Escola Politècnica Superior
d'Enginyeria de Vilanova i la Geltrú
UNIVERSITAT POLITÈCNICA DE CATALUNYA

# PROJECTE FI DE CARRERA

TÍTOL: **Implementación de servicios de Internet sobre un servidor Linux y su aplicación a la docencia de Telemàtica**

**AUTOR: Fº Javier Estrada Fernández**

**TITULACIÓ: Enginyeria Tècnica de Telecomunicaciones Sistemes electrónics**

**DIRECTOR: Immaculada Ruiz Vela**

**DEPARTAMENT: Departament d'Enginyeria Telemàtica**

**DATA: 9 de juliol de 2007**

TÍTOL: **Implementación de servicios de Internet sobre un servidor Linux y su aplicación a la docencia de Telemàtica**

**COGNOMS: Estrada Fernández** **NOM: Fº Javier**

**TITULACIÓ: Enginyeria Tècnica de Telecomunicacions**

**ESPECIALITAT: Sistemes Electrónics** **PLA: 1995**

**DIRECTOR: Immaculada Ruiz Vela**

**DEPARTAMENT: Enginyeria Telemàtica**

**QUALIFICACIÓ DEL PFC**

**TRIBUNAL**

**PRESIDENT** **SECRETARI** **VOCAL**

**DATA DE LECTURA:** 9 de juliol de 2007

**Aquest Projecte té en compte aspectes mediambientals: □ Sí X No**

**PROJECTE FI DE CARRERA**

**RESUM (màxim 50 línies)**

El proyecto consiste en la instalación y configuración de un servidor Linux que proporcione los siguientes servicios de Internet: DHCP (*Dynamic Host Configuration Protocol*), SSH (*Secure SHell*), DNS (*Domain Name Server*) y HTTP (*HyperText Transfer Protocol*). Este servidor permitirá, junto con los clientes adecuadamente configurados, preparar los guiones para diversas prácticas orientadas a la asignatura de Telemática.

La instalación de la distribución de Linux (*Ubuntu 6.06*) se realizará en una máquina compartiendo disco duro con otro sistema operativo (*Windows XP*). Dicha configuración implicará el estudio y conocimiento de las diferentes herramientas de particionamiento de discos y la aplicación de técnicas para su coexistencia.

Posteriormente, se instalarán los paquetes necesarios para los diferentes servicios y se comprobará su correcto funcionamiento en una red de área local con acceso a Internet vía ADSL.

Para finalizar, se redactarán los guiones de las prácticas (estudios previos, ejercicios y teoría) para la aplicación de estos servicios en la docencia de la asignatura de Telemática.

**Paraules clau (màxim 10):**



| | | | |
|---|---|---|---|
| Internet | Modelo OSI | Modelo TCP/IP | Linux |
| Cliente | Servidor Web | DHCP | SSH |
| DNS | http | | |

# Agradecimientos

Mi primer agradecimiento va destinado a la Tutora del Proyecto, Imma Ruiz, por su ayuda, predisposición y paciencia que ha tenido conmigo durante estos meses de trabajo.

También quería enviar mi más sincero agradecimiento a dos profesionales del diseño que, en primer lugar, son dos grandes amigos, Daniel Ribó y Lourdes Sánchez por su ayuda y consejos en la realización de los gráficos utilizados en el proyecto.

Por último agradecer el apoyo de todos los que comparten mi día a día, empezando por mi padre y mi madre, mi hermano Jose, mi cuñada Mari, mis sobrinos y a todos y cada uno de mis amigos que han estado a mi lado durante toda la carrera.

# Implementación de servicios de Internet sobre un servidor Linux y su aplicación a la docencia de Telemática

## ÍNDICE

# 1. Introducción

## 1. Introducción

Internet ha supuesto una revolución sin precedentes en el mundo de la informática y de las comunicaciones. Los inventos del telégrafo, teléfono, radio y ordenador sentaron las bases para esta integración de capacidades nunca antes vivida.

Internet es a la vez una oportunidad de difusión mundial, un mecanismo de propagación de la información y un medio de colaboración e interacción entre los individuos y sus ordenadores independientemente de su localización geográfica.

También podemos decir que representa uno de los ejemplos más exitosos de los beneficios de la inversión sostenida y del compromiso de investigación y desarrollo en infraestructuras informáticas. A raíz de la primitiva investigación en conmutación de paquetes, el gobierno, la industria y el mundo académico han sido copartícipes de la evolución y desarrollo de esta nueva y excitante tecnología.

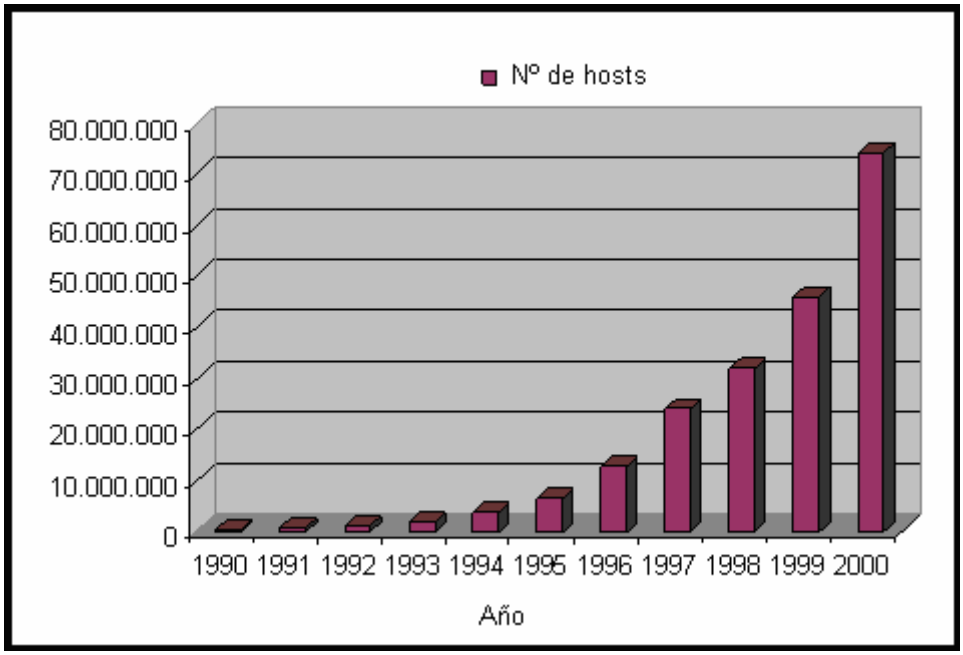


*Figura 1 Crecimiento del número de ordenadores en el mundo en la década de los 90.*

El proyecto que a continuación empieza consiste en la puesta en marcha de diferentes servicios que nos encontramos en cualquier red con conexión a Internet así como la configuración de los mismos para comprobar que cualquiera puede tener en su casa lo necesario para navegar por Internet así como trabajar en red sin necesidad de hacer un desembolso elevado en potentes equipos ni caras licencias que nos permitan utilizar un determinado software. Este es un punto importante en el proyecto ya que los medios físicos a utilizar van a ser dos máquinas que ninguna de ellas tiene menos de

4 años de antigüedad, con un valor en el mercado prácticamente despreciable y un software libre de licencias, es decir, completamente gratuito.

La puesta en marcha de dichos servicios la realizaremos en una máquina Linux que compartirá disco duro con el sistema operativo conocido por todos Windows XP. Linux es gratuito y todo el mundo lo puede descargar de la red sin pagar nada, instalárselo en su PC y convivir (incluso 'vivir' solo) con Windows sin que suponga problema alguno.

La finalidad del proyecto será realizar el montaje y configuración de un servidor Linux con diferentes servicios de Internet configurados para, posteriormente, realizar un guión de prácticas que permita a los futuros Ingenieros experimentar con estos servicios, conocerlos y saber configurarlos así como ser capaces de diagnosticar problemas con las herramientas que disponemos hoy en día (control remoto, analizadores de protocolos de red etc...)

## 1.1 Objetivos

- Conocer el funcionamiento de Internet, su historia, sus fundamentos y los diferentes protocolos que lo constituyen.
- Introducirse en el mundo de los sistemas operativos, conociendo sus características, su funcionamiento y las diferentes opciones que tenemos a la hora de elegir y utilizar uno en nuestro PC.
- Conocer la historia del software libre, el porqué de su creación así como el origen de Linux, desde sus inicios hasta la actualidad. Recoger información y conocer las diferentes distribuciones de Linux para poder elegir una de ellas e instalarla para el entorno de desarrollo del proyecto.
- Obtener los conocimientos necesarios para ser capaz de instalar un Sistema Operativo Linux, compartiendo disco duro con otro sistema operativo (por ejemplo Windows XP). Esto incluye conocer que tipos de archivos soporta cada uno, así como los conocimientos necesarios para que puedan coexistir sin problemas, compartiendo unidades, en una misma unidad física de disco duro perteneciente a una misma máquina.
- Instalar y configurar los servicios DHCP, SSH, DNS y Servidor Web en la máquina Linux para que trabajen en la red de área local.
- Redactar los estudios previos y ejercicios para los guiones de prácticas de la asignatura de TELEMÁTICA.

# 2. Introducción a los servicios de Internet

# 2. Introducción a los servicios de Internet

## 2.1 Un poco de historia

**Cronología de la aparición de Internet.**

La primera descripción documentada acerca de las interacciones sociales que podrían ser propiciadas a través del networking (trabajo en red) está contenida en una serie de memorandums escritos por J.C.R. Licklider, del Massachusetts Institute of Technology, en Agosto de 1962, en los cuales Licklider discute sobre su concepto de *Galactic Network* (Red Galáctica). El concibió una red interconectada globalmente a través de la que cada uno pudiera acceder desde cualquier lugar a datos y programas. En esencia, el concepto era muy parecido a la Internet actual. Licklider fue el principal responsable del programa de investigación en ordenadores de la DARPA desde Octubre de 1962. Mientras trabajó en DARPA convenció a sus sucesores Ivan Sutherland, Bob Taylor, y el investigador del MIT Lawrence G. Roberts de la importancia del concepto de trabajo en red.

En Julio de 1961 Leonard Kleinrock publicó desde el MIT el primer documento sobre la teoría de conmutación de paquetes. Kleinrock convenció a Roberts de la factibilidad teórica de las comunicaciones vía paquetes en lugar de circuitos, lo cual resultó ser un gran avance en el camino hacia el trabajo informático en red. El otro paso fundamental fue hacer dialogar a los ordenadores entre sí. Para explorar este terreno, en 1965, Roberts conectó un ordenador TX2 en Massachusetts con un Q-32 en California a través de una línea telefónica conmutada de baja velocidad, creando así la primera (aunque reducida) red de ordenadores de área amplia jamás construida. El resultado del experimento fue la constatación de que los ordenadores de tiempo compartido podían trabajar juntos correctamente, ejecutando programas y recuperando datos a discreción en la máquina remota, pero que el sistema telefónico de conmutación de circuitos era totalmente inadecuado para esta labor. La convicción de Kleinrock acerca de la necesidad de la conmutación de paquetes quedó pues confirmada.

A finales de 1966 Roberts se trasladó a la DARPA a desarrollar el concepto de red de ordenadores y rápidamente confeccionó su plan para ARPANET, publicándolo en 1967. En la conferencia en la que presentó el documento se exponía también un trabajo sobre el concepto de red de paquetes a cargo de Donald Davies y Roger Scantlebury del NPL. Scantlebury le habló a Roberts sobre su trabajo en el NPL así como sobre el de Paul Baran y otros en RAND. El grupo RAND había escrito un documento sobre redes de conmutación de paquetes para comunicación vocal segura en el ámbito militar, en 1964. Ocurrió que los trabajos del MIT (1961-67), RAND (1962-65) y NPL (1964-67) habían discurrido en paralelo sin que los investigadores hubieran conocido el trabajo de los demás. La palabra packet (paquete) fue adoptada a partir del trabajo del NPL y la velocidad de la línea propuesta para ser usada en el diseño de ARPANET fue aumentada desde 2,4 Kbps hasta 50 Kbps.

En Agosto de 1968, después de que Roberts y la comunidad de la DARPA hubieran refinado la estructura global y las especificaciones de ARPANET, DARPA lanzó un RFQ para el desarrollo de uno de sus componentes clave: los conmutadores de paquetes llamados *interface message processors* (IMPs, procesadores de mensajes de interfaz). El RFQ fue ganado en Diciembre de 1968 por un grupo encabezado por Frank Heart, de Bolt Beranek y Newman (BBN). Así como el equipo de BBN trabajó en IMPs con Bob Kahn tomando un papel principal en el diseño de la arquitectura de la ARPANET global, la topología de red y el aspecto económico fueron diseñados y optimizados por Roberts trabajando con Howard Frank y su equipo en la Network Analysis Corporation, y el sistema de medida de la red fue preparado por el equipo de Kleinrock de la Universidad de California, en Los Angeles.

A causa del temprano desarrollo de la teoría de conmutación de paquetes de Kleinrock y su énfasis en el análisis, diseño y medición, su *Network Measurement Center* (Centro de Medidas de Red) en la UCLA fue seleccionado para ser el primer nodo de ARPANET. Todo ello ocurrió en Septiembre de 1969, cuando BBN instaló el primer IMP en la UCLA y quedó conectado el primer ordenador *host*. El proyecto de Doug Engelbart denominado *Augmentation of Human Intelect* (Aumento del Intelecto Humano) que incluía NLS, un primitivo sistema hipertexto en el Instituto de Investigación de Standford (SRI) proporcionó un segundo nodo. El SRI patrocinó el *Network Information Center*, liderado por Elizabeth (Jake) Feinler, que desarrolló funciones tales como mantener tablas de nombres de *host* para la traducción de direcciones así como un directorio de RFCs (*Request For Comments*). Un mes más tarde, cuando el SRI fue conectado a ARPANET, el primer mensaje de *host* a *host* fue enviado desde el laboratorio de Leinrock al SRI. Se añadieron dos nodos en la

Universidad de California, Santa Bárbara, y en la Universidad de Utah. Estos dos últimos nodos incorporaron proyectos de visualización de aplicaciones, con Glen Culler y Burton Fried en la UCSB investigando métodos para mostrar funciones matemáticas mediante el uso de "storage displays" (N. del T.: mecanismos que incorporan buffers de monitorización distribuidos en red para facilitar el refresco de la visualización) para tratar con el problema de refrescar sobre la red, y Robert Taylor y Ivan Sutherland en Utah investigando métodos de representación en 3-D a través de la red. Así, a finales de 1969, cuatro ordenadores host fueron conectados conjuntamente a la ARPANET inicial y se hizo realidad una embrionaria Internet. Incluso en esta primitiva etapa, hay que reseñar que la investigación incorporó tanto el trabajo mediante la red ya existente como la mejora de la utilización de dicha red. Esta tradición continúa hasta el día de hoy.

Se siguieron conectando ordenadores rápidamente a la ARPANET durante los años siguientes y el trabajo continuó para completar un protocolo host a host funcionalmente completo, así como software adicional de red. En Diciembre de 1970, el Network Working Group (NWG) liderado por S.Crocker acabó el protocolo host a host inicial para ARPANET, llamado Network Control Protocol (NCP, protocolo de control de red). Cuando en los nodos de ARPANET se completó la implementación del NCP durante el periodo 1971-72, los usuarios de la red pudieron finalmente comenzar a desarrollar aplicaciones.

En Octubre de 1972, Kahn organizó una gran y muy exitosa demostración de ARPANET en la International Computer Communication Conference. Esta fue la primera demostración pública de la nueva tecnología de red. Fue también en 1972 cuando se introdujo la primera aplicación "estrella": el correo electrónico. En Marzo, Ray Tomlinson, de BBN, escribió el software básico de envío-recepción de mensajes de correo electrónico, impulsado por la necesidad que tenían los desarrolladores de ARPANET de un mecanismo sencillo de coordinación. En Julio, Roberts expandió su valor añadido escribiendo el primer programa de utilidad de correo electrónico para relacionar, leer selectivamente, almacenar, reenviar y responder a mensajes. Desde entonces, la aplicación de correo electrónico se convirtió en la mayor de la red durante más de una década. Fue precursora del tipo de actividad que observamos hoy día en la World Wide Web, es decir, del enorme crecimiento de todas las formas de tráfico persona a persona.

## 2.2 El modelo de referencia OSI, su estructura en capas

El desarrollo de redes sucedió con desorden en muchos sentidos. A principios de la década de 1980 se produjo un enorme crecimiento en la cantidad y el tamaño de las redes. A medida que las empresas tomaron conciencia de las ventajas de usar tecnología de networking, las redes se agregaban o expandían a casi la misma velocidad a la que se introducían las nuevas tecnologías de red.

Para mediados de la década de 1980, estas empresas comenzaron a sufrir las consecuencias de la rápida expansión. De la misma forma en que las personas que no hablan un mismo idioma tienen dificultades para comunicarse, las redes que utilizaban diferentes especificaciones e implementaciones tenían dificultades para intercambiar información. El mismo problema surgía con las empresas que desarrollaban tecnologías de networking privadas o propietarias. "Propietario" significa que una sola empresa o un pequeño grupo de empresas controlan todo uso de la tecnología. Las tecnologías de networking que respetaban reglas propietarias en forma estricta no podían comunicarse con tecnologías que usaban reglas propietarias diferentes.

Para afrontar el problema de incompatibilidad de redes, la Organización Internacional para la Estandarización (ISO) investigó modelos de networking como la red de Digital Equipment Corporation (DECnet), la Arquitectura de Sistemas de Red (SNA) y TCP/IP a fin de encontrar un conjunto de reglas aplicables de forma general a todas las redes. Con base en esta investigación, la ISO desarrolló un modelo de red que ayuda a los fabricantes a crear redes que sean compatibles con otras redes.

Siguiendo el esquema de este modelo se crearon numerosos protocolos, por ejemplo X.25, que durante muchos años ocuparon el centro de la escena de las comunicaciones informáticas. El advenimiento de protocolos más flexibles donde las capas no están tan demarcadas y la correspondencia con los niveles no era tan clara puso a este esquema en un segundo plano. Sin embargo sigue siendo muy usado en la enseñanza como una manera de mostrar como puede estructurarse una "pila" de protocolos de comunicaciones.

El modelo en sí mismo no puede ser considerado una arquitectura, ya que no especifica el protocolo que debe ser usado en cada capa, sino que suele hablarse de modelo de referencia. Este modelo está dividido en siete capas organizadas tal como muestra la siguiente figura:


*Figura 2 capas del modelos O.S.I.*

### 2.2.1 Intercambio de datos entre capas. Unidades de datos

El intercambio de información entre dos capas OSI consiste en que cada capa en el sistema fuente agrega información de control a los datos, y cada capa en el sistema de destino analiza y extrae la información de control de los datos como sigue:

Si un ordenador (host A) desea enviar datos a otro (host B), en primer término los datos deben empaquetarse a través de un proceso denominado encapsulamiento, es decir, a medida que los datos se desplazan a través de las capas del modelo OSI, reciben cabeceras, información final y otros tipos de información.

Del mismo modo cuando la información llega al host B, las cabeceras se extraen a través de las sucesivas capas OSI hasta llegar a proporcionar los datos finales al usuario.

