff3*, *tkdiff*, *meld*, *xxdiff*, *emerge*, *vimdiff*, *gvimdiff*, *ecmerge* y *opendiff* como herramientas válidas. En cualquier momento podremos comprobar la configuración que tenemos mediante el comando:

```
$ git config --list
```

Cuando necesitemos ayuda utilizando Git tenemos tres modos de conseguir ver su página del manual (manpage) para cualquier comando:

```
$ git help <comando>
$ git <comando> --help
$ man git-<comando>
```

4.7.1.- Configuración de Git (II).

Debido a la importancia que actualmente poseen las interfaces web, pasaremos a instalar y configurar el entornoweb de Git, éste integra un aspecto más intuitivo y cómodo para el usuario.

Partimos de que en nuestra máquina Debian ya está instalado el servidor web **Apache**, de manera que pasamos a instalar el entorno web de Git mediante el comando:

```
# apt-get install gitweb
```



A continuación vamos a crear los siguientes directorios para estructurar nuestro modo de trabajo con Git:

```
# mkdir /home/usuario/git
# mkdir /home/usuario/www_git
```

Lo siguiente que debemos realizar es editar el archivo de configuración de `gitweb` en el directorio de configuración de Apache:

```
# nano /etc/apache2/conf.d/gitweb
```

Debemos escribir las siguientes líneas en este archivo y comentar las existentes:

```
Alias /git /home/usuario/www_git
<Directory /home/usuario/www_git >
  Allow from all
  AllowOverride all
  Order allow,deny
  Options +ExecCGI
  DirectoryIndex gitweb.cgi
```

```
<files gitweb.cgi >
  SetHandler cgi-script
</files>
</directory>
SetEnv GITWEB_CONFIG /etc/gitweb.conf
```

A continuación, mover los archivos básicos del gitweb al directorio de trabajo Apache creado anteriormente:

```
# mv -v /usr/share/gitweb/* /home/usuario/www_git
# mv -v /usr/lib/cgi-bin/gitweb.cgi /home/usuario/www_git
```

Y hacemos los siguientes cambios en el archivo de configuración del **gitweb**:

```
#nano /etc/gitweb.conf
$projectroot = '/home/usuario/git/';
$git temp = "/tmp";
#$home link = $my uri || "/";
$home_text = "indextext.html";
$projects_list = $projectroot;
$stylesheet = "/git/gitweb.css";
$logo = "/git/git-logo.png";
$favicon = "/git/git-favicon.png";
```

Por último, recargamos el apache:

```
# /etc/init.d/apache2 reload
```

4.8.- Trabajando con Git (I).

Lo siguiente que debemos hacer es crear la carpeta del proyecto:

```
# cd /var/cache/git/
# mkdir proyecto.git
# cd proyecto.git
```

Luego, iniciamos un repositorio para nuestro nuevo proyecto y lo configuramos de acuerdo a nuestras necesidades:

```
# git init
```

al ejecutar el comando anterior nos devuelve un mensaje similar a "Initialized empty Git repository in /var/cache/git/proyecto.git/.git/", podemos comprobar que en el directorio `.git` hay una serie de archivos asociados al proyecto que se ha creado automáticamente.

```
# echo "Una breve descripción del proyecto" > .git/description
# git config --global user.name "Tu nombre"
# git config --global user.email "tu@correo.com"
```

si hemos creado documentos nuevos en nuestro proyecto antes de hacer el "commit", ejecutamos el siguiente comando para que los archivos presentes en la carpeta del proyecto sean añadidos al repositorio:

```
#git add .
```

Una vez incorporados los ficheros nuevos realizamos el "commit":

```
# git commit -a
```

Para marcar un repositorio como exportado se usa el archivo

`git-daemon-export-ok`:

```
# cd /var/cache/git/proyecto.git
# touch .git/git-daemon-export-ok
```

Para iniciar el servicio de Git que ejecuta un servidor para hacer público nuestro repositorio, ejecutamos el siguiente comando:

```
# git daemon --base-path=/var/cache/git --detach --syslog --export-all
```



Ahora el repositorio se encuentra corriendo en el puerto 9418 de nuestro computador; podemos comprobarlo ejecutando:

```
#netstat -nautp | grep git
```