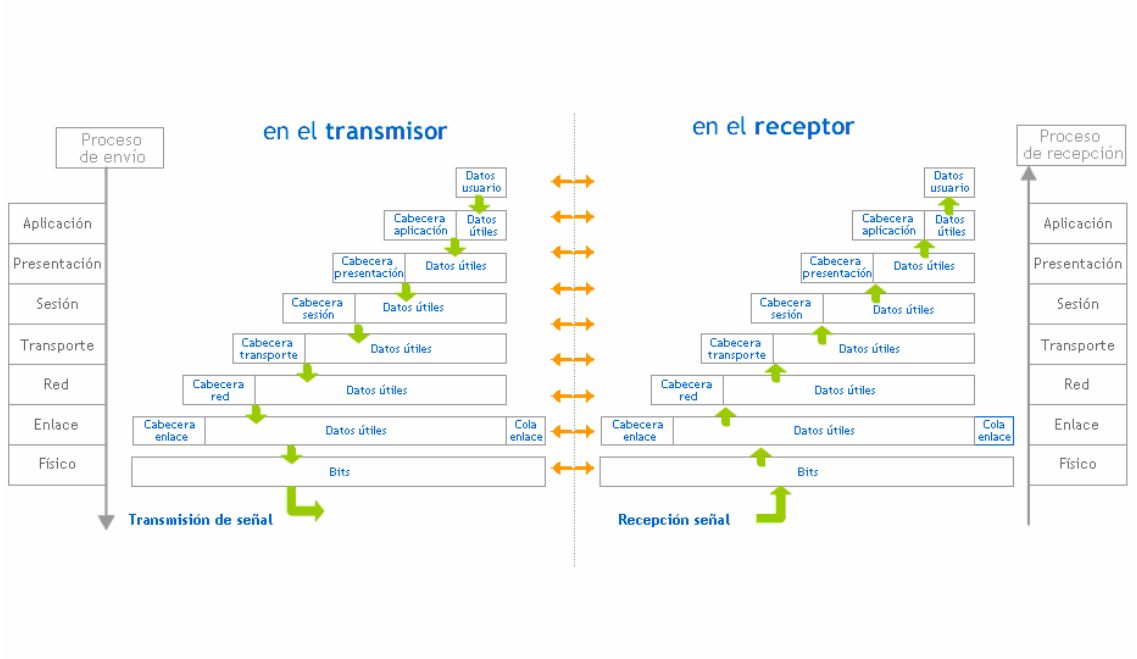


*Figura 3. Encapsulamiento y desencapsulamiento de datos entre capas.*

## 2.2.2 Función de las diferentes capas del modelo OSI

### Nivel Físico

La Capa Física del modelo de referencia OSI es la que se encarga de las conexiones físicas de la computadora hacia la red, tanto en lo que se refiere al medio físico (medios guiados: cable coaxial, cable de par trenzado, fibra óptica y otros tipos de cables; medios no guiado: radio, infrarrojos, microondas, láser y otras redes inalámbricas); características del medio (por ejemplo tipo de cable o calidad del mismo; tipo de conectores normalizados o en su caso tipo de antena; etc.) y la forma en la que se transmite la información (codificación de señal, niveles de tensión/intensidad de corriente eléctrica, modulación, tasa binaria, etc.).

Es la encargada de transmitir los bits de información a través del medio utilizado para la transmisión. Se ocupa de las propiedades físicas y características eléctricas de los diversos componentes; de la velocidad de transmisión, si ésta es uni o bidireccional (simplex, dúplex o full-dúplex). También de aspectos mecánicos de las conexiones y terminales, incluyendo la interpretación de las señales eléctricas/electromagnéticas.

Se encarga de transformar una trama de datos proveniente del nivel de enlace en una señal adecuada al medio físico utilizado en la transmisión. Estos impulsos pueden ser eléctricos (transmisión por cable); o electromagnéticos. Estos últimos, dependiendo de la frecuencia /longitud de onda de la señal pueden ser ópticos, de micro-ondas o de radio. Cuando actúa en modo recepción el trabajo es inverso; se encarga de

transformar la señal transmitida en tramas de datos binarios que serán entregados al nivel de enlace.

*Figura 4 Cable de red RJ45, cable coaxial, fibra óptica.*

**Nivel de Enlace**

Cualquier medio de transmisión debe ser capaz de proporcionar una transmisión sin errores, es decir, un tráfico de datos fiable a través de un enlace físico. Debe crear y reconocer los límites de las tramas, así como resolver los problemas derivados del deterioro, pérdida o duplicidad de las tramas. También puede incluir algún mecanismo de regulación del tráfico que evite la saturación de un receptor que sea más lento que el emisor.

La capa de enlace de datos se ocupa del direccionamiento físico, de la topología de la red, del acceso a la red, de la notificación de errores, de la distribución ordenada de tramas y del control del flujo.

*Figura 5 Switch Cisco Modelo Catalyst 24 puertos 10/100*

**Nivel de Red**

El cometido de la capa de red es hacer que los datos lleguen desde el origen al destino, aún cuando ambos no estén conectados directamente. Es decir que se encarga de encontrar un camino manteniendo una tabla de enrutamiento y atravesando los equipos que sea necesario, para hacer llevar los datos al destino. Los equipos encargados de realizar este encaminamiento se denominan en castellano encaminadores, aunque es más frecuente encontrar el nombre inglés routers y, en ocasiones enrutadores.

Adicionalmente la capa de red debe gestionar la congestión de red, que es el fenómeno que se produce cuando una saturación de un nodo tira abajo toda la red (similar a un atasco en un cruce importante de una ciudad grande).

*Figura 6 Router Cisco modelo 1721*

**Nivel de Transporte**

Su función básica es aceptar los datos enviados por las capas superiores, dividirlos en pequeñas partes si es necesario, y pasarlos a la capa de red. En el caso del modelo OSI, también se asegura que lleguen correctamente al otro lado de la comunicación. Otra característica a destacar es que debe aislar a las capas superiores de las distintas posibles implementaciones de tecnologías de red en las capas inferiores, lo que la convierte en el corazón de la comunicación. En esta capa se proveen servicios de conexión para la capa de sesión que serán utilizados finalmente por los usuarios de la red al enviar y recibir paquetes. Estos servicios estarán asociados al tipo de comunicación empleada, la cual puede ser diferente según el requerimiento que se le haga a la capa de transporte. Por ejemplo, la comunicación puede ser manejada para que los paquetes sean entregados en el orden exacto en que se enviaron, asegurando una comunicación punto a punto libre de errores, o sin tener en cuenta el orden de envío. Una de las dos modalidades debe establecerse antes de comenzar la comunicación para que una sesión determinada envíe paquetes, y ése será el tipo de servicio brindado por la capa de transporte hasta que la sesión finalice. De la explicación del funcionamiento de esta capa se desprende que no está tan encadenada a capas inferiores como en el caso de las capas 1 a 3, sino que el servicio a prestar se determina cada vez que una sesión desea establecer una comunicación. Todo el servicio que presta la capa está gestionado por las cabeceras que agrega al paquete a transmitir.

**Nivel de Sesión**

Esta capa ofrece varios servicios que son cruciales para la comunicación, como son: Control de la sesión a establecer entre el emisor y el receptor (quién transmite, quién escucha y seguimiento de ésta). Control de la concurrencia (que dos comunicaciones a la misma operación crítica no se efectúen al mismo tiempo). Mantener puntos de

verificación (checkpoints), que sirven para que, ante una interrupción de transmisión por cualquier causa, la misma se pueda reanudar desde el último punto de verificación en lugar de repetirla desde el principio. Por lo tanto, el servicio provisto por esta capa es la capacidad de asegurar que, dada una sesión establecida entre dos máquinas, la misma se pueda efectuar para las operaciones definidas de principio a fin, reanudándolas en caso de interrupción. En muchos casos, los servicios de la capa de sesión son parcialmente, o incluso, totalmente prescindibles.

**Nivel de Presentación**

El objetivo de la capa de presentación es encargarse de la representación de la información, de manera que aunque distintos equipos puedan tener diferentes representaciones internas de caracteres (ASCII, Unicode, EBCDIC), números (little-endian tipo Intel, big-endian tipo Motorola), sonido o imágenes, los datos lleguen de manera reconocible.

Esta capa es la primera en trabajar más el contenido de la comunicación que cómo se establece la misma. En ella se tratan aspectos tales como la semántica y la sintaxis de los datos transmitidos, ya que distintas computadoras pueden tener diferentes formas de manejarlas.

Por lo tanto, podemos resumir definiendo a esta capa como la encargada de manejar las estructuras de datos abstractas y realizar las conversiones de representación de datos necesarias para la correcta interpretación de los mismos. Esta capa también permite cifrar los datos y comprimirlos.

**Nivel de Aplicación**

Ofrece a las aplicaciones (de usuario o no) la posibilidad de acceder a los servicios de las demás capas y define los protocolos que utilizan las aplicaciones para intercambiar datos, como correo electrónico (POP y SMTP), gestores de bases de datos y servidor de ficheros (FTP). Hay tantos protocolos como aplicaciones distintas y puesto que continuamente se desarrollan nuevas aplicaciones el número de protocolos crece sin parar.

El usuario normalmente no interactúa directamente con el nivel de aplicación. Suele interactuar con programas que a su vez interactúan con el nivel de aplicación pero ocultando la complejidad subyacente. Así por ejemplo un usuario no manda una petición "HTTP/1.0 GET index.html" para conseguir una página en html, ni lee directamente el código html/xml.

A continuación se muestra un gráfico con diferentes protocolos que corren en los niveles OSI.

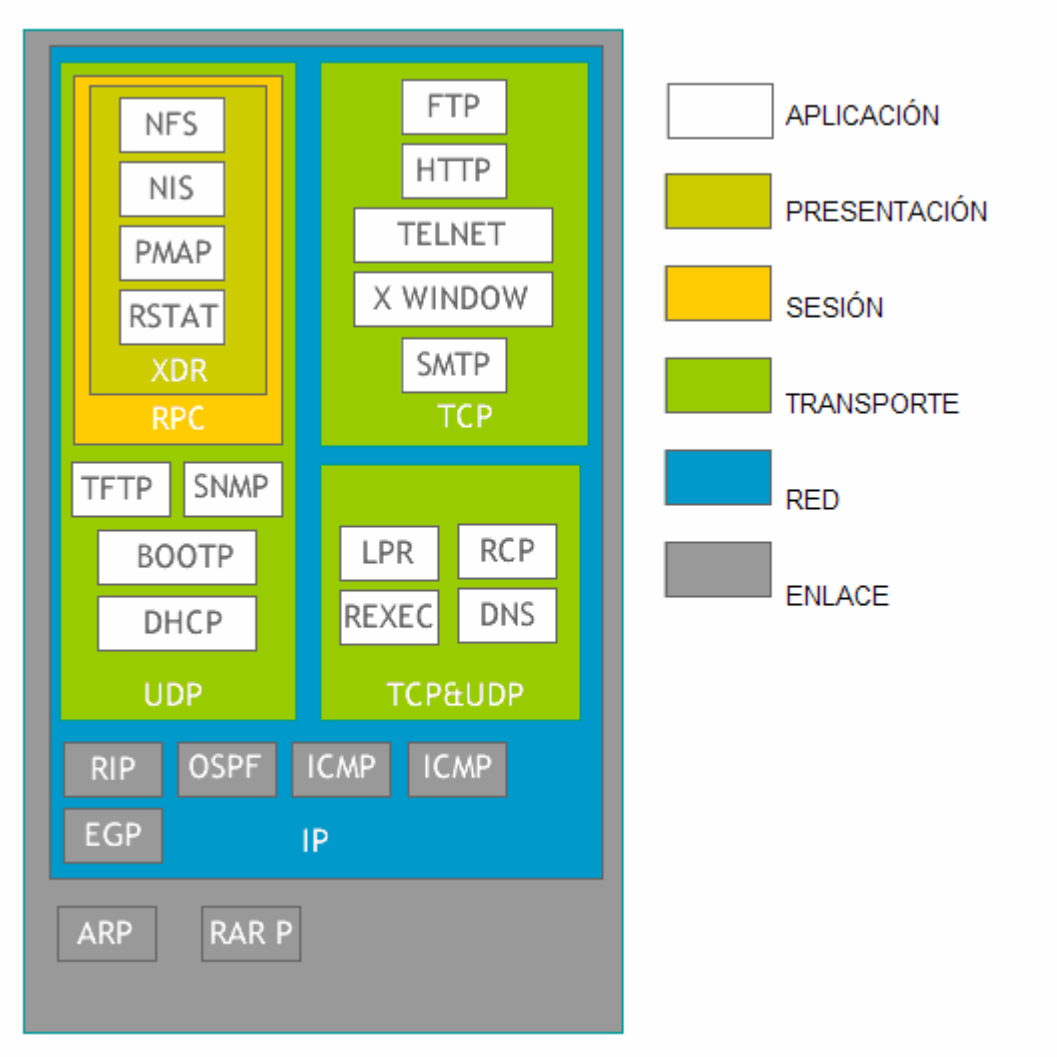


*Figura 7 Ubicación de diferentes protocolos en las capas OSI.*

## 2.3 Comparativa TCP/IP con OSI, los modelos de referencia

Los modelos de referencia OSI y TCP/IP tienen mucho en común. Ambos se basan en el concepto de un gran número de protocolos independientes. También la funcionalidad de las capas es muy similar. Por ejemplo, en ambos modelos las capas por encima de la de transporte, incluida ésta, están ahí para prestar un servicio de transporte de extremo a extremo, independiente de la red, a los procesos que deseen comunicarse. Estas capas forman el proveedor de transporte. También en ambos modelos, las capas encima de la de transporte son usuarios del servicio de transporte orientados a aplicaciones.

A pesar de estas similitudes fundamentales, los dos modelos tienen también muchas diferencias. En esta sección enfocaremos las diferencias clave entre los dos modelos de referencia. Es importante notar que estamos comparando los **modelos de referencia**, no las pilas de protocolos correspondientes que se pueden comprender como una misma teniendo en cuenta que el modelo TCP/IP 'une' algunas de las capas OSI quedando solo 4 de las 7:

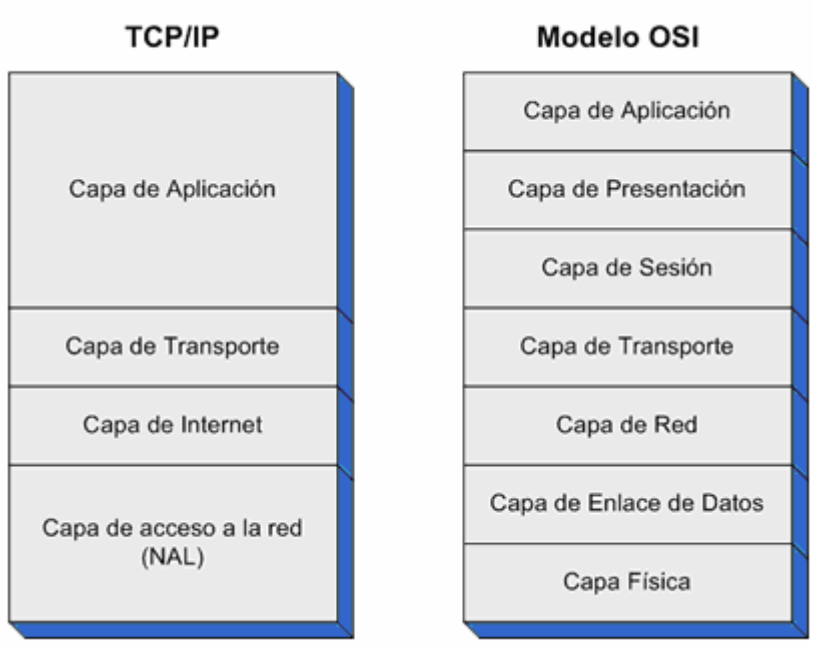


*Figura 8 Comparativa capas de los modelos OSI y TCP/IP*

En el modelo OSI son fundamentales tres conceptos:

1. Servicios.
2. Intertaces.
3. Protocolos.

La contribución más importante del modelo OSI es hacer explícita la distinción entre estos tres conceptos. Cada capa presta algunos **servicios** a la capa que se encuentra sobre ella. La definición de servicio dice lo que la capa hace, no cómo es que las entidades superiores tienen acceso a ella o cómo funciona la capa.

La **interfaz** de una capa les dice a los procesos de arriba cómo acceder a ella; especifica cuáles son los parámetros y qué resultados esperar; no dice nada sobre cómo trabaja la capa por dentro.

Los **protocolos** que se usan en una capa son asunto de la misma capa. Ésta puede usar los protocolos que quiera, siempre que consiga que se realice el trabajo (que provea los servicios que ofrece). La capa también puede cambiar los protocolos a voluntad sin afectar el software de las capas superiores.

Estas ideas ajustan muy bien con las ideas modernas acerca de la programación orientada a objetos. Al igual que una capa, un objeto tiene un conjunto de métodos (operaciones) que los procesos pueden invocar desde fuera del objeto. La semántica de estos métodos define el conjunto de servicios que ofrece el objeto. Los parámetros y resultados de los métodos forman la interfaz del objeto. El código interno del objeto es su protocolo y no está visible ni es de la incumbencia de las entidades externas al objeto.

El modelo TCP/IP originalmente no distinguía de forma clara entre servicio, interfaz y protocolo, aunque se ha tratado de reajustarlo después a fin de hacerlo más parecido a OSI. Por ejemplo, los únicos servicios reales que ofrece la capa de Internet son enviar paquetes IP y recibir paquetes IP.

En el modelo OSI se ocultan mejor los protocolos que en el modelo TCP/IP y se pueden reemplazar con relativa facilidad al cambiar la tecnología. La capacidad de efectuar tales cambios es uno de los principales propósitos de tener protocolos por capas en primer lugar.

El modelo de referencia se desarrolló antes de que se inventaran los protocolos. Este orden significa que el modelo no se orientó hacia un conjunto específico de protocolos, lo cual lo convirtió en algo muy general. El lado malo de este orden es que los diseñadores no tenían mucha experiencia con el asunto y no supieron bien qué funcionalidad poner en cada capa. Por ejemplo, la capa de enlace de datos originalmente tenía que ver sólo con redes de punto a punto. Cuando llegaron las redes de difusión, se tuvo que insertar una nueva subcapa en el modelo. Cuando la gente empezó a constituir redes reales haciendo uso del modelo OSI y de los protocolos existentes, descubrió que no cuadraban con las especificaciones de servicio requeridas, de modo que se tuvieron que injertar en el modelo subcapas de convergencia que permitieran *tapar* las diferencias. Por último, el comité esperaba originalmente que cada país tuviera una red controlada por el gobierno que usara los protocolos OSI, de manera que no se pensó en la interconexión de redes. Las cosas no salieron como se esperaba.

Lo contrario sucedió con TCP/IP: primero llegaron los protocolos y el modelo fue en realidad sólo una descripción de los protocolos existentes. No hubo el problema de

ajustar los protocolos al modelo, se ajustaban a la perfección. El único problema fue que el modelo no se ajustaba a ninguna otra pila de protocolos: en consecuencia, no fue de mucha utilidad para describir otras redes que no fueran del tipo TCP/IP.

La diferencia más obvia entre los dos modelos es la cantidad de capas: el modero OSI tiene siete capas y el TCP/IP cuatro. Ambos tienen capas de red, de transpone y de aplicación, pero las otras capas son diferentes.

Otra diferencia se tiene en el área de la comunicación sin conexión frente a la orientada a la conexión. El modelo OSI apoya la comunicación tanto sin conexión como la orientada a la conexión en la capa de red, pero en la capa de transporte donde es más importante (porque el servicio de transporte es visible a los usuarios) lo hace únicamente con la comunicación orientada a la conexión. El modelo TCP/IP sólo tiene un modo en la capa de red (sin conexión) pero apoya ambos modos en la capa de transporte, con lo que ofrece una alternativa a los usuarios. Esta elección es importante sobre todo para los protocolos simples de petición y respuesta.

# 3. Los sistemas operativos

# 3. Los Sistemas Operativos

## 3.1 Introducción


Un sistema computador está formado por los siguientes elementos:

- La maquinaria (hardware), que hace referencia a la máquina física.
- Los programas (software), que hacen referencia al conjunto de programas que se ejecutan y que permiten sacar provecho de las prestaciones que ofrece el sistema.

Sin programas un ordenador es prácticamente un objeto sin ninguna utilidad. El software permite almacenar información, procesarla y recuperarla.
Actualmente, un sistema computador es un sistema complejo que consta de uno o más procesadores, memoria, reloj, terminales, discos, impresoras, módems, etc.

Los dispositivos que contiene son de muy diversos tipos y tienen un funcionamiento muy variado. Además, en general hay más de un usuario que utiliza a la vez el sistema. Si un usuario quiere utilizar el sistema de manera eficiente y con una cierta protección de su información respecto al resto de usuarios, difícilmente lo podrá conseguir sin ningún tipo de ayuda.

El problema se ha resuelto poniendo por encima de la maquinaria una capa de software con el objetivo de gestionar las diferentes partes del sistema de manera eficiente y a la vez, presentar al usuario una máquina virtual mucho más sencilla de entender y utilizar. Esta capa es la conocida como software del sistema, y la parte más importante es el sistema operativo (S.O.).