Por último, le daremos permisos de escritura a un usuario que no sea root, de tal manera que con dicho usuario se puedan hacer cambios remotos en el repositorio:

```
# adduser usuariogit
# passwd usuariogit
# chown -Rv usuariogit:usuariogit /var/cache/git/proyecto.git
```

Para acceder al repositorio podemos hacerlo de la manera convencional, basta con ejecutar el comando:

```
#git clone git://servidor/proyecto.git proyecto
```

O, también, podemos acceder vía web; mediante la `http://localhost/git/`

El comando `git commit` sólo seguirá la pista de los archivos que estaban presentes la primera vez que se ejecutó `git add`. Si son añadidos nuevos archivos o subdirectorios, debe indicarse al Git mediante el siguiente comando:

```
$ git add ARCHIVOSNUEVOS
```

De manera similar, si queremos que Git se olvide de determinados archivos, porque (por ejemplo) han sido borrados:

```
$ git rm ARCHIVOSVIEJOS
```

Renombrar un archivo es lo mismo que eliminar el nombre anterior y agregar el nuevo. También puedes usar `git mv` que tiene la misma sintaxis que el comando `mv`. Por ejemplo:

```
$ git mv ARCHIVOVIEJO ARCHIVONUEVO
```

4.8.1.- Trabajando con Git (II).

Algunas veces es interesante ir hacia atrás y borrar todos los cambios a partir de cierto punto porque, a lo mejor, estaban todos mal. Entonces, utilizaremos para ello el comando:

```
$ git log
```

Ese comando muestra una lista de commits recientes, y sus hashes SHA1. A continuación, debemos escribir:

```
$ git reset --hard SHA1_HASH
```

con él recuperamos el estado de un commit dado y se borran para siempre cualquier recuerdo de commits más nuevos. Otras veces es necesario saltar a un estado anterior temporalmente. En ese caso hay que escribir:

```
$ git checkout SHA1_HASH
```

este comando nos lleva atrás en el tiempo, sin tocar los commits más nuevos. Y con el comando:

```
$ git checkout master
```

podemos volver al presente. También existe la posibilidad de restaurar sólo archivos o directorios en particular, para ello los agregamos al final del comando:

```
$ git checkout SHA1_HASH algun.archivo otro.archivo
```

Esta forma de **checkout** puede sobrescribir archivos sin avisar. Para prevenir accidentes, es recomendable hacer commit antes de ejecutar cualquier comando de checkout.

Podemos obtener una copia de un proyecto administrado por git escribiendo:

```
$ git clone git://servidor/ruta/a/los/archivos
```

en el caso de ya tener una copia de un proyecto usando **git clone**, podemos actualizar a la última versión con:

```
$ git pull
```

Supongamos que estamos en un grupo de desarrollo y hemos realizado un script que nos gustaría compartir con otros. Para hacer esto con Git, en el directorio donde guardamos el script ejecutamos:

```
$ git init  
$ git add .  
$ git commit -m "Primer envío"
```

con lo cual, si los demás miembros del grupo de desarrollo ejecutan:

```
$ git clone tu.maquina:/ruta/al/script
```

podrán descargar el script. Esto asume que tienen acceso por **ssh**. Si no es así, ejecutamos **git daemon** y los demás desarrolladores utilizarían para obtener una copia del script:

```
$ git clone git://tu.maquina/ruta/al/script
```

De aquí en adelante, cada vez que modifiquemos el script y creemos que está listo para el lanzamiento, ejecutaremos:

```
$ git commit -a -m "Siguiendo envío"
```

y los demás desarrolladores pueden actualizar su versión yendo al directorio que tiene el script y ejecutando:

```
$ git pull
```

En el caso de que queramos averiguar qué cambios hicimos desde el último commit ejecutaremos:

```
$ git diff
```

Otra de las ventajas de Git reside en la facilidad de crear nuevas ramas de trabajo, llamadas **branch**, donde probar nuevas características y hacer cambios complejos sin que afecte a la rama de trabajo principal. Luego, el proceso de fusión (**merge**), o la vuelta a un estado anterior es igual de fácil.

Github.com es una web donde alojar proyectos utilizando el sistema de control de versiones Git. Esta web ofrece, aparte del almacenaje de proyectos, herramientas para "socializarlos", como pueden ser feeds rss, wikis o gráficos de cómo los desarrolladores trabajan en sus repositorios.