En la figura siguiente se muestra un esquema general del sistema computador:

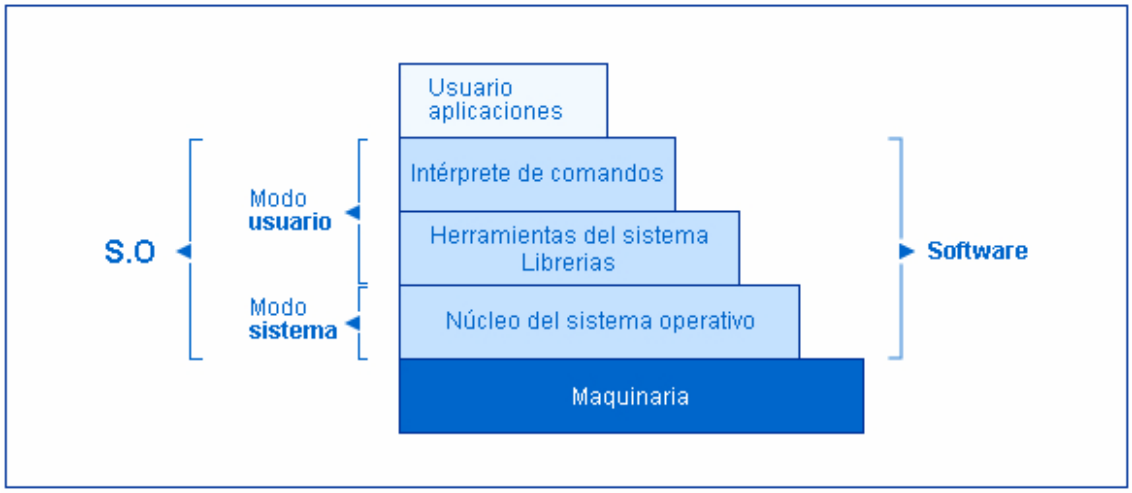


*Figura9 Capas de un sistema computador.*

La capa inferior corresponde a la maquinaria del ordenador. Por encima de ésta encontramos el sistema operativo. Es muy difícil dar una definición exacta del termino sistema operativo, el cual generalmente se define a partir de les funciones que desarrolla.

Básicamente las funciones son las siguientes:

1) Gestión eficiente de los recursos del sistema

   El SO controla el acceso eficiente a los recursos del ordenador: la memoria principal, el tiempo de la unidad central de proceso (UCP) y los dispositivos. Es decir, el SO se encarga principalmente de tareas de protección y de utilització eficiente del sistema.

   En general, en un sistema computador tenemos muchos programas que se ejecutan al mismo tiempo, son programas que pueden pertenecer a uno o a diversos usuarios, los cuales 'compiten' por los diferentes recursos del sistema. El sistema operativo reparte el tiempo de UCP entre los diferentes programas y consigue una ejecución concurrente, proteje el acceso a la memòria y coordina el acceso a los dispositivos compartidos (discos, RAM) y a los no compartidos (por ejemplo la impressora).

2) Presentación a los usuarios de una màquina virtual mucho más sencilla de utilitzar

   El sistema operativo proporciona un entorno de trabajo a los usuarios y a las aplicaciones y permite utilizar el ordenador de forma más fácil e intuitiva. El

sistema operativo presenta al usuario una máquina virtual mucho más fácil de entender y utilizar ya que esconde la complejidad de la maquinaria.

Por encima del nucleo del sistema operativo tenemos el software del sistema, el cual consta del intérprete de comandos (*shell*), compiladores, editores y, en general, programas que facilitan la comunicación entre el sistema operativo y el usuario.

Finalmente, encima del software del sistema está el software de aplicación. Una parte de este software la utiliza el usuario sobretodo para resolver problemas específicos. En éste nivel encontramos hojas de cálculo, procesadores de texto, juegos, etc.

## 3.2 El Software Libre

El Movimiento de software libre comenzó en 1983 cuando Richard Stallman anunció el proyecto GNU. La meta del movimiento es dar libertad a los usuarios de ordenadores reemplazando software con términos de licencia restrictivos por software libre.

La mayoría de los miembros del movimiento de software libre creen que todo el software debería venir acompañado con las libertades explicitadas en la definición de software libre. Muchos sostienen que es inmoral prohibir o impedir a las personas que hagan efectivas esas libertades y que esas libertades son necesarias para crear una sociedad decente donde los usuarios puedan ayudarse mutuamente y tomar el control sobre el uso de un ordenador.

Algunos seguidores del movimiento de software libre no creen que el software propietario sea estrictamente inmoral. Sin embargo, razonan que la libertad es valiosa como una propiedad del software, independiente de su calidad técnica en sentido estricto. Más aún, podrían usar el término "software libre" para distanciarse a sí mismos de afirmaciones tales como que el "software de código abierto" es siempre superior técnicamente al software propietario (lo cual es a menudo falso de forma fácilmente demostrable, al menos a corto plazo). En este sentido, objetan que los defensores del "software de código abierto", concentrándose solamente en méritos técnicos, animan a los usuarios a sacrificar su libertad (y los beneficios a largo plazo que se derivan de su uso) a cambio de ventajas a corto plazo que el software propietario pueda proporcionar.

Los partidarios del código abierto argumentan en favor de las virtudes pragmáticas del software libre (también llamado "software de código abierto") más que de cuestiones de moralidad. Su desacuerdo básico con la Fundación de Software Libre es su

condena genérica del software propietario. Hay muchos programadores que disfrutan apoyando y usando software libre pero se ganan la vida desarrollando software propietario, y no consideran sus acciones inmorales. Las definiciones "oficiales" de software libre y software de código abierto son ligeramente diferentes, siendo la primera considerada más estricta generalmente, mientras que las licencias del software de código abierto son generalmente oscuras, de modo que en la práctica, casi todo el software de código abierto es software libre.

## 3.3. ¿Qué es Linux?

Linux, conocido correctamente como GNU/Linux, es un sistema operativo libre, tipo UNIX, desarrollado originalmente para PC de arquitectura Intel, pero que con el tiempo se ha adaptado a diversas plataformas, incluyendo PowerPC, Macintosh, Amiga, Atari, DEC Alpha, Sun Sparc, ARM, y muchas otras. Linux es libre, es muy barato (incluso gratis en el caso de muchas distribuciones) y si se quiere, se puede elaborar a partir de los 'ingredientes crudos' (ficheros fuente). Linux pretende que el acuerdo con POSIX mantenga la máxima compatibilidad posible con otros sistemas de estilo UNIX. Con miliones de usuarios alrededor del mundo, Linux es probablemente el SO tipo UNIX más popular del mundo.

El kernel o nucleo actual del Linux fue concebido e implementado por primera vez por Linus Torvalds, estudiante del Finnish College, para utilitzarlo en el PC 386 de su casa. Publicó su código en Internet e invitó a la comunidad hacker a utilitzarlo y modificarlo. El nucleo compacto demostró ser estable y rápido, y más tarde fue adoptado por el proyecto GNU como nucleo para su colección de utilidades UNIX.

### 3.3.1 Biografía de Linus Torvalds, padre de Linux

Sus padres tomaron su nombre de Linus Pauling, físico y químico estadounidense galardonado con el Premio Nobel de Química en 1957 por su trabajo en el que describía los enlaces químicos en la Naturaleza y en el 1954 con el Nobel de la Paz por su oposición a las pruebas nucleares.

Torvalds comenzó sus andanzas informáticas a la edad de 11 años cuando su abuelo, un matemático y estadístico de la Universidad, compró uno de los primeros microordenadores Commodore en 1980 y le pidió ayuda para usarlo.

En 1988 Linus es admitido en la Universidad de Helsinki. Ese mismo año Andrew S. Tannenbaum saca a la luz el S.O. Minix. Dos años después, en 1990, Torvalds empieza a aprender el lenguaje de programación C en su universidad.

A finales de los años 80 tomó contacto con los computadores IBM, PC y en 1991 adquirió una computadora con procesador modelo 80386 de Intel.

A la edad de 21 años, con 5 años de experiencia programando (uno en C), ya conocía lo bastante del sistema operativo (S.O.) Minix como para tomarle algunas ideas prestadas y empezar un proyecto personal. Basándose en Design of the Unix Operating System, publicado por Maurice J. Bach en 1986, y modificando gradualmente el Kernel de Minix, crearía una adaptación que ejecutará programas informáticos creados por el proyecto GNU, pero sobre una arquitectura de ordenadores compatibles, IBM/PC.

Este proyecto personal desembocó el 5 de octubre de 1991 con el anuncio de la primera versión de Linux capaz de ejecutar BASH (Bourne Again Shell) y el compilador conocido como GCC (GNU Compiler Collection).

En enero de 1992 se adoptó la Licencia Pública General (GPL) para Linux. Ésta añade libertades de uso a Linux totalmente opuestas a las del software privativo, permitiendo su modificación, redistribución, copia y uso ilimitado. Este modelo de licenciamiento facilita lo que es conocido como el modelo de desarrollo de bazar, que ha dado estabilidad y funcionalidad sin precedentes a éste.

En 1997, Linus Torvalds recibe los premios 1997 Nokia Foundation Award de Nokia y Lifetime Achievement Award at Uniforum Pictures. Ese mismo año finaliza los estudios superiores (1988 - 1997) tras una década como estudiante e investigador en la Universidad de Helsinki, coordinando el desarrollo del núcleo del S.O. desde 1992.

Torvalds trabajó en Transmeta de febrero de 1997 a Junio de 2003 pero actualmente trabaja para el Open Source Development Labs en Beaverton, Oregon. Solo el 2% del código del Linux actual está escrito por él, pero en su persona sigue descansando la paternidad de este núcleo del sistema operativo.

Torvalds posee la marca registrada "Linux" y supervisa el uso o en su defecto el abuso de la marca a través de la organización sin fines de lucro Linux Foundation.

En una entrevista a mediados del año 2005, Linus comentó "Aunque soy el padre de Linux, mis hijos usan Windows".

Ahora Linux es un proyecto que cada vez gana más terreno en el ámbito de la computación y gracias a las políticas del software libre hacen que este sistema operativo este más al alcance de todos para adquirirlo, especialmente en la rama de la

educación, ya que es más viable si no se cuenta con los recursos necesarios para hacerse con una licencia de un software propietario.

Hoy en día existe una gran diversidad de herramientas en software libre dedicadas a la educación, lo que hace de linux una herramienta indispensable, fácil de adquirir y con bajo costo a largo plazo especialmente para escuelas, universidades, incluso grandes empresas, las cuales se benefician y obtienen un gran desempeño, eficiencia y excelentes resultados, además de grandes ahorros.

### 3.3.2 El sistema GNU/Linux

El sistema central del Linux es el nucleo (kernel), el código del sistema operativo que hace funcionar cualquier ordenador. El nucleo está en constante desarrollo y en cualquier momento hay disponible la última versión estable y la última versión experimental. El progréso en el desarrollo és muy rápido, y los nucleos recientes són sorprendentes en todos los aspectos. El diseño del nucleo es modular, de forma que el código real del SO és muy pequeño pero capaz de cargar cuaqluier funcionalidad que se necesite. Debido a esto, el nucleo es pequeño y rápido pero altamente extensible, en comparación con otros sistemas operativos que ralentizan el ordenador y malgastan memoria en cargarlo todo siempre, tanto si se necesita como si no. Hay hasta una útil distribución del Linux que funciona completamente desde un disquet de 1,4 MB.

### ¿Por qué utilitzar Linux?

Hay muchas razones. Es libre, solo se requiere tiempo. Por experimentar el desafio de dominar algo nuevo. Para compartir comunidad con otras personas de intereses parecidos. La oportunidad única de participar en algo que és, de manera discutible, "el projecto de ingeniería de colaboración más grande de la historia de la humanidad". Para explorar una alternativa al, a veces detestado, e incorrectamente llamado el sistema operativo "mas popular" para PC. Los sistemas Linux sobresalen en muchas áreas que abastecen des de las inquietudes del usuario final, como puede ser la estabilidad, la velocidad y la facilidad de uso, hasta temas más serios como el desarrollo y la gestión de redes. El hecho de ser estable, completamente equipado, con aplicaciones libres incluidas en la mayoría de distribuciones. Hoy en dia, Linux ofrece una variedad de paquetes de productividad y ofimáticos comerciales que también pueden importar y exportar ficheros de otras plataformas, incluyendo el Windows y el MacOS.

### Estabilidad

Linux ha sido largamente elogiado por su estabilidad; las máquinas Linux son conocidas por funcionar durante meses, incluso años, sin fallar, congelarse, ni tener que ser reiniciadas nuevamente.

Linux no padeció el efecto 2000 porque almacena los datos de manera diferente a otros ordenadores (su fecha problemática es el 2038, momento en el que una pequeña modificación del núcleo habrá solucionado el problema). Por otro lado, al ser extremadamente seguro comparado con otras plataformas, los virus para Linux son prácticamente in-existentes.

### Velocidad

Las máquinas Linux también son conocidas por ser extremadamente rápidas, porque el sistema operativo es muy eficiente a la hora de gestionar los recursos como la memoria, la potencia de la CPU o espacio en disco. Una buena parte de la Web, está impulsada realmente por antiguos ordenadores 486 que funcionen con Linux y con el servidor web Apache (mas adelante profundizaremos en él); también la NASA, Scandia, Fermilabs y otras han construido superordenadores, muy potentes pero baratos, mediante el agrupamiento de máquinas Linux que funcionan en paralelo.

### Interficie gráfica

Linux tiene como mínimo una docena de interficies gráficas diferentes y altamente configurables (conocidas como gestores de ventanas) que funcionan por encima de XFree86, una implementación libre del sistema X Window. Los gestores de ventanas más populares actualmente son el KDE (K Desktop Environment, Entorno de escritorio K) y el GNOME (GNU Network Object Model Environment, Entorno de modelos de objetos de red de GNU). Los dos ofrecen la funcionalidad "arrastra y suelta" asociada con otros entornos amigables (como el Macintosh), pero son extremadamente flexibles y pueden adquirir diferentes aspectos. Podemos hacer que una máquina Linux con el KDE parezca una máquina Mac, Windows, BeOS, o NextStep, con unos cuantos clics de ratón. Hoy en día, incluso las tareas más complejas como la administración del sistema, la instalación de paquetes, la actualización y la configuración de redes pueden realizarse de manera muy sencilla mediante programas gráficos.

### Desarrollo de software

Los sistemas Linux vienen de serie con compiladores C i C++, un ensamblador, y a menudo también incluyen implementaciones Pascal, FORTRAN y BASIC. Además, hay disponibles lenguajes modernos como el Perl o el Python, o lenguajes clásicos como el LISP, con todas las funcionalidades completamente libres. Adicionalmente, el código fuente para casi cualquier programa Linux está disponible libremente. Esto no significa solamente que los errores se descubren y corrigen casi de inmediato, sino también que el desarrollo de software se realiza a un ritmo mucho mas rápido del que pueden ofrecer las empresas de software comercial extremadamente prósperas. Este fenómeno se llama código abierto (Open Source).

### Trabajo en red

El trabajo en red es connatural a Linux. Se basa en UNIX, en el cual, se desarrolló el trabajo en red. Probablemente todos los protocolos de trabajo en red en uso en Internet son nativos de UNIX y/o Linux, y por tanto se puede esperar que UNIX y Linux trabajen en red mejor que cualquier otra plataforma. Configurar una red sobre una máquina Linux es sorprendentemente sencillo, ya que Linux se ocupa de la mayor parte del trabajo; solo hay que darle la dirección correcta. El Linux está hecho para trabajar en red. Una gran parte de la Web funciona en máquinas Linux, especialmente a causa del servidor web Apache, que derrotó totalmente a sus competidores comerciales, probando así la eficacia y la viabilidad del concepto de Código Abierto.

### Productividad

La disponibilidad del software de productividad ha explotado en los últimos años y muchas empresas han estado produciendo software excelente para la plataforma Linux. El Netscape Navigator y el Communicator están disponibles gratuitamente (con algunas restricciones de licencia) como también el Word Perfect y un montón de aplicaciones, que normalmente vienen como estándar en las distribuciones Linux. Normalmente los paquetes de productividad de Linux pueden leer y escribir ficheros de paquetes de productividad de otras plataformas; Linux se ha esforzado siempre en conseguir la máxima compatibilidad. Linux no tiene ningún problema en coexistir en la misma máquina con otros sistemas operativos.

## 3.4 Distribuciones Linux. La elección

### 3.4.1 ¿Qué es una distribución Linux?

Linux es un sistema de libre distribución, no hay que pagar una licencia para adquirirlo, por lo tanto podemos encontrar todo lo necesario (ficheros, programas…) para su funcionamiento en multitud de servidores conectados a Internet. Reunir todos estos ficheros y programas así como instalarlos en nuestro sistema y configurarlo, podría resultar un trabajo complicado y solo apto para usuarios con cierta experiencia. Esta es la gran razón por lo que aparecieron lo que conocemos como distribuciones de Linux.

Son empresas y organizaciones que se han encargado de hacer este trabajo, ahorrándonoslo a los que queremos ser usuarios de este S.O.

Una distribución no es más que una recopilación de programas y ficheros organizados y preparados para su instalación. Se pueden obtener a través de Internet, por ejemplo vía FTP (las propias empresas o distribuidores se encargan de mantener y administrar estos servidores) o comprando los CDs. La única diferencia está en que con la compra de la distribución, compras también el soporte técnico de la misma.

Existen muchas y muy variadas distribuciones creadas por diferentes empresas y organizaciones.

La desconcertante elección entre un número siempre creciente de distribuciones de Linux puede crear confusión entre aquellos nuevos en Linux. Listamos a continuación las distribuciones que generalmente se consideran las más extendidas entre los usuarios de Linux de todo el mundo. Hay otras muchas que probar pero como norma general, las presentes son populares y tiene foros y listas de correo muy activas donde preguntar las dudas que nos puedan surgir.

### 3.4.2 Listado distribuciones más populares

- Mandrake
- Red Hat
- Gentoo
- debian
- Suse
- Ubuntu
- Knnopix
- Slackware Linux
- Lycoris
- Xandros
- Lindows


#### Mandrake

Mandrake Linux, creada por Gaël Duval, es una distribución que ha experimentado un enorme aumento de popularidad desde su primera versión de julio de 1998. Los desarrolladores partieron de la distribución de Red Hat, cambiaron el entorno de escritorio predeterminado por KDE, y añadieron un instalador fácil de usar rompiendo el mito de que Linux es difícil de instalar. Las herramientas de detección de hardware de Mandrake y sus programas para el particionamiento de discos son consideradas por muchos como las mejores de la industria, y muchos usuarios se encontraron usando Mandrake allí donde otras distribuciones no habían conseguido entregar la facilidad de uso necesaria. Desde entonces Mandrake Linux ha madurado y se ha convertido en una distribución popular entre los nuevos usuarios de Linux y aquellos hogares que buscan un sistema operativo alternativo.

El desarrollo de Mandrake es completamente abierto y transparente, con paquetes nuevos que se añaden al directorio llamado "cooker" a diario. Cuando una nueva versión entra en fase beta, la primera beta se crea a partir de los paquetes que se encuentran en "cooker" en ese momento. El proceso de pruebas de la beta solía ser corto e intensivo, pero desde la versión 9.0 ha pasado a ser más largo y exigente. Las listas de correo sobre la versión beta suelen estar saturadas, pero sigue siendo posible recibir una respuesta rápida sobre cualquier fallo o duda que envíes. Como resultado de este tipo de desarrollo se obtiene una distribución puntera y altamente actualizada. Como contrapartida, los usuarios pueden encontrarse con más fallos que en otras distribuciones.

- Ventajas: Amigable para el usuario, herramientas de configuración gráfica, enorme soporte de la comunidad, posibilidad de cambiar el tamaño de particiones NTFS.
- Inconvenientes: Algunas versiones contienen fallos, la compañía está pasando por problemas financieros.
- Sistema de paquetes: RPM
- Descarga Gratuita: Si

### Red Hat

Para muchos el nombre de Red Hat equivale a Linux, ya que probablemente se trata de la compañía de linux más popular del mundo. Fundada en 1995 por Bob Young y Marc Ewing, Red Hat Inc solo ha mostrado beneficios recientemente gracias a otros servicios en lugar de a la distribución en si. Aun y así, Red Hat es la primera elección para muchos profesionales y seguirá siendo un peso pesado durante mucho tiempo.

Su curiosa mezcla de conservadurismo y paquetes punteros mezclados sobre muchas aplicaciones desarrolladas en casa. Los paquetes no son los más actuales, una vez se anuncia una nueva versión beta, las versiones de los paquetes se mantienen, excepto para actualizaciones de seguridad. Como resultado se obtiene una distribución bien probada y estable. El programa de betas y las facilidades para enviar fallos está abierta al público y hay un gran espíritu en las listas de correo públicas.

Red Hat Linux se ha convertido en la distribución linux dominante en servidores en todo el mundo. Otra de las razones del éxito de Red Hat es la gran variedad de servicios populares que ofrece la compañía. Los paquetes de software son fácilmente actualizables usando la Red Hat Network, un repositorio oficial de software e información. Una larga lista de servicios de soporte son accesibles en la compañía y, aunque no siempre baratos, se tiene virtualmente asegurado un excelente soporte de personal altamente cualificado. Todos estos factores han contribuido a que Red Hat sea una marca reconocida en el mundo de la industria de las TI.

- Ventajas: Ampliamente usada, excelente soporte de la comunidad, muchas innovaciones.
- Inconvenientes: Limitada vida útil de la edición gratuita, soporte multimedia pobre.
- Sistema de paquetes: RPM
- Descarga Gratuita: Si

## Gentoo

Gentoo Linux fue creada por Daniel Robbins, un conocido desarrollador de Stampede Linux y FreeBSD. Fue el contacto del autor con FreeBSD y su función de autobuild llamada "ports" lo que le inspiró a incorporar los "ports" en Gentoo bajo el nombre de "portage". La primera versión estable de Gentoo fue anunciada en Marzo del 2002.

Gentoo Linux es una distribución basada en código fuente. Mientras que los sistemas de instalación proveen de varios niveles de paquetes pre-compilados, para obtener un sistema Linux básico funcionando, el objetivo de Gentoo es compilar todos los paquetes de código en la máquina del usuario. La principal ventaja de esto es que todo el software se encuentra altamente optimizado para la arquitectura de tu computadora.

También, actualizar el software instalado a una nueva versión es tan fácil como teclear un comando, y los paquetes, mantenidos en un repositorio central, se mantienen bastante actualizados. La parte mala sería que instalar Gentoo y convertirla en una distribución completa, con los últimos entornos gráficos, multimedia y de desarrollo es un trabajo largo y tedioso, puede costar varios días incluso en una máquina rápida.

- Ventajas: Fácil instalación de paquetes de software individuales, altamente actualizada. Puedes crear tu propia distribución atendiendo a las necesidades del usuario.
- Inconvenientes: Instalación larga y tediosa, ocasionalmente inestable y con riesgos de romperse, no aconsejada para servidores con funciones críticas.
- Sistema de paquetes: SRC
- Descarga gratuita: Si

## Debian

Debian GNU/Linux inició su andadura de la mano de Ian Murdock en 1993. Debian es un proyecto totalmente no-comercial; posiblemente el más puro de los ideales que iniciaron el movimiento del software libre. Cientos de desarrolladores voluntarios de alrededor del mundo contribuyen al proyecto, que es bien dirigido y estricto, asegurando la calidad de una distribución conocida como Debian. En cualquier momento del proceso de desarrollo existen tres ramas en el directorio principal: "estable", "en pruebas" e "inestable" (también conocida como "sid").

Cuando aparece una nueva versión de un paquete, se sitúa en la rama inestable para las primeras pruebas, si las pasa, el paquete se mueve a la rama de pruebas, donde se realiza un riguroso proceso de pruebas que dura muchos meses. Esta rama solo es declarada estable tras una muy intensa fase de pruebas.

Como resultado de esto, la distribución es posiblemente la más estable y confiable, aunque no la más actualizada. Mientras que la rama estable es perfecta para servidores con funciones críticas, muchos usuarios prefieren usar las ramas de pruebas o inestable, más actualizadas, en sus ordenadores personales. Debian es también famosa por su reputación de ser difícil de instalar, a menos que el usuario tenga un profundo conocimiento del hardware de la computadora. Compensando este fallo está "apt-get" un instalador de paquetes Debian. Tan pronto como Debian está en funcionamiento, todas las actualizaciones de cualquier tipo pueden realizarse mediante la herramienta apt-get.

- Ventajas: 100% libre, web y recursos de la comunidad excelentes, bien probada, instalación de software sencillísima usando apt-get.
- Inconvenientes: Instalador arcaico, la versión estable no está actualizada.
- Sistema de paquetes: DEB
- Descarga gratuita: Si

### SuSE

SuSE es otra compañía orientada a los escritorios, aunque variedad de otros productos para empresas están disponibles. La distribución ha recibido buenas críticas por su instalador y la herramienta de configuración YaST, evolucionada por los desarrolladores de la propia SuSE. La documentación que viene con las versiones comerciales, ha sido repetidas veces evaluada como la más completa, útil y usable con diferencia a la de sus competidores. SuSE Linux 7.3 recibió el premio "Producto del año 2001" que entrega el Linux Journal. La distribución tiene un gran porcentaje de mercado en Europa y América del norte, pero no se vende en Asia y otras partes del mundo .El desarrollo de SuSE se realiza completamente a puerta cerrada, y no se lanzan betas públicas para probar. Siguen la política de no permitir descargar el software hasta tiempo después de que salgan a la venta las versiones comerciales. A pesar de todo, SuSE no entrega imágenes ISO de fácil instalación de su distribución, usando el software empaquetado para la gran mayoría de su base de usuarios.

- Ventajas: Atención profesional en cada detalle, herramienta de configuración de fácil uso (YaST).
- Inconvenientes: Solo disponible en algunas partes del mundo en las tiendas de software o mediante instalación FTP, incluye componentes propietarios, que no permiten su redistribución.
- Sistema de paquetes: RPM
- Descarga gratuita: SuSE no proporciona imágenes ISO para descarga, no obstante la versión Profesional de su distribución es accesible para la instalación FTP normalmente 1 o 2 meses más tarde de la versión oficial. La instalación mediante FTP no es difícil, pero requiere una buena conexión.

### Ubuntu

Ubuntu es una distribución de Linux de tipo escritorio, basada en Debian. El proyecto se encuentra patrocinado por Canonical Ltda. Económicamente se sostiene con aportaciones de la misma empresa que posee por dueño al sudafricano Mark Shuttleworth.

Ubuntu debe su nombre al movimiento homónimo encabezado por el obispo Desmond Tutu, el cual ganó el Premio Nobel de la Paz en 1984 por su lucha en contra del Apartheid en Sudáfrica. Mark Shuttleworth, el mecenas del proyecto, es sudafricano y por lo tanto se encontraba muy familiarizado con la corriente. Tras ver similitudes entre los ideales de los Proyectos GNU, Debian y en general con el movimiento de software libre, decidió aprovechar la ocasión para difundir los ideales de ubuntu.

Ubuntu esta basado en la distribución Debian (En la versión "en pruebas"), pero a diferencia de esta, se saca una versión cada seis meses, aproximadamente, y se mantiene actualizada durante 18 meses después de su lanzamiento. Otra diferencia es el empleo del escritorio Gnome como escritorio principal. No obstante, esta distribución pretende no romper la compatibilidad con debian, de modo que puedan intercambiarse paquetes sin problemas.

Por todo ello, esto hace una distribución muy orientada al escritorio, pero con bastante estabilidad. Fundamentalmente comparte las ventajas de debian (exceptuando que tiene ligeramente menos paquetes, y que estos no están tan probados), añadiéndole el hecho de tener una distribución bastante actualizada.

- Ventajas: 100% libre, web y recursos de la comunidad excelentes, instalación de software sencillísima usando apt-get.

- Inconvenientes: La versión estable no está actualizada.
- Sistema de paquetes: DEB
- Descarga gratuita: Si

### Knoppix

Desarrollada por Klaus Knopper en Alemania. Esta distribución basada en Debian tiene como una de sus mejores características su herramienta de detección de hardware. Su arranque automático, gran cantidad de software, su sistema de descompresión y la posibilidad de instalarlo al disco duro han convertido a knoppix en una herramienta indispensable. Puede ser usada como un disco de rescate, una herramienta para enseñar linux para aquellos que no lo han visto o una herramienta para probar una nueva computadora antes de comprarla. También puede ser usada como una completa distribución linux para el uso diario.

Frecuentemente se lanzan nuevas versiones de Knoppix, aproximadamente, una nueva versión cada 1 o 2 semanas. Las actualizaciones incluyen parches de fallos así como el último software de la rama inestable de Debian.

- Ventajas: Excelente autodetección de hardware, funciona directamente del CD sin instalación en el disco duro, puede ser usada como herramienta de recuperación.
- Inconvenientes: Bajo rendimiento y velocidad si se utiliza directamente del CD.
- Sistema de paquetes: DEB
- Descarga gratuita: Si

### Slackware

Creada por Patrick Volkerding en 1992, Slackware Linux es la distribución más antigua que sobrevive hoy en día. No ofrece extras vistosos, y se mantiene con un instalador basado en texto, y sin herramientas de configuración gráfica. Mientras otras distribuciones intentan desarrollar interficies fáciles de usar para muchas utilidades comunes, Slackware no ofrece nada amistoso, y toda la configuración se realiza mediante los archivos de configuración. Es extremadamente estable y segura, muy recomendada para servidores.

- Ventajas: Alta estabilidad y ausencia de fallos, sigue fielmente los principios de UNIX.

- Inconvenientes: Toda la configuración se realiza mediante la edición de ficheros de texto, autodetección de hardware limitada.
- Sistema de paquetes: TGZ
- Descarga gratuita: Si

### Lycoris

Lycoris, conocida también como Redmond Linux, fue fundada por Joseph Cheek, CTO de la compañía, y trabajador de Linuxcare y Microsoft. El objetivo principal era crear una instalación fácil de usar para realizar la transición entre Windows y Linux lo más fácilmente posible. Para realizar esto, Lycoris Desktop/LX tiene cientos de modificaciones, incluyendo un clon de Mi PC y del Entorno de Red, así como un atractivo tema que se parece mucho a Windows XP. Esta distribución se basa actualmente en Caldera OpenLinux.

El número de paquetes de Lycoris Desktop/LX es bastante reducido, normalmente una aplicación por tarea y el Panel de Control para las tareas comunes de administración. Se le ha criticado que algunos paquetes están obsoletos, a lo que sus desarrolladores responden que su objetivo es hacer una distribución fácil de usar.

- Ventajas: Amigable para principiantes, diseñada para parecerse a Windows.
- Inconvenientes: Contiene paquetes obsoletos, requiere una licencia para uso comercial.
- Sistema de paquetes: RPM
- Descarga gratuita: Si

### Xandros

Xandros nació de las cenizas de Corel Linux, un fallido intento de crear un Linux para las masas del año 1999, pero abandonado poco tiempo después cuando la compañía cayó en dificultades financieras. Xandros compró la distribución en agosto del 2001 y lanzó su primera y única versión en octubre del 2002. Xandros Desktop es sin lugar a dudas la distribución más sencilla de usar del mercado, y altamente recomendable para los nuevos usuarios de Linux.

- Ventajas: Diseñada para principiantes, funciona al instante, excelente administrados de archivos y otras utilidades.

- Inconvenientes: Incluye software propietario, lo que no permite su redistribución, no hay descarga gratuita.
- Sistema de paquetes: DEB
- Descarga gratuita: No

### Lindows

Lindows.com fue iniciado por Michael Robertson, fundador y CEO de MP3.com, en octubre del 2001, en San Diego, EEUU. El objetivo inicial era desarrollar un SO basado en Linux capaz de utilizar no solo las aplicaciones de Linux, sino también las principales herramientas de Windows tales como MS Office. Este ambicioso objetivo fue abandonado ya que los desarrolladores de LindowsOS parecían haber subestimado el esfuerzo necesario para cumplirlo. La primera versión hecha pública de LindowsOS fue anunciada en Noviembre del 2002 bajo el nombre de LindowsOS 3.0.

Las principales características del producto son una instalación rápida y sencilla, y Click-N-Run, una infraestructura para instalar software del repositorio de la compañía. Las opiniones sobre el producto son para todos los gustos, parece ser que Click-N-Run todavía necesita mucho trabajo para pulir sus fallos. La enorme campaña de marqueting puede resultar agobiante, pero han conseguido que LindowsOS (y Linux) aparezcan en los principales medios y que vengan preinstalados en ordenadores nuevos de tiendas de EEUU y Reino Unido.

### 3.4.3 Comparativa de requisitos básicos para la instalación de una distro

| Distro | CPU | Memoria | RAM | Espacio en | disco duro |
|---|---|---|---|---|---|
| Linux | (procesador) | Mínimo | Recomendado | Mínimo | Recomendado |
| openSUSE 10.1 | Intel Pentium 1-4, Celeron, AMD Duron, Athlon, Semprom u Opteron | 256 MB | 512 MB | 500 MB | 3 GB |
| Mandrila 2006 | Intel Pentium 1-4, Celeron, AMD Duron, Athlon, Semprom, Opteron, K6, Via C3 | 128 MB | 256 MB | 500 MB | 4 GB |
| Fedora Core 5 | Intel Pentium 1-4, Celeron, AMD Duron, Athlon, Semprom u Opteron | 64 MB* 194 MB** | 256 MB | 500 MB | 3 GB |
| Debian 3.1 | Intel Pentium 1-4, Celeron, AMD Duron, Athlon, Semprom u Opteron | 32 MB* 194 MB** | 256 MB | 500 MB | 3 GB |
| Debian 3.0 | Intel Pentium 1-4, Celeron, AMD Duron, Athlon, Semprom u Opteron | 16 MB* 64 MB** | 128 MB | 450 MB | 4 GB |
| Ubuntu 6.06 | Intel o AMD con velocidad de 500mhz | 256 MB | 256 MB | 2 GB | 3 GB |
| Ubuntu 5.10 | Intel o AMD con velocidad de 500mhz | 192 MB | 256 MB | 2 GB | 3 GB |
| Slackware | 486 o superior | 16 MB | 32 MB | 100 MB | 3.5 GB |
| Damn Samll 1.0 | 486DX o superior | 16 MB | 64 MB | -- | -- |
| Damn Samll 3.0 | 486DX o superior | 16 MB | 128 MB | -- | -- |
| Slax 5.1 | 486, Pentium o AMD | 36 MB | fluxbox 96 MB KDE 144 MB | -- | -- |
| Puppy 2 | Pentium 166MMX o superior (Intel o AMD) | 128 MB | 128 MB | -- | -- |
| simplyMEPIS 6 | Intel Pentium o AMD Athlon | 128 MB | 512 MB | 2 GB | 3 GB |
| Knoppix 5 | Intel Pentium o AMD Athlon | 32 MB* 96 MB** | 128 MB | -- | -- |

*Figura 10 Comparativa de requisitos.*

# 4. Desarrollo del servidor de servicios de Internet

# 4. Desarrollo del servidor de servicios de Internet

## 4.1 Escenario de trabajo

El esquema de trabajo consistirá en un PC que actuará como cliente con el sistema operativo más popular en el mundo, Windows XP. Por tanto, se trata de una instalación típica a la que luego se le añadirán las herramientas necesaria para realizar las peticiones de servicio, como por ejemplo Putty o CRT para realizar las conexiones SSH. Al otro lado se tendrá un equipo que actuará de Servidor. Éste equipo necesitará una instalación y configuración más compleja que el cliente. Todo irá explicando seguidamente, ya que va ser el encargado de proporcionar los siguientes servicios que se muestran en la siguiente figura. Estos servicios los abastecerá con un sistema operativo Linux, concretamente con la distribución **Ubuntu 6.06 LTS Dapper Drake**

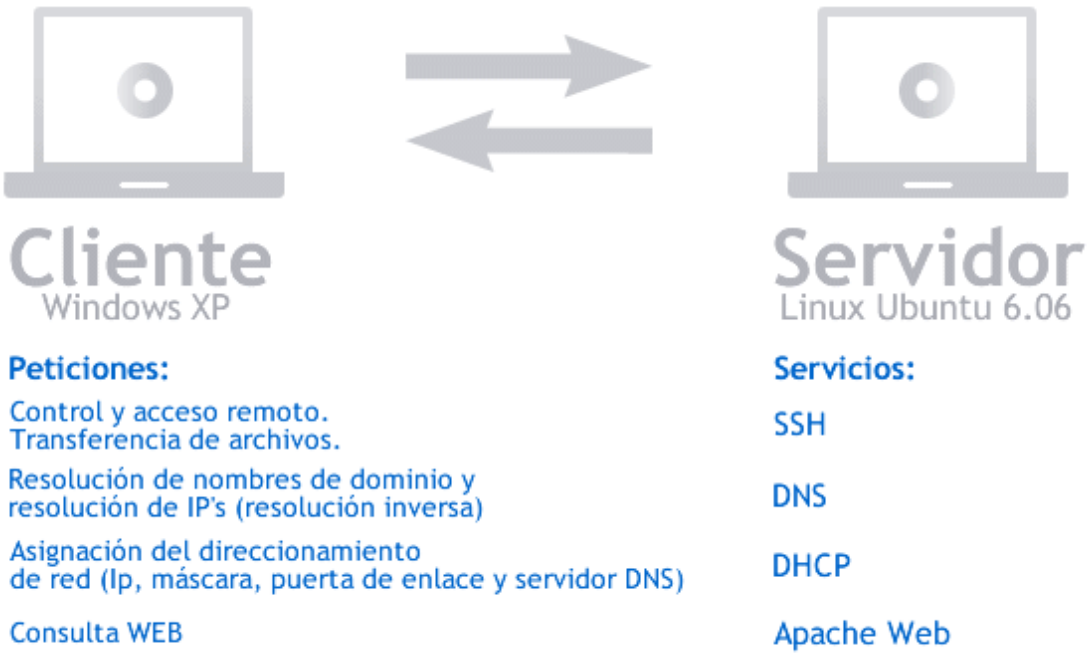


*Figura 11 Esquema de trabajo*

Basándose en este esquema, se configurará Ubuntu para que comparta disco duro físico con otro sistema operativo (WXP) y conseguir que el cliente sea atendido cuando realice las peticiones que mostramos.

## 4.2 Particionamiento del disco duro

Las particiones son los trozos en que está dividido un disco duro. Cuando se dice que un disco duro tiene tres particiones, significa que el disco está dividido en tres partes, no necesariamente iguales. Es decir, está dividido en tres unidades lógicas (las particiones) dentro de una unidad física (el disco duro).

Todo disco duro para poderse utilizar con cualquier sistema operativo debe tener como mínimo una partición. Si sólo tenemos una significará que el disco duro no está dividido y la partición ocupa todo el disco duro. Se puede elegir el tamaño de cada partición en función de lo que irá en cada una de ellas y de la capacidad total del disco duro. En caso de que sobre espacio, será espacio no particionado. El espacio no particionado de un disco duro no es accesible ya que no forma parte de ninguna partición.

Las particiones pueden ser primarias o lógicas. Las particiones primarias son necesarias para arrancar el PC. De las particiones primarias existentes, la definida como activa será la que usará el ordenador para iniciar el sistema operativo instalado en la misma. Sólo pueden existir cuatro particiones en un disco duro. Si una de esas particiones es extendida, en ella podremos albergar ilimitadas particiones lógicas. Las particiones lógicas son aquellas en las que no vamos a instalar ningún sistema operativo y se utilizan mayoritariamente para separar, guardar o ordenar la información como si de una carpeta se tratara, solo que ésta tendría un tamaño limitado.

A cada partición se le debe asignar un sistema de archivos. Un sistema de archivos es el método para nombrar, almacenar y organizar archivos en el equipo. Existen varios tipos de sistemas de archivos: FAT, FAT32, NTFS, EXT2, EXT3... Cada sistema operativo está pensado para un tipo específico de sistema de archivos. Por ejemplo, si se va a instalar un Windows 95, 98 ha de tener al menos una partición primaria con el sistema de archivos FAT32. Si se va a instalar Windows NT, Windows 2000 ó Windows XP el sistema de archivos a utilizar es NTFS, aunque algunos de estos también aceptan el sistema de archivos FAT32. Y como en nuestro caso se va a instalar Linux hay que asignarle el sistema de archivos EXT3 o EXT2. Esto se hace al crear una partición o al darle formato.

En este caso se cuenta con un disco duro de 80 Gb en el cual hay instalado ya Windows XP. No hay creada ninguna partición por lo que toda su capacidad está destinada a XP. No es una buena recomendación. Aunque solo se vaya a tener un sistema operativo, es interesante crear otra partición para separar lo que sería el

almacenamiento de datos, del espacio requerido por el propio sistema operativo y los programas o paquetes que se vayan a utilizar. Si hay problemas y se quiere reinstalar el S.O. se hará solo en la partición en la que esté ubicado, por lo que si tenemos otra donde se almacenan datos (documentos, hojas de calculo, fotos, videos…) no se verá afectada y se podrá realizar la reinstalación sin perder dichos datos. Al realizar una reinstalación se formatea toda la partición, ésta es la razón por la que es recomendable tener en particiones separadas los datos, del sistema operativo.

El disco de 80Gb lo vamos a particionar de la siguiente manera:

- Se dará, aproximadamente, 15Gb para el sistema operativo que ya reside en el ordenador, Windows Xp. El tipo de archivo será NTFS, el soportado por dicho sistema operativo. Con este tipo de archivos, desde la partición Linux solo se podrá acceder en modo lectura, no se podrá escribir pero si consultar.