<https://www.github.com/>

Para empezar a usar Git, Github.com pone a disposición de quien los necesite, varios manuales en inglés tanto para configurar Git como crear nuestros primeros repositorios en Github.com

4.9.- Seguridad documentación en Git.

Git se encuadra en la categoría de los sistemas de gestión de código fuente distribuida. Con Git, cada directorio de trabajo local es un repositorio completo no dependiente, de acceso a un servidor o a la red. El acceso a cada repositorio se realiza a través del protocolo de Git, montado sobre ssh, o usando HTTP, aunque no es necesario ningún servidor web para poder publicar el repositorio en la red.

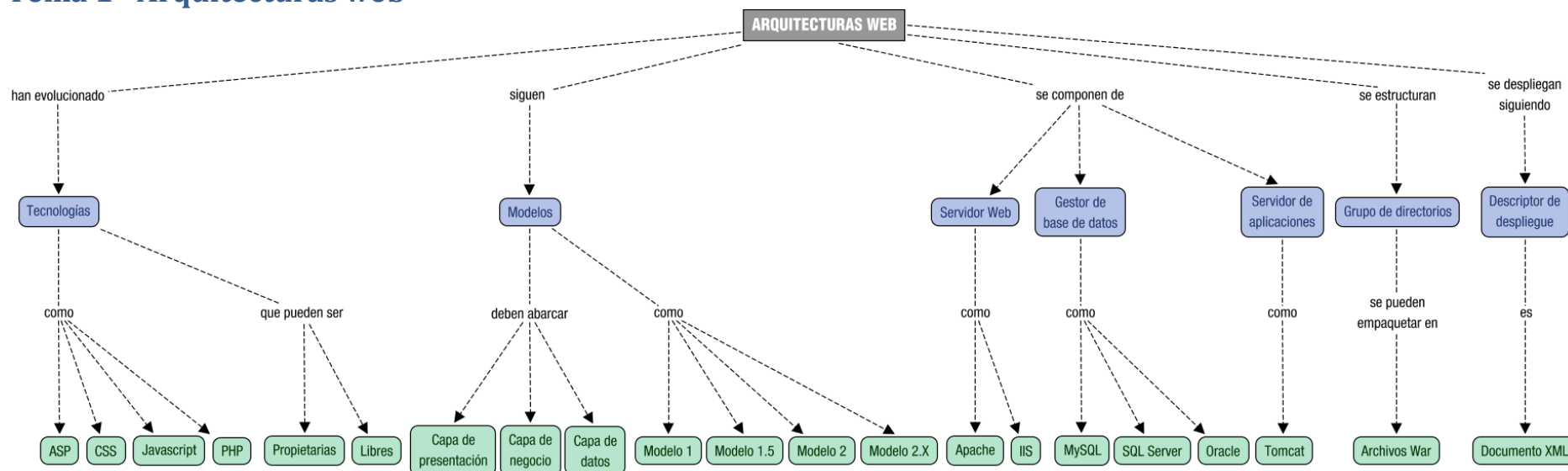


En el flujo de trabajo que hemos dibujado hasta el momento, los desarrolladores no subían directamente cambios al repositorio público, sino que era el responsable del mismo quien aceptaba los cambios y los incorporaba después de revisarlos. Sin embargo, Git también soporta que los desarrolladores puedan subir sus modificaciones directamente a un repositorio centralizado al más puro estilo CVS o Subversion (eliminando el papel de responsable del repositorio público).

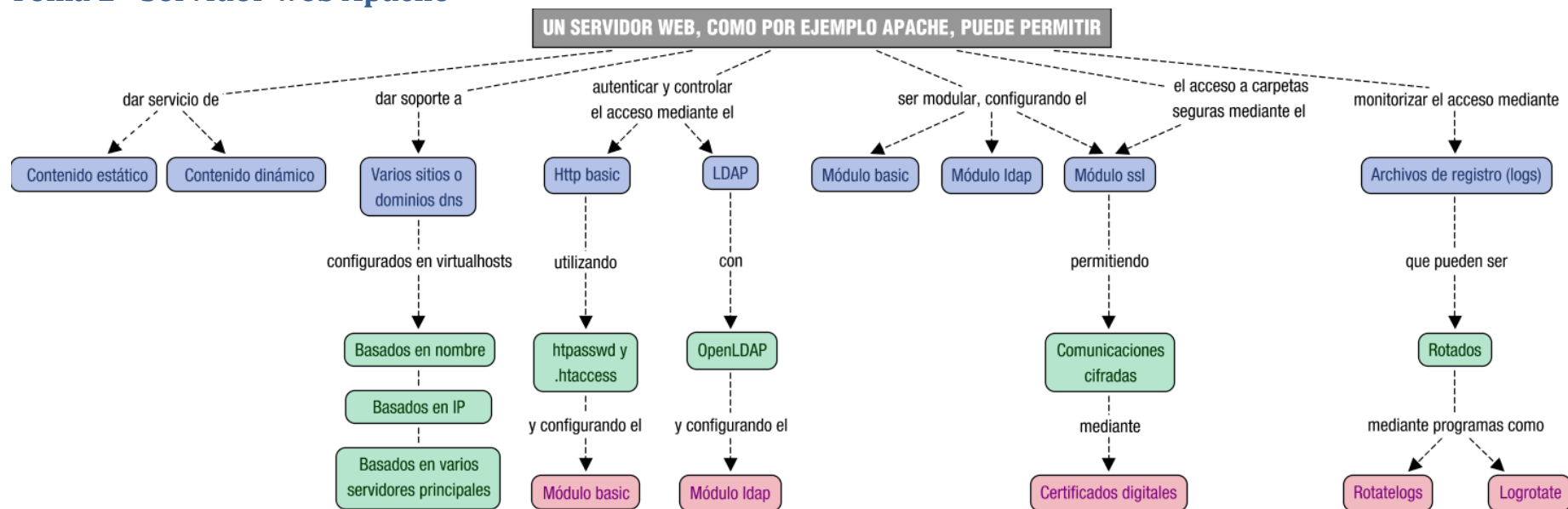
Para que un repositorio público pueda ser utilizado de esta forma, es necesario permitir la ejecución del comando `push`. En primer lugar deberemos crear el repositorio público (tal y como hemos visto en secciones anteriores) y habilitar alguno de los siguientes accesos:

- ✓ Acceso directo por sistema de ficheros con permisos de escritura para el usuario sobre el directorio del repositorio.
- ✓ Acceso remoto vía SSH (consultar `man git-shell`) y permisos de escritura para el usuario sobre el directorio del repositorio.
- ✓ Acceso HTTP con WebDav debidamente configurado.
- ✓ Acceso mediante protocolo Git con el servicio "receive-pack" activado en el git daemon.