- Otros 15Gb para Linux. La distribución elegida, como ya se ha comentado anteriormente es Ubuntu v6.06. El tipo de archivo a utilizar será el EXT3, versión más moderna que la EXT2

- 1Gb para la memoria Swap que utilizará Linux. Este tamaño de partición está elegido según una recomendación que da la propia distribución: debe ser el doble que la memoria RAM del ordenador. Tiene 512Mb de memoria RAM por tanto se le dará 1Gb a la partición Swap.

  Swap (intercambiar en inglés) es también llamado el espacio de intercambio. Es una zona del disco (un fichero o partición) que se usa para guardar las imágenes de los procesos que no han de mantenerse en memoria física.

  La mayoría de los sistemas operativos modernos poseen un mecanismo llamado memoria virtual, que permite hacer creer a los programas que tienen más memoria que la disponible realmente. Como en realidad no se tiene físicamente toda esa memoria, algunos procesos no podrán ser ubicados en la memoria RAM. En este caso es cuando es útil el espacio de intercambio: el sistema operativo puede buscar un proceso poco activo y moverlo al área de intercambio y de esa forma liberar la memoria principal para cargar otros procesos. Mientras no haga falta, el proceso extraído de memoria puede quedarse en el disco, ya que ahí no gasta memoria física. Cuando sea necesario, el sistema vuelve a hacer un intercambio, pasándolo del disco a memoria RAM. Es un proceso lento (comparado con usar sólo la memoria RAM), pero permite dar la impresión de que hay más memoria disponible.

- Por último, el espacio restante del disco duro se destinará para tener una partición de almacenamiento de datos, es decir, una partición en la cual no haya ningún sistema operativo instalado y solo sirva para guardar todos los documentos, archivos etc que se quiera y poder disponer de ellos tanto desde Windows como desde Ubuntu. En el siguiente gráfico tenemos como queda dividido el disco duro:

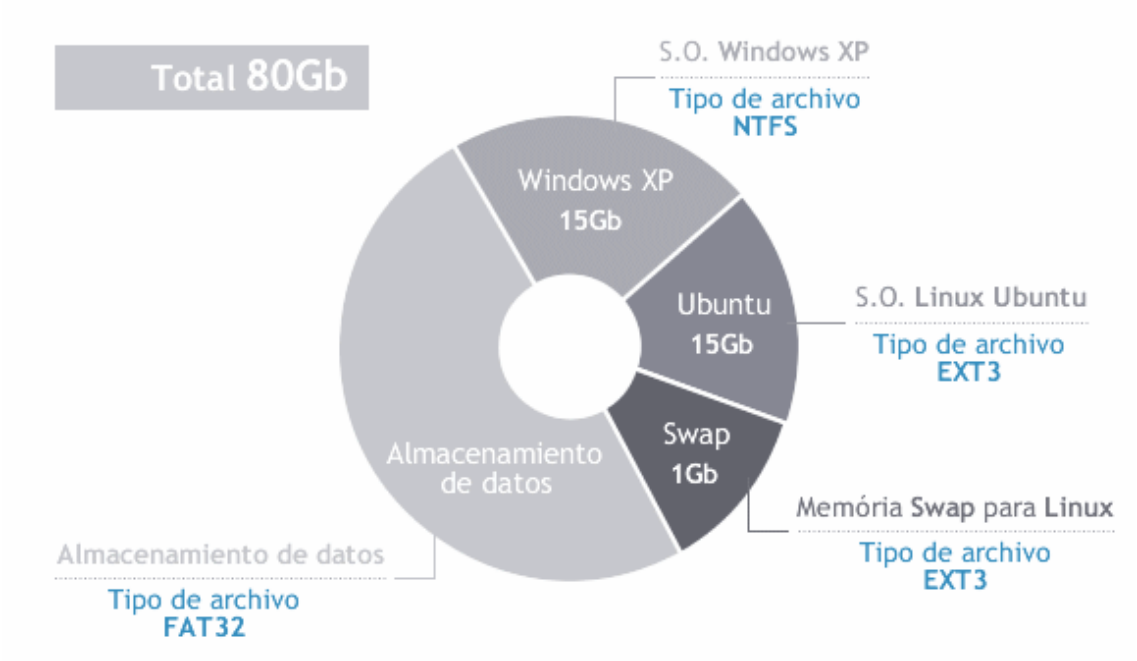


*Figura 12 Particiones del disco duro.*

Para conseguir que esta partición sea accesible en lectura y escritura tanto desde Windows como desde Linux hay que indicar que aloje archivos tipo FAT32. Este tipo de archivos es compatible tanto con Windows como con Linux, en lectura y escritura por lo que desde cualquiera de los sistemas operativos se podrá acceder, consultar y/o modificar lo que se quiera. Windows la reconoce y permite utilizarla automáticamente. En Linux hay que configurarla para poder hacer uso de ella. Simplemente se debe montar el volumen. Hay que crear un directorio (o carpeta) dentro de \etc con el nombre que se le quiera dar a este volumen (o partición compartida). Una vez creada la carpeta, se monta el volumen. Para montarlo hay que modificar el archivo 'fstab'. Con la aplicación Gparted se comprueba el nombre de la partición FAT32:

*/dev/hd* /media/hd* vfat*

Se edita el archivo fstab y los valores siguientes a vfat los cambiamos por estos: umask=000 0 0

Ahí se le está indicando que es del tipo FAT32. Una vez hecho esto solo queda montarla con los comandos siguientes:

*sudo umount –a*

*sudo mount -a*

y ya estará lista para usarse.

Una vez realizadas las particiones en el disco duro se puede proceder a instalar la distribución Linux: ***Ubuntu 6.06 LTS Dapper Drake***.

## 4.3 Instalación de la distribución

La versión de Ubuntu que se va a utilizar, la 6.06 Dapper Drake, permite explorar el nuevo sistema operativo sin necesidad de instalarlo. Esto es conocido como LiveCD, una funcionalidad muy útil para usuarios nuevos en Linux que quieren ver que es lo que tendrán en su ordenador después de instalar este S.O. con el único trabajo de tener que introducir el CD, encender la máquina y esperar a que aparezca la distribución.

Se Introduce el cd y tras unos instantes, el escritorio gráfico de Ubuntu se habrá cargado en la memoria RAM de la máquina y aparecerá en nuestra pantalla. Podemos explorar los menús o probar las aplicaciones para ver una pequeña muestra de lo que Ubuntu puede hacer sin todavía tener nada instalado en el sistema. Hay que tener en cuenta que una vez instalado es mucho más rápido que al ejecutarlo de forma virtual. Para iniciar la instalación hay que seleccionar y hacer doble clic con el ratón en el siguiente icono:


*Figura. 13 Icono de instalación ubuntu*

El proceso de instalación es sencillo, siempre y cuando se haya realizado correctamente el particionamiento del disco (esa es la más parte complicada de la instalación). Simplemente irá pidiendo sucesivamente el idioma, tipo de teclado, franja horaria… así como el o los nombres de usuarios que van a utilizar el equipo y el propio

nombre del equipo. Una vez se tiene todo esto seleccionado únicamente hay que indicar en que partición del disco se instalará. Se seleccionará la partición destinada a Linux con tipo de archivos EXT3 y en seguida se empezará a instalar todo él software necesario así como cantidad de paquetes de aplicaciones con sus correspondientes librerías. El único problema que podría ocurrir es en el reconocimiento de Hardware. Podría suceder que no reconociera alguna parte de nuestra maquinaria (tarjeta de video, sonido, alguna partición) si fuese de esta manera (no es el caso, reconoció perfectamente todo el hardware) se debería investigar en la gran cantidad de documentación que hay en la comunidad Ubuntu alrededor del mundo si es un problema de incompatibilidad de hardware o que tipo de problema se trata.

## 4.4. Instalación y configuración de los servicios

### 4.4.1 Introducción

Ya se tiene instalado el sistema operativo sobre el cual van a correr los servicios que hay que configurar. Como ya se ha comentado anteriormente serán: DHCP, SSH, DNS y servidor WEB.

Estos servicios son de los más comunes que se pueden encontrar en cualquier red informática que disponga de conexión a Internet. El objetivo principal será ponerlos en funcionamiento, configurarlos para que den el servicio correctamente en la LAN sobre la que trabajan y a partir de ahí desarrollar un seguido de ejercicios, prácticos y teóricos, para aplicar su funcionamiento a la docencia de la asignatura de TELEMÁTICA.

Todo el software que se va a utilizar en la parte de servidor es completamente gratuito y libre de licencias, a excepción del sistema operativo Wxp que ya ‘habitaba’ en el disco. Decir que la única función de éste último será la de demostrar que puede compartir medio físico con un s.o. Linux ya que ni tan siquiera se inicializará.

A partir de aquí empieza la parte más interesante del proyecto: la puesta en funcionamiento de los servicios sobre los que se realizarán los ejercicios de aprendizaje. EL escenario donde se va a trabajar es el mostrado en la siguiente figura:

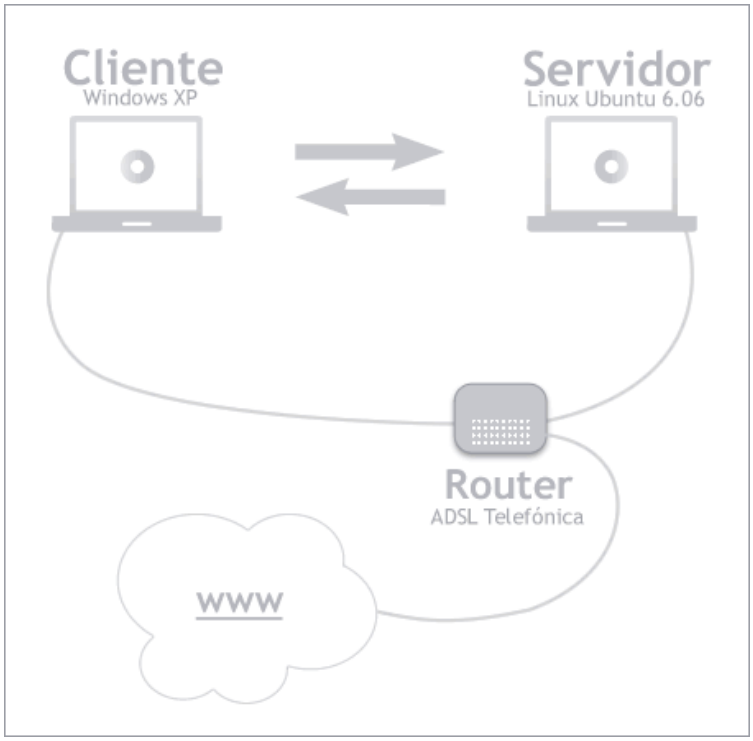


*Figura 14 Red de área local donde se implementarán los servicios.*

Es una pequeña LAN doméstica con una conexión a Internet por ADSL. El ISP es Telefónica, nos proporciona el router ADSL el cual hará alguna de las funciones que se van a desarrollar (servidor DHCP, DNS) por tanto hay que tenerlo en cuenta a la hora de las configuraciones que se hagan de esos servicios, es decir, deshabilitarlas para que sea el servidor el que las realice. Esto se debe realizar ya que si no deshabilitamos estas opciones sería el router quien diera la respuesta en primer lugar por lo que no se podría comprobar el correcto funcionamiento del servidor que se está configurando.

### 4.4.2. DHCP, Dinamic Host Configuration Protocol

#### 4.4.2.1 Introducción

Las siglas DHCP significan Dynamic Host Configuration Protocol. Es utilizado para grandes redes. El demonio actúa dándole información de la red a las estaciones de trabajo, tales como la dirección IP, la máscara de red, DNS Server, la puerta de enlace, etc.

Al igual que otros protocolos similares, utiliza el 'escenario' cliente-servidor, para que los nodos clientes obtengan su configuración del nodo servidor.

El protocolo de Configuración Dinámica de Hosts (DHCP) permite la transmisión de la configuración de los hosts sobre una red TCP/IP. Se encarga de la configuración automática de los parámetros de red.

#### 4.4.2.2 Historia. BOOTP, el antecesor

DHCP fue creado por el Grupo de Trabajo Dynamic Host Configuration del IETF (Internet Engineering Task Force, organización de voluntarios que define protocolos para su uso en Internet). Su definición se encuentra en los RFC's 2131, el protocolo DHCP, y el 2132, opciones de DHCP.

DHCP es una extensión de BOOTP, es decir, mejora BOOTP, y es compatible con él (un cliente puede realizar una petición estática BOOTP a un servidor DHCP).

BOOTP son las siglas de Bootstrap Protocol. Es un protocolo de red UDP utilizado por los clientes de red para obtener su dirección IP automáticamente. Normalmente se realiza en el proceso de arranque de los ordenadores o del sistema operativo. Este protocolo permite a los ordenadores sin disco obtener una dirección IP antes de cargar un sistema operativo avanzado. Históricamente ha sido utilizado por las estaciones de trabajo sin disco basadas en UNIX (las cuales también obtenían la localización de su imagen de arranque mediante este protocolo) y también por empresas para introducir una instalación pre-configurada de Windows en PCs recién comprados (típicamente en un entorno de red Windows NT). Originalmente requería el uso de un disquete de arranque para establecer las conexiones de red iniciales, pero el protocolo se integró en la BIOS de algunas tarjetas de red y en muchas placas base modernas para permitir el arranque directo desde la red.

La siguiente figura muestra una tabla comparativa entre los dos protocolos:

| BOOTP | DHCP |
|---|---|
| Diseñado antes que DHCP. | Diseñado después que BOOTP. |
| Pensado para configurar estaciones de trabajo sin disco con capacidades de inicio limitadas. | Diseñado para configurar equipos de la red que cambian de ubicación frecuentemente (como los equipos portátiles) y que tienen discos duros locales y capacidades de inicio completas. |
| Las concesiones de direcciones IP de BOOTP dinámico caducan a los 30 días de forma predeterminada. | Las concesiones de direcciones IP de DHCP dinámico caducan a los ocho días de forma predeterminada. |
| Admite un número limitado de parámetros de configuración de los clientes llamados *extensiones de proveedor*. | Admite un conjunto mayor y extensible de parámetros de configuración de los clientes llamados *opciones*. |
| Describe un proceso de configuración de inicio en dos fases, como se muestra a continuación:<br>• Los clientes se ponen en contacto con los servidores BOOTP para realizar la determinación de direcciones y la selección del nombre del archivo de inicio.<br>• Los clientes se comunican con los servidores del Protocolo de transferencia de archivos trivial (TFTP, Trivial File Transfer Protocol) para transferir el archivo de imagen de inicio. | Describe un proceso de configuración de inicio de una fase, en el que un cliente DHCP negocia con un servidor DHCP para determinar su dirección IP y obtener otros detalles de configuración inicial que necesita para funcionar en la red. |
| Los clientes BOOTP no vuelven a crear los enlaces con el servidor BOOTP ni renuevan la configuración excepto cuando se reinicia el sistema. | Los clientes DHCP no requieren que el sistema se reinicie para volver a crear el enlace o renovar la configuración con el servidor DHCP. En su lugar, los clientes entran automáticamente en estado de renovación del enlace según intervalos de tiempo establecidos para renovar la asignación de la dirección concedida con el servidor DHCP. Este proceso se produce en segundo plano y es transparente para el usuario. |

*Figura 15 Tabla comparativa BOOTP y DHCP*

DHCP es un protocolo que permite asignar direcciones IP dinámicas, de forma totalmente automática. Por ello, no pierde las prestaciones de BOOTP, su antecesor, sino que las amplía permitiendo nuevas formas de asignación de direcciones y nuevas opciones para poder pasar a los clientes toda la información necesaria. DHCP es un protocolo implementado en los principales sistemas operativos así como otros dispositivos.

DHCP puede usarse cuando el número de IPs es menor que el número de hosts y todos no están conectados a la vez, como en un proveedor de servicio de Internet ISP (Telefónica, ONO, JAZZTEL…).

#### 4.4.2.3 Funcionamiento y tipo de configuraciones

DHCP se encuentra en la capa de aplicación de la pila de protocolos OSI.

DHCP está formado por dos partes: un protocolo para el intercambio de los parámetros de red específicos de cada host y un mecanismo para la asignación de direcciones de red.

Un servidor DHCP tiene dos bases de datos. La primera es estática, al igual que BOOTP y la segunda contiene una pila de direcciones IP disponibles. Esta segunda base de datos hace a DHCP dinámico. Cuando un cliente DHCP pide una dirección IP temporal, DHCP la coge de la pila de direcciones IP disponibles y se la asigna durante un periodo de tiempo negociado.

El servidor admite tres tipos de configuración de direcciones IP:

1. Estática. Se configura en el servidor la dirección de red que se corresponde con la dirección LAN del cliente (equivalente a BOOTP).
2. Dinámica, por tiempo ilimitado. Se indica un rango de direcciones que se asignan a cada cliente de carácter permanente, hasta que el cliente la libera.
3. Dinámica, arrendada. Las direcciones se otorgan por un tiempo ilimitado. Un cliente debe renovar su dirección para poder seguir utilizándola.

Cuando el servidor DHCP recibe una petición, primero chequea su base de datos estática. Si existe una entrada para esa dirección física (M.A.C. adress), se devuelve la dirección IP estática correspondiente. Si no se encuentra la entrada, el servidor selecciona una IP disponible de la base de datos dinámica y añade la nueva asociación a la base de datos.

**Alquiler**:

La dirección asignada desde la pila es temporal. El servidor DHCP emite un alquiler por un periodo determinado de tiempo. Cuando el alquiler termina, el cliente debe, dejar de usar la IP o renovar el alquiler. El servidor tiene la opción de aceptar o denegar la renovación.

**Operación**:

El cliente realiza los siguientes pasos:

Envía un mensaje DHCP DISCOVER broadcast usando el puerto destino 67.

Aquellos servidores que puedan dar este tipo de servicio responden con un mensaje DHCP OFFER, donde se ofrece una IP que será bloqueada. En estos mensajes también puede ofrecer la duración del alquiler que por defecto es de una hora. Si los clientes no reciben dicho mensaje, intenta establecer conexión cuatro veces más, cada dos segundos, si aún así no hay respuesta, el cliente espera cinco minutos antes de intentarlo de nuevo.

El cliente elige una de las IPs ofertadas y envía un mensaje DHCP REQUEST al servidor seleccionado.

El servidor responde con un mensaje DHCP ACK y crea la asociación entre la dirección física del cliente y su IP. Ahora el cliente usa la IP hasta que el alquiler expire.

Antes de alcanzar el 50% del tiempo del alquiler, el cliente envía otro mensaje DHCP REQUEST para renovar el alquiler.

Si el servidor responde con DHCPACK, el cliente puede seguir usando la IP durante otro periodo de tiempo. Si se recibe un DHCPNACK, el cliente debe de dejar de usar esa IP y empezar de nuevo el proceso de obtención de una IP.

Si después de transcurrir el 87.5% del alquiler no se recibe respuesta, se manda otro DHCPREQUEST.

Si se recibe un DHCPACK antes de que expire el tiempo de alquiler, se obtiene más tiempo de alquiler. En caso contrario, se debe comenzar de nuevo el proceso de obtención de una IP. El cliente puede terminar el alquiler antes de que expire el tiempo. En este caso, el cliente envía un mensaje DHCPRELEASE al servidor.