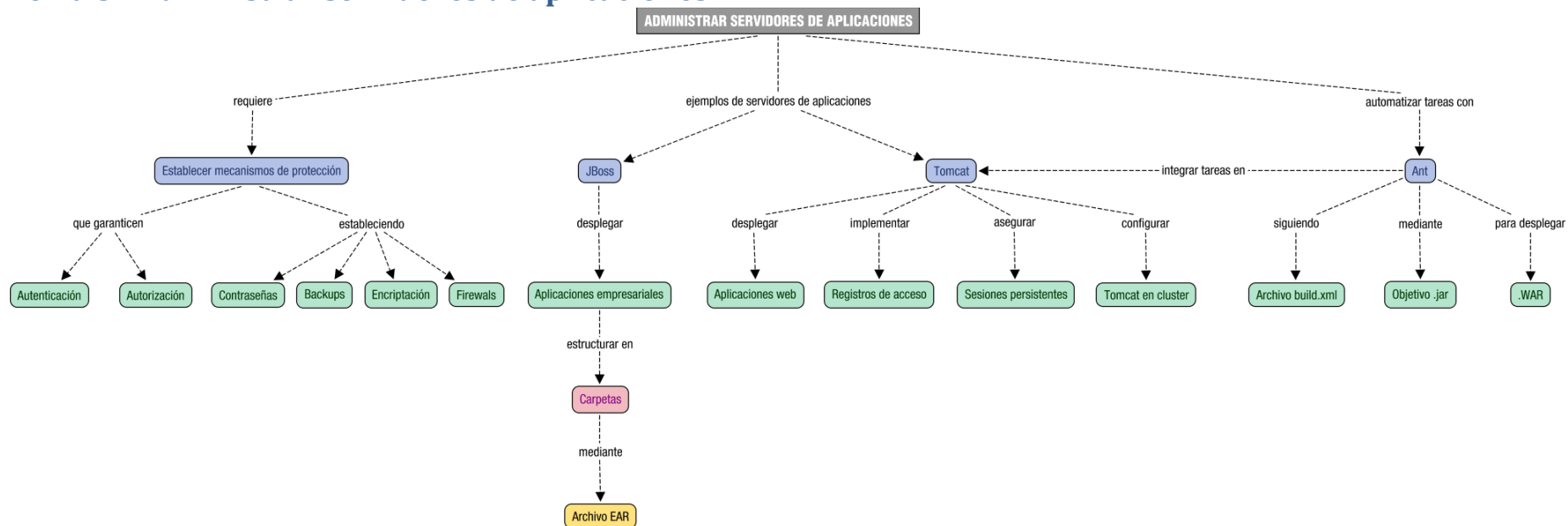
Tema 1 - Arquitecturas web



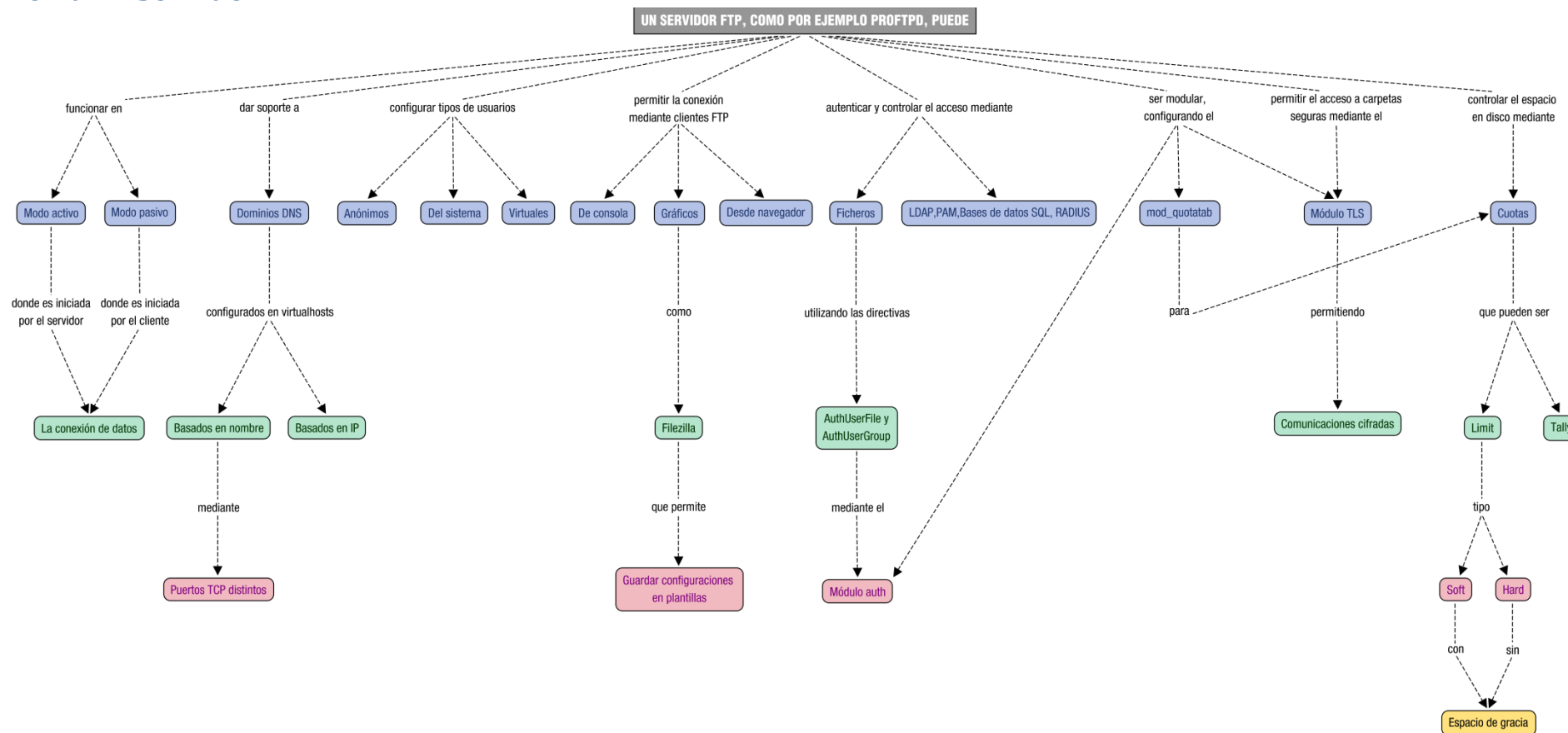