Este sería el esquema secuencial de los pasos seguidos en la asignación del direccionamiento:

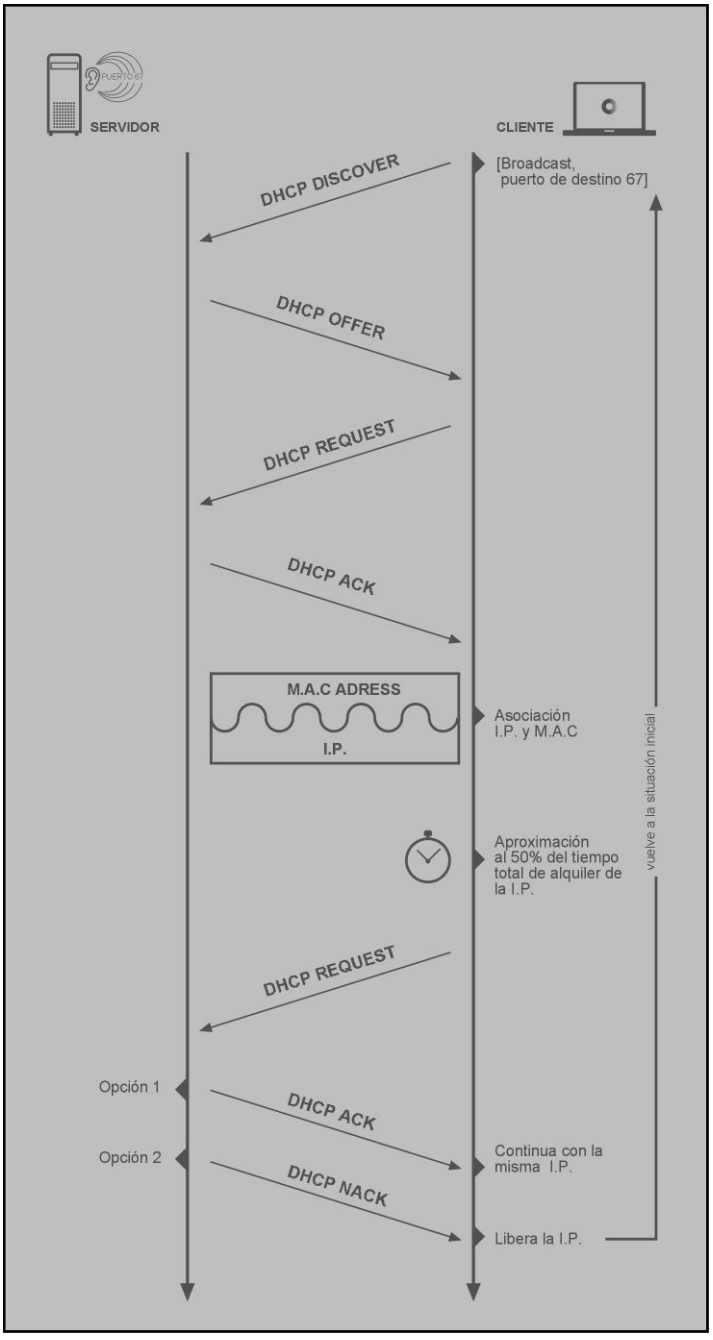


*Figura 16 Mensajes en la asignación de direccionamiento DHCP.*

#### 4.4.2.4 Formato del paquete

Para hacer DHCP compatible con BOOTP, los diseñadores de DHCP han usado casi el mismo formato de paquete. Solo se ha añadido un bit de control al paquete. Sin embargo, se han añadido opciones extra para permitir las diferentes interacciones con el servidor. Los campos diferentes de DHCP son los siguientes:

Flag: 1 bit. El primero del campo sin uso, que permite al cliente el forzar que la respuesta del servidor sea broadcast en vez de unicast. Si la respuesta es unicast, la dirección destino será la del cliente y este no la conoce, por lo que puede descartar el mensaje. Al ser broadcast, todos los computadores reciben y procesan el mensaje.

Opciones: Se han añadido varias posibilidades a la lista de opciones. La opción con etiqueta 53 es la que define el tipo de interacción entre el cliente y el servidor. Otras opciones definen parámetros con el tiempo de alquiler, etc. El campo de opción en DHCP puede tener hasta 312 bytes.

La asignación de direcciones IP se configurará de un modo u otro, dependiendo de cada situación. Puede interesar un direccionamiento estático para clientes sin disco o por facilidades administrativas, pero controlando la asignación de cada dirección a cada cliente (es más cómodo para el administrador configurar un servidor, que cada cliente; interesa el direccionamiento estático para evitar que se conecten clientes no identificados o por otras razones, como la configuración DNS).

El direccionamiento dinámico por tiempo ilimitado se utiliza cuando el número de clientes no varía demasiado, facilitando mucho la tarea del administrador.

El arrendamiento de direcciones se emplea para racionar las direcciones IP, minimizando el coste administrativo. En función de la frecuencia de inserciones/eliminaciones de clientes y de la cantidad de direcciones disponibles se concederá un mayor o menor tiempo de arrendamiento. El tiempo será bajo (ej. 15 minutos); si se conectan/desconectan los clientes con mucha frecuencia e interesa que este disponible el máximo número de direcciones. Por el contrario se utilizará un tiempo largo para que cada cliente mantenga su dirección IP (ej. en una universidad un tiempo de 4 meses, tiempo máximo que esta desconectado en vacaciones para asumir que el cliente ya no esta en la red). Un portátil puede tener una dirección permanente o de larga duración en su red habitual de trabajo y tiempos cortos en otras redes.

#### 4.4.2.5 Ventajas y desventajas del DHCP

Las ventajas del uso de DHCP serían:

a) Solo se configura un servidor para entregar las IPs para clientes de red.

b) Se entregan todos los parámetros básicos de TCP-IP.

c) Facilidad de configuración.

Las desventajas serían:

a) Al entregar las direcciones IP dentro de la red, habiendo un DNS, no hay un puente intermedio entre DNS y DHCP directo. Es decir, hay que agregar las máquinas “a mano” en el DNS.

b) Los mensajes tienden a fallar sobre todo si las tarjetas de red hacen la negociación de velocidad (más conocido como Network Speed Auto-Sense, que falla con una rapidez increíble) ya que la red se llena de “basura física”.

#### 4.4.2.6 Instalación y configuración

##### 4.4.2.6.1 Situación Inicial

Antes de empezar a instalar y configurar este servicio hay que detenerse un momento y pensar en que situación va a trabajar, es decir, conocer la red en la que se va a instalar este servidor.

La red sale a Internet mediante un router ADSL, el cual hace función, entre otras cosas, de servidor DHCP. Por tanto esta opción se deshabilitará para que sea el servidor Linux Ubuntu el que asigne el direccionamiento a los equipos de la LAN. Se accede al router ADSL por http. Entramos en su interficie gráfica y se deshabilita la opción: ‘*DHCP server’*, se pone *disable.*

Se va a trabajar con toda la clase C 192.168.1.X. Por lo tanto:

IP de la net: 192.168.1.0 máscara: 255.255.255.0

IP del Router: 192.168.1.1

Rango de IP’s para la LAN: 192.168.1.90 a la 192.168.1.100 (con estas 10 direcciones IP sobrará, es un rango escogido al azar de las 254 direcciones que hay disponibles).

Servidor DNS principal: 192.168.1.33 (servidor DNS en nuestra máquina Linux que más adelante se hablará de él).

Servidor DNS secundario: 192.168.1.1 (otra de las funciones que realiza el router ADSL)

Interfaces del servidor: 'eth0' Tarjeta de red Ethernet y 'eth1' tarjeta de red inalámbrica.

Una vez conocido el 'escenario' en el que va a trabajar se puede proceder a la instalación y la configuración del servicio DHCP

#### 4.4.2.6.2 Instalación y configuración de dhcp3-server

Se instala dhcp3-server en la máquina Linux.

```
sudo apt-get install dhcp3-server
```

Se hace una copia de seguridad del archivo dhcp3-server.

```
sudo cp /etc/default/dhcp3-server /etc/default/dhcp3-server_backup
```

Esto es importante realizarlo siempre antes de empezar a editar cualquier archivo de configuración para prevenir algún posible problema y no perder la configuración que tenía antes de empezar a realizar los cambios y poder realizar 'marcha atrás'.

Se edita el archivo dhcp3-server para indicarle que interfaz es la que va a trabajar como servidor. Se realiza en la línea que contenga lo siguiente:

*...*

*INTERFACES=" "*

Y se le indica que va ser la interfaz *eth0:*

*INTERFACES="etho "*

Salvar los cambios hechos y se procede a editar el fichero que dará la configuración de Ip's a los clientes. Esto lo está en el archivo: *dhcpd.conf.* Lo primero es realizar la copia de backup:

```
sudo cp /etc/dhcp3/dhcpd.conf /etc/dhcp3/dhcpd.conf_backup
```

Se edita el archivo:

```
vi dhcpd.conf
```

El editor VI es el editor por excelencia de UNIX. Es bastante potente y complicado de manejar a pleno rendimiento debido a la gran cantidad de opciones que tiene, pero es fácil dar los primeros pasos e ir avanzando poco a poco. Es difícil de utilizar al principio, pero puede llegar a convertirse en el más cómodo y rápido. En UNIX existen otros editores más potentes, con entornos de trabajo más amigables, otros más sencillos de manejar, pero el único editor que está en todas las versiones y se maneja igual es el vi. Esta es la gran razón por la que se ha querido utilizar.

Lo primero que hay que hacer es comentar, para que no se ejecuten, las líneas que vienen por defecto con el tiempo de 'alquiler' de las IP's que asignará:

```
# option definitions common to all supported networks...

#option domain-name "example.org";

#option domain-name-servers ns1.example.org, ns2.example.org;

#default-lease-time 600;

#max-lease-time 7200;
```

*Figura 17 Configuración tiempo indefinido alquiler de las IP's.*

Con esto las IP's que se asignen a los clientes tendrán tiempo indefinido (hasta que se desconecte la máquina o se apague).

Hay que notar que el tiempo de alquiler viene expresado en segundos y con solo descomentar las líneas siguientes le podemos indicar de cuanto tiempo queremos que disponga el cliente del alquiler de la IP:

```
#default-lease-time 600;

#max-lease-time 7200;
```

*Figura 18 Configuración del tiempo de alquiler de la IP.*

Ahora tocará definir el rango de IP's que se asignarán así como la configuración de red que tendrán los clientes (DNS, máscara de red…). En el mismo archivo dhcpd.conf se encuentran las líneas donde se define todo esto:

```
# A slightly different configuration for an internal subnet.
subnet 192.168.1.0 netmask 255.255.255.0 {
 range 192.168.1.90 192.168.1.100;
 option domain-name-servers 192.168.1.33, 192.168.1.1 ;
 option domain-name "homeip.net";
 option routers 192.168.1.1;
 option broadcast-address 192.168.1.255;
                                    }
```

*Figura 19 Dhcpd.conf, configuración de los parámetros de red.*

Una vez salvados los cambios hay reiniciar el demonio dhcp:

```
sudo /etc/init.d/dhcp3-server restart
```

Para comprobar con el cliente si el servidor hace la asignación correcta del direccionamiento de red hay que, como ya se ha comentado anteriormente, deshabilitar el servidor DHCP que tenia anteriormente (en el router de Telefónica) o si tiene la asignación de Ip estática configurarlo como dinámico. Se reinicia la máquina y se comprueba que la IP que se le asigna esté dentro del rango que hemos definido así como los demás parámetros de red.

### 4.4.2.6.3 Configuración del Cliente

La configuración de cliente es más fácil. Es un cliente con Windows XP. Se abre 'conexiones de red dentro' del panel de control:

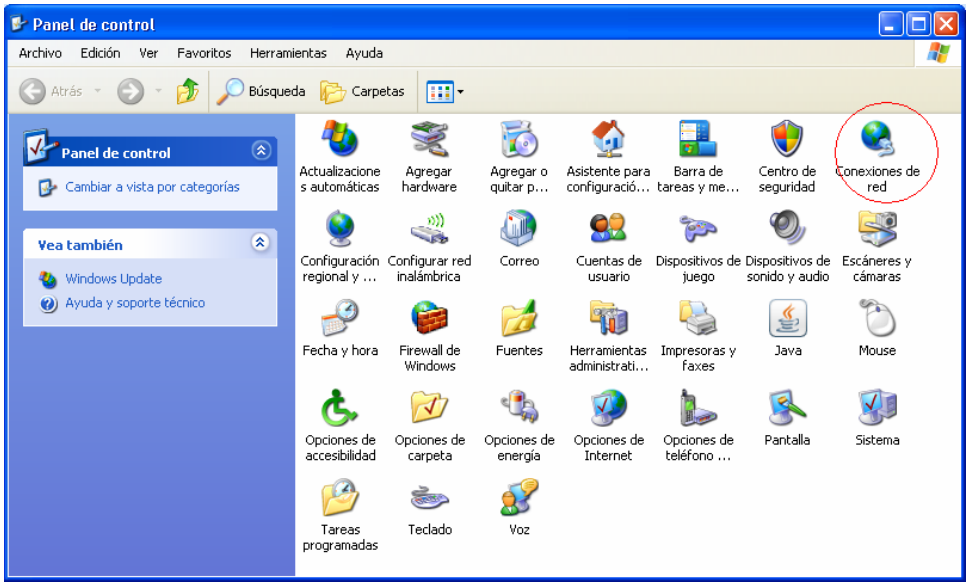


*Figura 20 Panel de Control Wxp*

Y en la conexión que nos aparezca: Botón derecho → Propiedades:

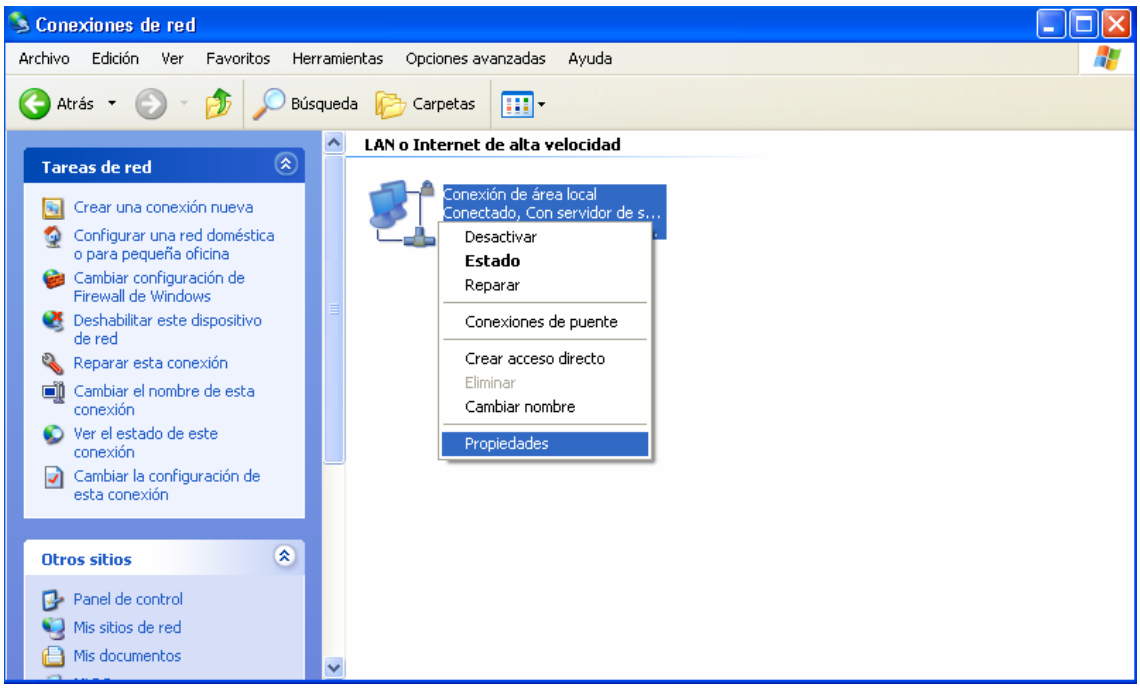


*Figura 21 Propiedades de la conexión de red.*

Una vez dentro de las propiedades de la conexión de red, se busca 'protocolo TCP/IP' y doble clic:

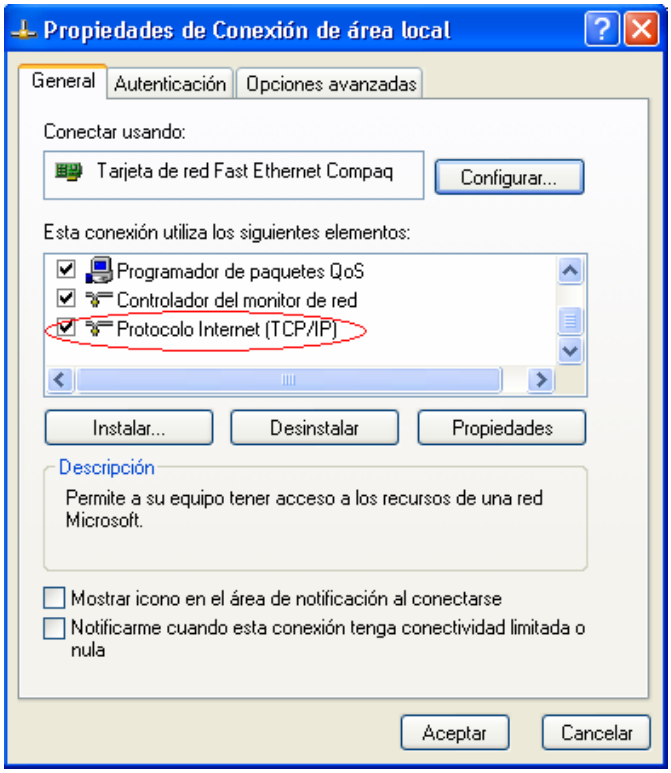


*Figura 22 Propiedades del protocolo TCP/IP.*

Dentro de las propiedades del Protocolo TCP/IP se seleccionarán las opciones marcadas en la siguiente figura:

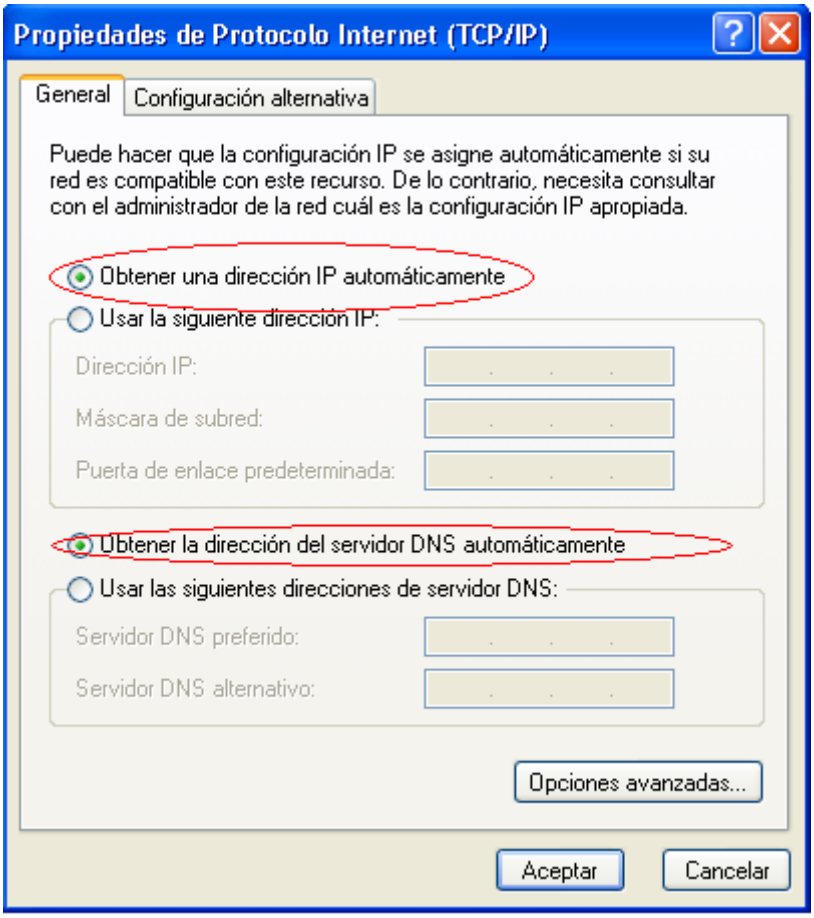


*Figura 23 Selección del direccionamiento automático.*

Se aplican y aceptan los cambios y ya estará configurado el cliente para que obtenga una configuración de red automática gracias al protocolo DHCP.

Arrancaremos el PC y comprobaremos que la asignación es del rango que se ha configurado en el servidor. Podemos realizar la comprobación también con las instrucciones *'ipconfig /release'* y *'ipconfig /renew'.* Con ellas liberaremos y renovaremos la IP que tenemos y así podremos comprobar si es correcta la asignación de la IP. A continuación se ven dichas instrucciones.

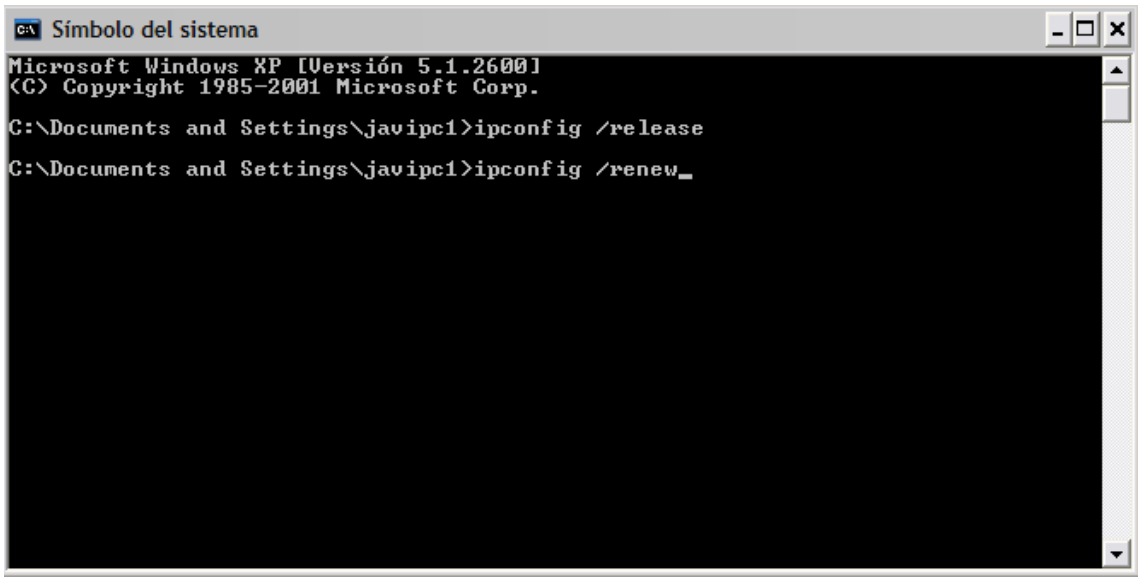


*Figura 24 Comandos para liberar y renovar la IP*

### 4.4.3. SSH, Secure Shell

#### 4.4.3.1 Introducción

SSH (Secure Shell, RFC 4252) es una aplicación, con protocolo propio, diseñada para sustituir herramientas de acceso remoto usadas tradicionalmente en los sistemas UNIX como pueden ser el rsh (*Remote Shell*), rlogin (*Remote Login*) o rcp (*Remote Copy*), Telnet…

El protocolo que define esta aplicación es la gran ventaja y característica con respecto a las otras comentadas. Está diseñado para la transmisión segura de datos.

Este protocolo se sitúa por debajo de la capa de transporte TCP y proporciona servicios análogos a los del protocolo SSL/TLS.

A parte de proporcionar seguridad en las conexiones también tiene otras funcionalidades como la redirección de puertos TCP o la comunicación entre clientes y servidores de ventanas X.

La primera versión de SSH fue obre de Tatu Ylönen, en la Universidad Tecnológica de Helsinki sobre el año 1995. Se sigue trabajando en la versión 2.0 y aunque la funcionalidad que da esta versión mas moderna es prácticamente la misma que la de la primera, tiene muchas mejoras y es sustancialmente distinta a la anterior. Se denomina SSH2 y es con la que se profundizará en el tema.

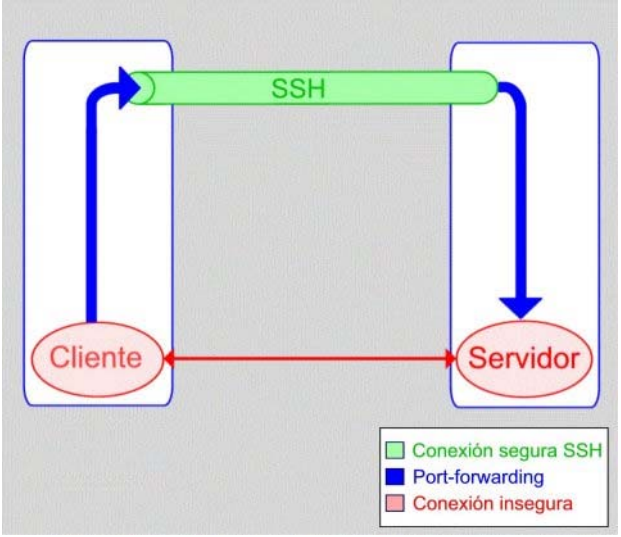


*Figura 25 Esquema de una conexión SSH.*

### 4.4.3.2 Características principales de SSH

**Confidencialidad**

SSH sirve para comunicar datos, que habitualmente son entrada de una aplicación remota y la salida que genera, o bien la información que se transmite por un puerto redirigido.

La confidencialidad de estos datos se garantiza mediante el cifrado.
En SSH se aplica un cifrado simétrico a los datos, por lo tanto, será necesario realizar previamente un intercambio seguro de claves entre cliente y servidor. En SSH2 se pueden utilizar algoritmos de cifrado distintos en los dos sentidos de la comunicación, una diferencia importante respecto al protocolo SSL/TLS.

SSH2 también permite ocultar ciertas características del tráfico como la longitud real de los paquetes.

**Autenticación de entidad**

El protocolo SSH proporciona mecanismos para autenticar tanto el servidor como el usuario que se quiere conectar.

La autenticación del servidor suele realizarse conjuntamente con el intercambio de claves. En SSH2 el método de intercambio de claves se negocia entre el cliente y el servidor, aunque actualmente sólo hay uno definido, basado en el algoritmo de Diffie-Hellman. Este algoritmo (los creadores fueron Whitfield Diffie y Martin Hellman)

permite el intercambio secreto de claves entre dos partes que no han tenido contacto previo, utilizando un canal inseguro y de manera anónima.

Para autenticar al usuario existen distintos métodos; dependiendo de cuál se utilice, puede ser necesaria también la autenticación del ordenador cliente, mientras que otros métodos permiten que el usuario debidamente autenticado acceda al servidor desde cualquier ordenador cliente.

**Autenticación de mensaje**

La autenticidad de los datos se garantiza añadiendo a cada paquete un código MAC calculado con una clave secreta. También existe la posibilidad de utilizar algoritmos MAC distintos en cada sentido de la comunicación.

**Eficiencia**

SSH contempla la compresión de los datos intercambiados para reducir la longitud de los paquetes. SSH2 permite negociar el algoritmo que utilizará en cada sentido de la comunicación, aunque solamente existe uno definido en la especificación del protocolo.

En SSH no está prevista la reutilización de claves de sesiones anteriores, es decir, en cada nueva conexión se vuelven a calcular las claves. Esto es así porque SSH está pensado para conexiones que tienen una duración más o menos larga, como suelen ser las sesiones de trabajo interactivas con un ordenador remoto, y no para las conexiones cortas pero consecutivas, que son más típicas del protocolo de aplicación http. Hay que decir que SSH define mecanismos para intentar acortar el proceso de negociación.

**Extensibilidad**

En SSH2 también se negocian los algoritmos de cifrado, de autenticación de usuario, de MAC, de compresión y de intercambio de claves.

Cada algoritmo se identifica con una cadena de caracteres que representa su nombre. Los nombres pueden corresponder a algoritmos oficialmente registrados, o bien a algoritmos propuestos experimentalmente o definidos localmente.

#### 4.4.3.3 La Capa de Transporte SSH y su protocolo

El protocolo de capa de transporte se encarga del establecimiento de la conexión de transporte, de la autentificación del servidor, intercambio de claves y de las peticiones de servicio de los demás protocolos.

El cliente se conectará mediante TCP al servidor. El servidor debe estar escuchando peticiones de conexión en el puerto 22 que es el asignado al servicio SSH.

Una vez establecida la conexión lo primero que hacen servidor y cliente es establecer la versión que se utilizará en la conexión. Tanto el cliente como el servidor envían una línea que contiene el texto “SSH-x.y-implementación”, donde x.y es el número de versión del protocolo (por ejemplo, 2.0) e implementación es una cadena identificativa del software del cliente o servidor. Si los números de versión no concuerdan, el servidor decide si puede continuar o no: si no puede, simplemente cierra la conexión.

Cuando han acordado la versión a utilizar pasan a intercambiarse mensajes con el protocolo de paquetes SSH inicialmente sin cifrar y sin MAC.

El primer paquete SSH se puede enviar juntamente con la línea que indica la versión, sin esperar a recibir la línea de la otra parte. Si las versiones coinciden, el protocolo continúa normalmente; si no, puede ser necesario reiniciarlo.

En SSH2 cada parte envía un mensaje KEXINIT que contiene una cadena de 16 bytes aleatorios llamada cookie, y las listas de algoritmos soportados por orden de preferencia: algoritmos de intercambio de claves y, para cada sentido de la comunicación, algoritmos de cifrado simétrico, de MAC y de compresión. También se incluye una lista de idiomas soportados por los mensajes informativos. Para cada tipo de algoritmo, se escoge el primero de la lista del cliente que esté también en la lista del servidor.

Los algoritmos criptográficos que contempla SSH2 son los siguientes:

- Diffie-Hellman, para el intercambio de claves.
- RC4, Triple DES, Blowfish, Twofish, IDEA y CAST-128 para el cifrado.
- Para el MAC: HMAC-SHAI y HMAC-MD5 (con todos los bytes o con los 12 primeros).

Los paquetes que vienen a continuación son los de intercambio de claves, y dependen del algoritmo escogido (aunque SSH2 sólo prevé el algoritmo de Diffie-Hellman).

Se puede suponer que la mayoría de implementaciones tendrán un mismo algoritmo preferido de cada tipo. De este modo, para reducir el tiempo de respuesta, se puede enviar el primer mensaje de intercambio de claves después del KEXINIT sin esperar el de la otra parte, utilizando estos algoritmos preferidos. Si la suposición resulta acertada, el intercambio de claves continúa normalmente, y si no, los paquetes enviados anticipadamente se ignoran y se vuelven a enviar con los algoritmos correctos.

Independientemente del algoritmo utilizado, después del intercambio de claves, se obtiene un secreto compartido y un identificador de sesión. Con el algoritmo Diffie-Hellman, este identificador es el a.C. de una cadena formada, entre otras, por las cookies del cliente y el servidor. Las claves de cifrado, de MAC y los vectores de inicialización se calculan aplicando funciones hash de varias formas a distintas combinaciones del secreto compartido y del identificador de sesión.

Finalmente utilizarán un mensaje NEWKEYS para avisarse de que el próximo paquete utilizará un algoritmo y unas claves nuevas.

Estos procesos se pueden repetir cuando se quiera. Las especificaciones de SSH2 recomiendan hacerlo después de cada 1Gb de información traspasada y/o de cada hora de conexión.

Si se produce algún error se genera un mensaje DISCONNECT, que puede contener un texto explicativo del error, y se cierra la conexión. A continuación se muestran otros mensajes que pueden enviarse en cualquier momento:

- IGNORE: su contenido debe ser ignorado, pero se puede usar para contrarrestar el análisis de flujo de tráfico.
- DEBUG: sirven para enviar mensajes informativos.
- UNIMPLEMENTED: se envían en respuesta a mensajes de tipo desconocido.

En SSH2, una vez finalizado el intercambio de claves, el cliente envía un mensaje SERVICE_REQUEST para solicitar un servicio, que puede ser autenticación de usuario, o bien acceso directo al protocolo de conexión si no es necesaria autenticación. El servidor responde con SERVICE_ACCEPT si permite el acceso al servicio solicitado, o con DISCONNECT en caso contrario.

#### 4.4.3.4 El protocolo de paquetes SSH

El protocolo de paquetes SSH se encarga de construir e intercambiar las unidades del protocolo, que son los paquetes SSH. Cuando se envían datos a los mensajes de los niveles superiores se les aplica los siguientes pasos en este mismo orden mostrado:

- La compresión.
- El código de autenticación MAC.
- El cifrado.

El receptor efectúa los pasos inversos:

- Descifrado.
- Verificación de autenticidad..
- Descompresión.

En la siguiente figura vemos un paquete SSH2:

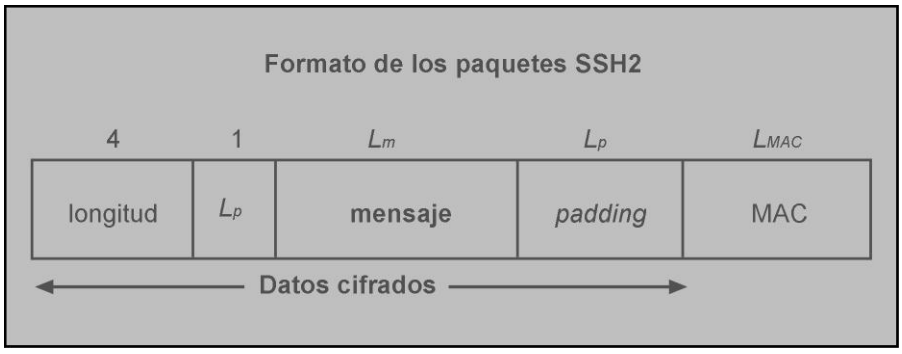


*Figura 26 Formato del paquete SSH*

- El primero es la longitud del resto del paquete, excluido el MAC (por lo tanto, es igual a 1+*a.m.*+*PL*).

- El segundo campo indica cuántos bytes de padding existen. Los bytes de padding son los encargados de 'rellenar' el paquete, si es necesario, para que tenga la longitud adecuada para el cifrado. Este número de bytes debe ser tal que la longitud total del paquete, excluido el MAC, sea múltiple de 8 (o de la longitud de bloque en los cifrados de bloque, si es más grande que 8).

- El tercer campo es el contenido del mensaje, comprimido si se da el caso. El primer byte del contenido siempre indica de qué tipo de mensaje se trata, y la estructura del resto de bytes depende del tipo.

- El cuarto son los bytes aleatorios de padding. Siempre están presentes incluso cuando el cifrado utilizado sea en flujo, y su longitud tiene que ser como mínimo igual a 4. Por lo tanto, la longitud mínima de un paquete, sin contar el MAC, es de 16 bytes.

- El quinto campo es el código de autentificación MAC, obtenido mediante la técnica HMAC a partir de una clave secreta, un número de secuencia implícito de 32 bits y el valor de los otros cuatro campos del paquete. La longitud del MAC depende del algoritmo acordado, y puede ser 0 si se utiliza el algoritmo nulo.

Cuando se cifran los paquetes, se aplica el cifrado a todos los campos excepto el del MAC, pero incluyendo la longitud. Eso significa que el receptor tiene que descifrar los 8 primeros bytes de cada paquete para conocer la longitud total de la parte cifrada.

#### 4.4.3.5 El protocolo de autenticación de usuarios

Se contemplan distintos métodos de autenticación de usuario:

**Autenticación nula.** El servidor permite que el usuario acceda directamente, sin ninguna comprobación, al servicio solicitado. Un ejemplo sería el acceso a un servicio anónimo.

**Autenticación basada en listas de acceso.** A partir de la dirección del sistema cliente y el nombre del usuario de este sistema que solicita el acceso, el servidor consulta una lista para determinar si el usuario está autorizado a acceder al servicio. Dada su vulnerabilidad a ataques de falsificación de dirección IP, este método sólo se puede utilizar en SSH1: SSH2 no lo soporta.

**Autenticación basada en listas de acceso con autenticación de cliente.** Es igual que el anterior, pero el servidor verifica que el sistema cliente sea efectivamente quien dice ser, para evitar los ataques de falsificación de dirección.

**Autenticación basada en contraseña.** El servidor permite el acceso si el usuario da una contraseña correcta.

**Autenticación basada en clave pública.** En lugar de dar una contraseña, el usuario se autentica demostrando que posee la clave privada que corresponde a una clave pública reconocida por el servidor.

En SSH2 el cliente va mandando mensajes USERAUTH_REQUEST, que incluyen el nombre de usuario, el método de autenticación solicitado y al servicio que se quiere acceder. Si el servidor permite el acceso responderá con un mensaje USERAUTH_SUCCESS; si no, enviará un mensaje del tipo USERAUTH_FAILURE, que contiene la lista de métodos de autenticación que se pueden continuar intentando, o bien cerrará la conexión si ya se han producido demasiados intentos o ha pasado demasiado tiempo. El servidor puede enviar opcionalmente un mensaje informativo USERAUTH_BANNER antes de la autenticación.

Los mensajes de solicitud de autenticación contienen la siguiente información según el método:

Para la autenticación nula no se precisa ninguna información adicional.

En la autenticación basada en listas de acceso (solamente aplicable a SSH1) es necesario dar el nombre del usuario en el sistema cliente. La dirección de este sistema se supone disponible por medio de los protocolos subyacentes (TCP/IP).

Cuando se utilizan listas de acceso con autenticación de cliente, en SSH2 el cliente envía su nombre DNS completo, el nombre del usuario local, la clave pública del sistema cliente (y certificados, si tiene), la firma de una cadena de bytes que incluye el identificador de sesión, y el algoritmo con el que se ha generado esta firma. El servidor debe validar la clave pública y la firma para completar la autenticación.

En la autenticación con contraseña sólo es preciso enviar directamente la contraseña. Este método no debería estar permitido si el protocolo de la sub-capa de transporte SSH utiliza el algoritmo de cifrado nulo.

La autenticación basada en clave pública es parecida a la de listas de acceso con autenticación de cliente. En SSH2 el cliente debe enviar un mensaje que contenga la clave pública del usuario (y los certificados, si tiene), el algoritmo que corresponde a esta clave y una firma en la cual interviene el identificador de sesión. El servidor dará por buena la autenticación si verifica correctamente la clave y la firma.

Opcionalmente, para evitar cálculos e interacciones innecesarias con el usuario, el cliente puede enviar antes un mensaje con la misma información pero sin la firma, para que el servidor responda si la clave pública que se le ofrece es aceptable.

Cuando el proceso de autenticación se haya terminado correctamente, en SSH2 se pasa al servicio que el cliente haya solicitado en su último mensaje USERAUTH_REQUEST (el que ha dado lugar a la autenticación correcta). Actualmente sólo existe un servicio definido, que es el de conexión.

#### 4.4.3.6 El protocolo de conexión

El protocolo de conexión gestiona las sesiones interactivas para la ejecución remota de comandos, mandando los datos de entrada de cliente a servidor y los de salida en sentido inverso. También se encarga de la redirección de puertos TCP.

La conexión SSH se puede utilizar, por ejemplo, como túnel de otras conexiones a través de un cortafuegos que esté situado entre el cliente y el servidor SSH.

SSH contempla la posibilidad de utilizar lo que se conoce como agente de autenticación. Este agente es un proceso que permite automatizar la autenticación del usuario basada en claves públicas cuando es necesario realizarla desde un ordenador remoto.

Cada sesión interactiva, conexión TCP redirigida o conexión a un agente es un canal. Pueden existir distintos canales abiertos en una misma conexión SSH, cada uno identificado con un número en cada extremo (los números asignados a un canal en el cliente y en el servidor pueden ser diferentes).

SSH2 prevé catorce tipos de mensajes que se pueden enviar durante la fase de conexión. Éstas son algunas de las funciones que permiten realizar los mensajes:

**Abrir un canal.** Se pueden abrir canales de distintos tipos: sesión interactiva, canal de ventanas X, conexión TCP redirigida o conexión con un agente de autenticación.

**Configurar parámetros del canal.** Antes de empezar una sesión interactiva el cliente puede especificar si necesita que se le asigne un pseudoterminal en el servidor. Existen otros mensajes para indicar si se quiere conexión con el agente de autenticación o redirección de conexiones de ventanas X.

**Empezar sesión interactiva.** Una vez configurados los parámetros necesarios, el cliente puede dar el nombre de un comando que se deba ejecutar en el servidor o bien indicar qué quiere ejecutar un intérprete de comandos. Además de un proceso remoto, en SSH2 también existe la posibilidad de iniciar un "subsistema", como podría ser una transferencia de ficheros (SFTP).

**Enviar datos.** En SSH2 existen dos tipos de mensaje con este fin: uno para enviar datos normales en cualquier sentido y para cualquier canal (incluyendo las sesiones interactivas), y otro para enviar datos especiales (por ejemplo, los de la salida de error stderr). Además de los datos de la sesión, el cliente

también puede enviar un mensaje para indicar que ha recibido una señal o que se ha producido un cambio en las dimensiones del terminal.

**Cerrar canal.** Cuando termina la ejecución normal del proceso o intérprete de comandos, el servidor envía un mensaje indicando el código de salida (el valor numérico que devuelve el proceso). Si ha terminado a causa de una señal, en SSH2 envía un mensaje con el número de señal. Existen otros mensajes que sirven para indicar que ya no hay más datos de entrada, para solicitar el cierre de un canal desde un extremo y para confirmar el cierre desde el otro extremo.

En la siguiente figura se muestra el intercambio de todos estos mensajes.