Tema 2 - Servidor web Apache



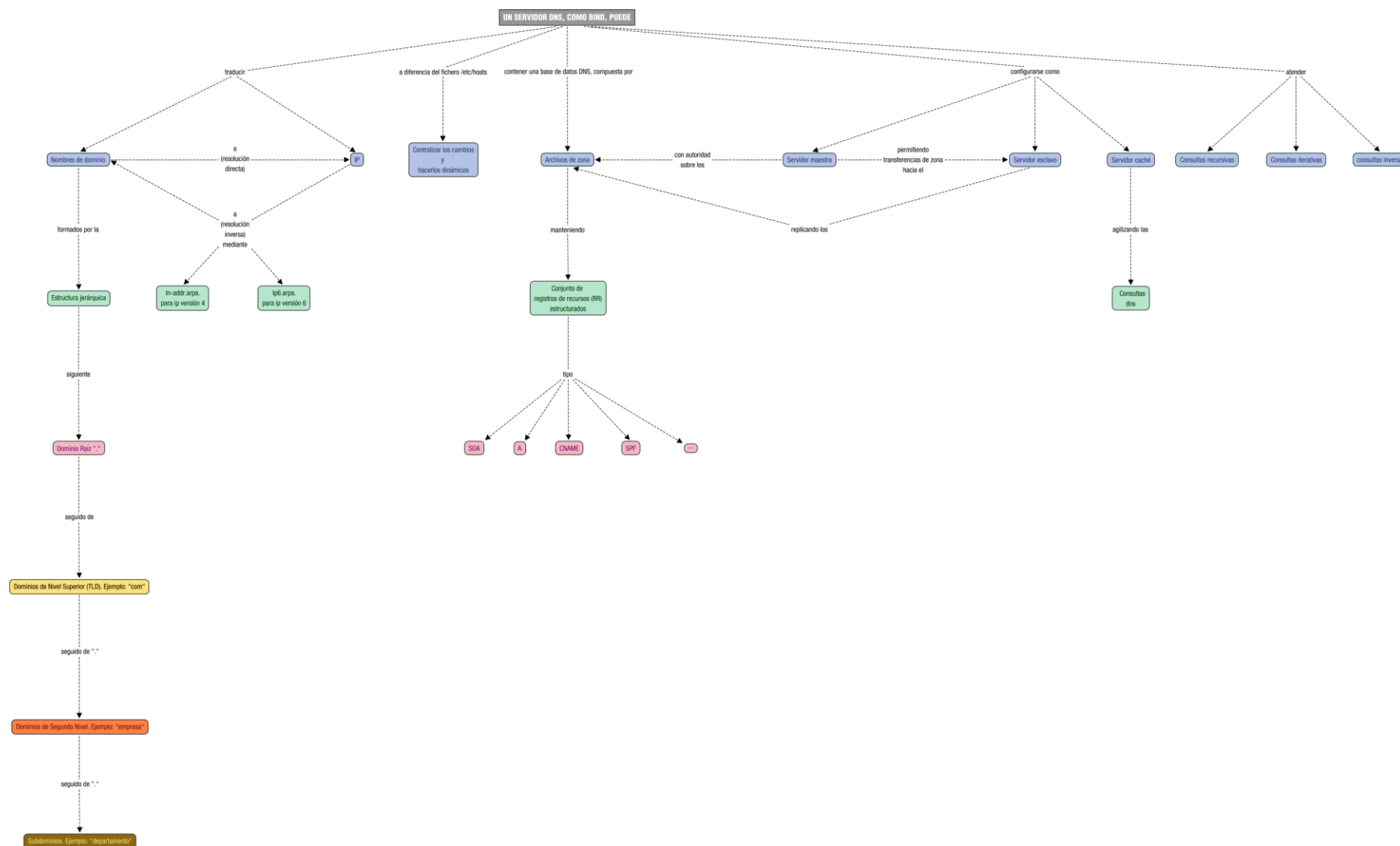
Tema 3 - Administrar servidores de aplicaciones



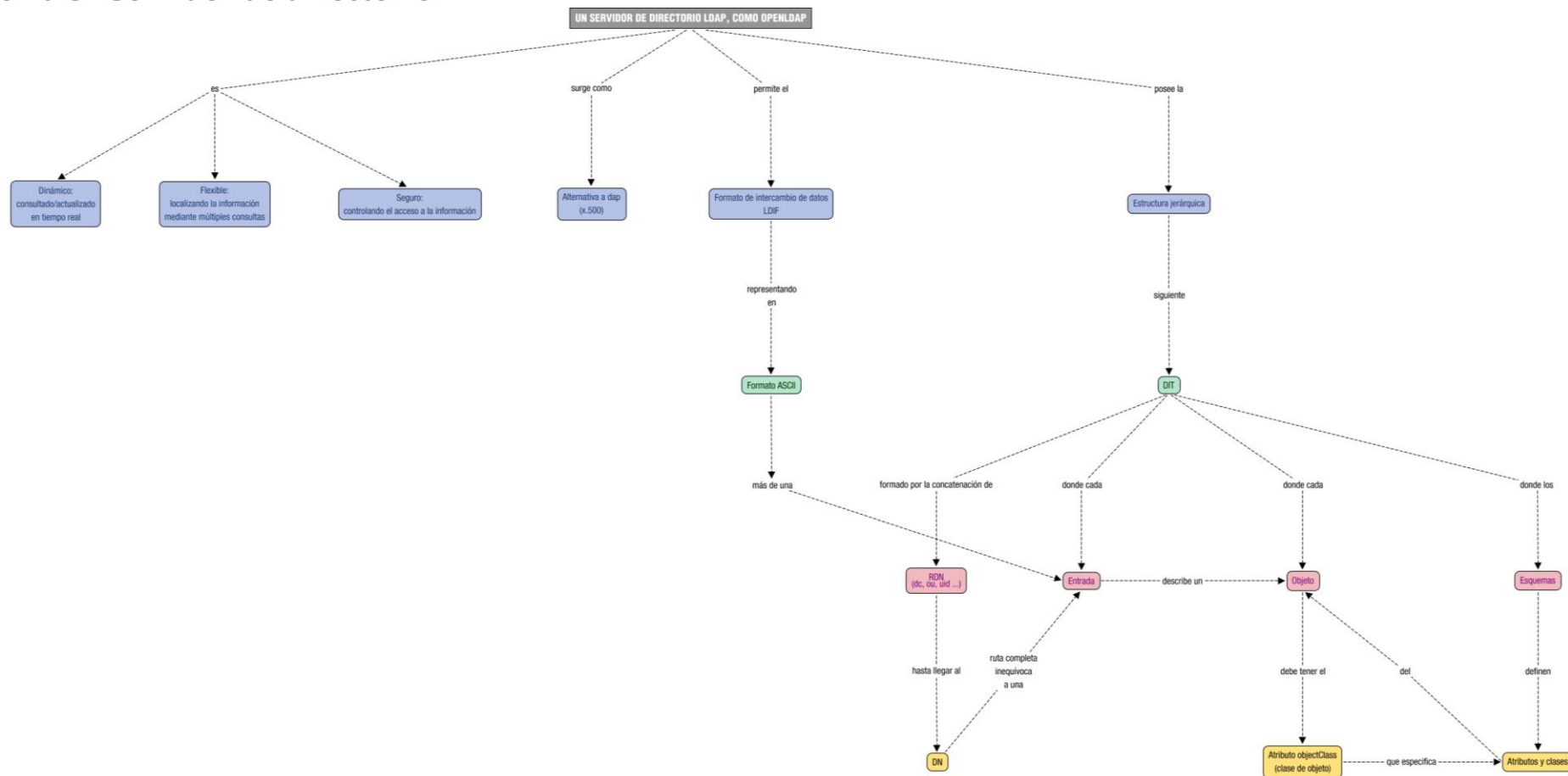
Tema 4 - Servidor FTP



Tema 5 - Servidor DNS (BIND)



Tema 5 - Servidor de directorio LDAP



Tema 6 - Desarrollar aplicaciones web