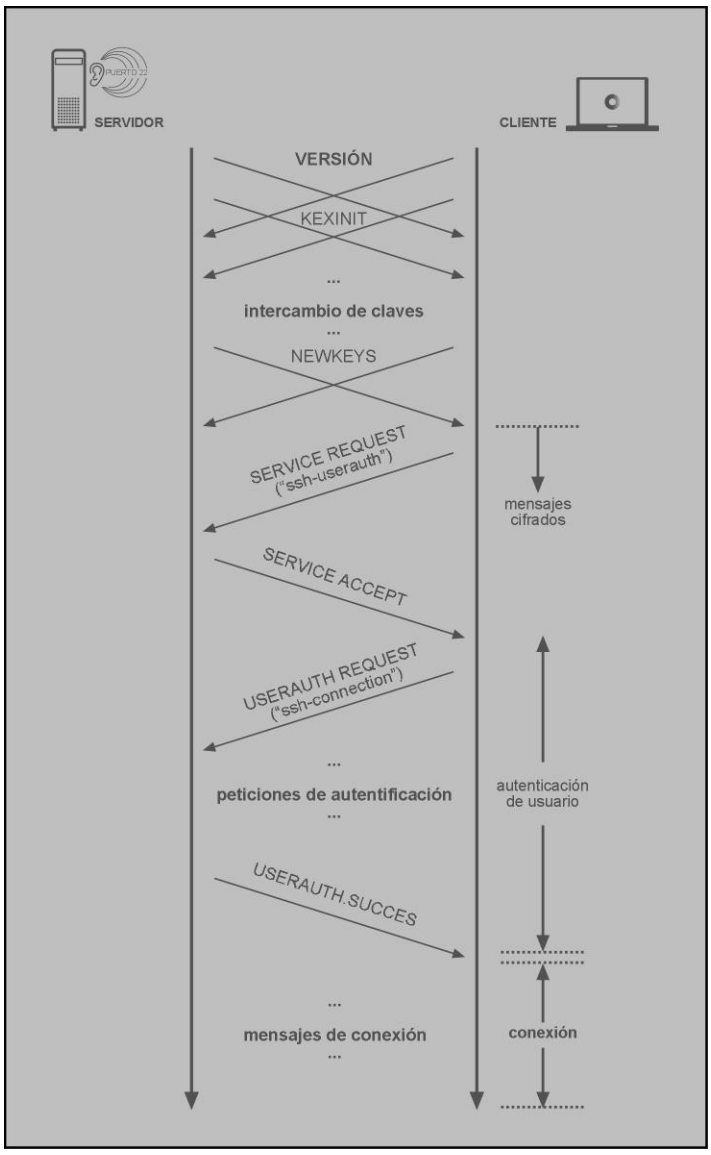


*Figura 27 Intercambio de mensajes en SSH2*

#### 4.4.3.7 Ataques contra SSH

Este protocolo está diseñado para que un atacante no pueda leer el contenido de los mensajes, ni alterarlos ni tampoco cambiar la secuencia de los mismos.
La confidencialidad queda garantizada con el método de intercambio de claves basado en criptografía de clave pública, que protege contra los ataques denominados como “maní in i.e. miadle”.

Este método permite que el cliente se asegure de que se está conectando al servidor auténtico. Para comprobar que la clave pública que envía el servidor es realmente la suya, se pueden usar certificados, o bien una base de datos local del cliente en la que estén guardadas las claves de los servidores reconocidos. Y para autenticar al usuario mediante una clave pública (la suya o la del cliente desde el cual se conecta, dependiendo del método de autenticación), también existen las dos opciones: certificados o una base de datos de claves en el servidor.

El protocolo SSH1 era vulnerable a ataques de repetición, eliminación o re-ordenación de paquetes porque no utilizaba números de secuencia, y también al reenvío de paquetes en sentido contrario si se utilizaba una sola clave de cifrado para ambos sentidos. Estos problemas ya no están presentes en SSH2.

Una característica interesante añadida a SSH2 es que las longitudes de los paquetes se envían cifradas. Un atacante que vea los datos intercambiados como un flujo de bytes no puede saber dónde empieza y dónde acaba cada paquete SSH2 (si tiene acceso al nivel de paquetes TCP puede intentar hacer deducciones, pero sin una certeza absoluta). Esto, juntamente con la posibilidad de incluir padding arbitrario (hasta 255 bytes) y enviar mensajes IGNORE, puede servir para ocultar las características del tráfico y dificultar los ataques con texto claro conocido.
El protocolo SSH está diseñado para ofrecer determinadas protecciones, pero el nivel de seguridad que proporcione en cada caso vendrá dado por las implementaciones y por el uso que se haga del mismo. Se pueden deshabilitar o restringir las características (métodos de autenticación de usuario, redirección de puertos TCP, etc.) que en un determinado entorno impliquen alguna vulnerabilidad o posibilidad de abuso.

#### 4.4.3.8 Herramientas

##### 4.4.3.8.1 OpenSSH

OpenSSH es una versión libre del paquete de herramientas de comunicación segura del protocolo SSH/Sex para redes, una solución de seguridad que tiene la confianza de un número cada vez mayor de usuarios de Internet. Muchos usuarios de telnet, rlogin, TFP y otros programas parecidos, no se dan cuenta que sus contraseñas se están transmitiendo sin cifrar a través de la red. Pones cifra todo el tráfico (incluidas las contraseñas) para eliminar de un modo efectivo las «escuchas», los secuestros de las conexiones y otros ataques a nivel de red. Además, Pones ofrece amplias posibilidades para la creación de túneles seguros, aparte de una variedad de métodos de autenticación.

SSH es un protocolo que permite establecer conexiones seguras a través de redes que no lo son, además es capaz de servir de túnel seguro para otros protocolos que no lo son. Podemos entonces realizar tareas de mantenimientos de sistemas y conexiones remotas al estilo UNIX de forma segura.

SSH2, la segunda versión de SSH, resuelve algunas de las deficiencias de su antecesor SSH1, ofreciendo de esta manera un alto nivel de cifrado de datos y un método de autentificación bastante fiable. Es además una alternativa fiable a los no seguros: telnet o rlogin, rsh, rcp, dist.

Algunas de las principales características:

- Protocolo criptográfico en un modelo cliente/servidor
- Autenticación de las más variadas formas:
  - por contraseña
  - por host
  - por sistema de llaves
- Integración con sistemas de autenticación como:
  - Cerberos
  - Escurrid
  - PGP
  - TIS Gauntlet
  - PAM

    - Integración con sistemas de autenticación como:
- Seguriza los protocolos de aplicación de manera casi transparente
- Implementación sobre la mayoría de los sistemas operativos y plataformas

SSH previene, además, de una serie de ataques como los procedentes de Sniffers:

- IP Spoofing
- MACpoofing
- DNS Spoofing
- Telnet Hickjacking
- ARP Spoofing
- IP Routing Spoofing
- ICMP Spoofing

#### 4.4.3.8.2 Putty

PuTTY es un cliente de software libre SSH, Telnet, rlogin, y TCP raw. Estaba disponible originariamente solo para Windows, pero también está disponible en varias plataformas Unix, con trabajo en desarrollo para puertos al Mac OS clásico y Mac OS X. Esto es un software beta escrito y mantenido principalmente por Simon Tatham, y es open source, licenciado bajo la Licencia MIT.

Algunas características de PuTTY son:

- El almacenamiento de hosts y preferencias para uso posterior.
- Control sobre la clave de encriptación SSH y la versión de protocolo.
- Clientes de línea de comandos SCP y SFTP, llamados "pscp" y "psftp" respectivamente.
- Control sobre el reenvío de puerto con SSH, incluyendo manejo empotrado de reenvío X11.
- Completos emuladores de terminal xterm, VT102, y ECMA-48.
- Soporte IPv6.
- Soporte 3DES, AES, RC4, Blowfish, DES.
- Soporte de autenticación de clave pública.

- Soporte para conexiones de puerto serie locales.

#### 4.4.3.9 Instalación y configuración de SSH

Se va a realizar lo siguiente:

- Instalar y configurar un servidor SSH en la máquina Linux. El paquete a utilizar será OpenSSH.
- Instalar y configurar un cliente SSH en el PC con sistema operativo Windows Xp. La aplicación de libre distribución Putty es la utilizada para realizar esta parte.

- Comprobación del correcto funcionamiento de la configuración realizando conexiones SSH a la máquina servidor. Configurar otro servicio, como por ejemplo un servidor DNS corporativo, desde la máquina cliente al servidor Linux.

Se procede a la instalación del paquete OpenSSh. Hay que instalar y configurar un servidor SSH en la máquina Linux.

En el equipo con nombre: PC1PFC se entrará con el usuario plano: javipc1 y se activará el usuario privilegiado root. Se abre un Terminal y se introducen los siguientes comandos para la instalación del paquete:

```
apt-get install openssh-server
```

Una vez instalado hay que comprobar que el demonio ssh está escuchando por el puerto 22 intentando conectar con la propia máquina mediante la Ip de loopback: 127.0.0.1 .

Se obtiene lo siguiente:

```
javipc1@PC1PFC:~$ ssh 127.0.0.1
The authenticity of host ‘127.0.0.1 (127.0.0.1)’ can’t be established.
RSA key fingerprint is
49:28:a6:6d:63:e8:bb:dd:e3:25:ff:56:3d:ac:7b:b1.
Are you sure you want to continue connecting (yes/no)?
```

*Figura 28 Comprobación de escucha del puerto 22 SSH*

Se le indica: *‘yes’*

```
Warning: Permanently added '127.0.0.1' (RSA) to the list of known
hosts.
javipc1@127.0.0.1’s password:
```

*Figura 29 Autenticación del usuario.*

Y se introduce nuestro password de usuario *javipc1:*

```
Linux PC1PFC 2.6.17-10-generic #2 SMP Tue Dec 5 22:28:26 UTC 2006 i686
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
Last login: Sun Apr 30 16:05:30 2007
javipc1@PC!PFC:~$
```

*Figura 30 Intercambio de claves.*

Al ser la primera conexión con el servidor, verifica dicha conexión ya que es un host desconocido. Ahora ya es un host conocido y se guarda la entrada de la clave en el directorio HOME.

Si se lista el contenido del fichero sale:

```
javipc1@PC1PFC:~/.ssh$ pwd
/home/javipc1/.ssh
javipc1@PC1PFC:~/.ssh$ ls
known_hosts
javipc1@PC1PFC:~/.ssh$ cat known_hosts
|1|Z4vAldjZlyCWX7qXzzTzi+PdRnw=|BhNX64JRBnggzBqga6Q2+nkiy0Q= ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEA6Rm8mIRRmgiP5KIOsS68gTw9auscOi40fhyKyOnmgX
TMZZFSv1GGThHfzSHdacLXfzAJiFOqWtCoH2CTHolsQOV5LcNcljTfWrAz4SwkZEBN5yKQ
lh9pAxStP2oJSXr6QyWgKI//cSIimzSOQonH8dw5Fxvt913D4Yq7MtPOsPrLIV/+FstfxE
XJ00wjJ69s0ScUxipzPFvJ4fsSf8A/F2fdMMQw8McbHar4a9bDcs+s1FLJ09iPn5MsSArn
X97fCRpSdzACrj4+ihTeR0fqKDUyyfukNTEMZx5QD5aoHqnciDsIu9nLXEYQWKqAkTlPG+
wd9bI3kyI303v5BINbrQ==
javipc1@PC1PFC:~/.ssh$
```

*Figura 31 Directorio HOME, hosts conocidos.*

Las claves se guardan en un fichero denominado "*know_hosts",* la próxima vez que se conecte con una máquina con la que ya había conectado anteriormente si su entrada está en este fichero no volverá a preguntar.

Una vez instalado y configurado el servidor SSH procedemos a instalar el cliente.

Utilizamos la aplicación ya comentada *Putty.* Es instalación típica (siguiente…) y se establece una sesión SSH (puerto 22, no usamos ni telnet ni las demás opciones que

nos da) con la máquina servidor. Hay que introducir la IP que tiene el servidor, host al que nos queremos conectar: 192.168.1.33.

Se podría también establecer la conexión introduciendo el nombre de PC que se definirá en el servidor DNS. Escribiendo dicho nombre, pc1pfc.homeip.net, el servidor DNS es capaz de resolverlo y asociarlo a la IP de la máquina con el servidor SSH. Esta parte se verá en el siguiente capítulo.

Pedirá el login y el password. Se puede entrar como usuario plano, javipc1, o como administrador/usuario con privilegios root. Cuando se hayan introducido el nombre de usuario y la contraseña correctamente estará conectado y podrá trabajar en el servidor igual que si se estuviera delante de la misma.

Con esto se habrá instalado y configurado un servidor SSH y comprobado su correcto funcionamiento estableciendo una conexión, segura y encriptada, desde una máquina cliente y trabajando desde esta última sobre la remota, Server SSH.

#### 4.4.3.10 SFTP, transmisión segura de ficheros

SFTP es una aplicación incluida dentro del paquete OpenSSH y tiene la misma función que el conocido FTP (file transfer protocol) pero con la ventaja, al igual que el SSH respecto al Telnet, que la información viaja encriptada. Puede tener el inconveniente de ser un poco más lento que el ftp ya que los datos enviados tienen que encriptarse previamente y ser desencriptados en el destino para hacer uso de ellos. Una desventaja paliada por la seguridad que nos da trabajar con esta herramienta.

El servidor FTP en este caso no es un demonio independiente sino que constituye un subsistema del servidor principal del SSH. Se habilita a través de un atributo en la configuración del demonio SSH en la ruta:

```
/etc/ssh/sshd_config
```

Añadiendo una línea de la forma:

```
Subsystem sftp /usr/lib/ssh/sftp-server
```

Se conectará mediante el cliente sftp de Putty. Se usará el usuario root para tener acceso a todas las carpetas del servidor. Se accede al directorio que contenga el archivo que se quiera transferir y se introduce el comando get:

```
get ‘nombredelarchivo.extension”
```

El archivo aparecerá en la carpeta en la que se encuentre Putty. Con esto se habrá comprobado que tanto SSH como SFTP funcionan correctamente.

### 4.4.4. Domain Name Server

#### 4.4.4.1 Introducción

El sistema Domain Name Server no es mas que el encargado de traducir los nombres que nosotros tecleamos en nuestro navegador (www.loquesea.com), por la dirección de red real que tienen (IP) para que así podamos llegar hasta dicha página a lo largo de la red. Si nuestro servidor DNS no sabe a quien corresponde el nombre que hemos introducido (no lo tiene definido en él, no es el servidor autoritario y tampoco lo tiene en la memoria caché) le preguntará al servidor DNS que tenga como 'forwarder' y así sucesivamente hasta encontrar la respuesta.  En la siguiente figura vemos el proceso:

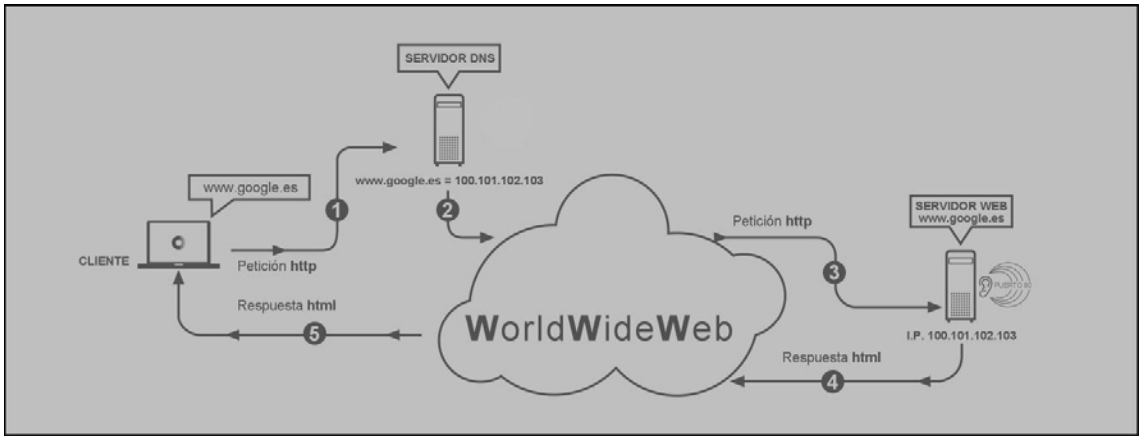


*Figura 32  Proceso de consulta de una página web.*

#### 4.4.4.2 Historia del DNS

A mediados de los 70, la *ARPANET* era una comunidad pequeña y amistosa de unas cientos de máquinas. Un solo archivo, *HOSTS.TXT*, contenía toda la información que se necesitaba saber sobre esas máquinas: contenía un mapeo de nombre a dirección para cada máquina en la *ARPANET*.

El archivo */etc/hosts* se obtenía entonces del archivo HOSTS.TXT (el cual tenía información adicional que no era necesaria).

Sin embargo, cuando la ARPANET se cambió a los protocolos TCP/IP, la población de la red explotó, y con ello se presentaron los siguientes problemas:

• **La carga y el tráfico** de red para la máquina que contenía las tablas que hacían posible el mapeo de nombres a direcciones IP se volvió inmanejable.

• **Colisiones de nombres**. El NIC (organismo encargado de administrar la creación de nombres) no podía garantizar que alguien asignara el mismo nombre a máquinas distintas (Esto se debe a que con el método del HOSTS.TXT se tenía un dominio plano, es decir, sin jerarquías).

• **Consistencia***:* Mantener la consistencia del archivo a lo largo de una red en crecimiento se hacía cada vez más difícil (Por ejemplo, cuando el archivo HOSTS.TXT llegaba a una máquina muy lejana ya era obsoleto).

Este método se volvía ineficiente a medida que aumentaban el número de máquinas. Es allí donde entra DNS (*Domain Name System,* Sistema de Dominios de Nombres). DNS es una base de datos distribuida. DNS permite un control local sobre los segmentos de la base de datos general, logrando que cada segmento esté disponible a lo largo de toda la red utilizando un esquema cliente servidor. La robustez y un desempeño adecuado se logran gracias a la duplicidad de servidores y el *caching* (almacenamiento temporal en la memoria).

Los programas llamados servidores de nombres (name servers) comprenden la mitad del mecanismo cliente - servidor de DNS. Los servidores de nombres contienen información acerca de un segmento de la base de datos y la ponen a disposición de los clientes, llamados resolvers.

Los resolvers son solo bibliotecas de rutinas que crean preguntas y las envían a lo largo de la red hacia el servidor de nombres.

La estructura de una base de datos DNS se asemeja mucho a un árbol invertido (un árbol cuyo tronco esta hacia arriba y no abajo, como es común). Cada "hoja" o "nodo" de ese árbol es un dominio, comenzando todos los dominios desde el dominio root (el cual se denota con un punto "."). Cada uno de esos dominios a su vez pueden subdividirse en más sub-dominios. Existen muchos dominios (com.  edu. es.) y debajo de ellos hay aun más sub-dominios.

Con un ejemplo se ve mas claro: Una máquina llamada bach dentro del dominio ing.ula.ve. tiene el nombre unívoco (llamado oficialmente de dominio completamente calificado) bach.ing.ula.ve (FQDN, Full Qualified Domain Name) debido a que incluye tanto el nombre de la máquina como al dominio al cual pertenece. La dirección completa de la máquina se lee de izquierda a derecha (Desde lo más específico, el nombre del host, pasando por cada uno de los "dominios" a los cuales pertenecen).

Cada máquina en la red pertenece a un dominio, cuyo servidor de nombres contiene la información acerca de la máquina. Esta información puede incluir direcciones IP, información acerca de enrutamiento de correo, etc. (Una máquina también puede tener uno o más aliases de dominio, lo cual quiere decir que existen 2 referencias hacia la máquina, una de ellas es un apuntador de un dominio *(alias)* a su *nombre canónico* (u oficial).

DNS con su estructura (aparentemente complicada) permite eliminar los problemas que presentaba la existencia de un archivo de datos planos:

- Elimina el problema de nombres repetidos (a cada organización se le asigna un dominio único, por lo que pueden existir dos máquinas con el mismo nombre mientras estén en dominios separados).
- Elimina el problema de carga y tráfico de red en una sola máquina ya que la información esta distribuida. (Y está disponible de manera redundante).
- Finalmente hay consistencia, ya que la actualización de la información se hace de manera automática, sin intervención del administrador de la red (al menos no de la manera tradicional).

#### 4.4.4.3 Servidores de Nombres (*Name Servers*)

Las máquinas que guardan información acerca de un espacio de dominio son llamadas servidores de nombres. Los servidores, generalmente tienen información completa acerca de una parte del espacio de dominio de nombres, llamado zona. El servidor de nombres entonces se dice que tiene autoridad para esa zona (Los servidores de nombres pueden ser autoridades para más de una zona).

La diferencia entre una zona y un dominio es que la zona contiene los nombres de dominio y datos que representan a un dominio. Un dominio es un nombre que agrupa a otras máquinas o dominios inferiores.

Las especificaciones de DNS definen 2 tipos de servidores: maestros primarios y maestros secundarios. Un servidor primario maestro obtiene los datos de las zonas sobre la cual tiene autoridad desde archivos que están en la misma máquina que el servidor de nombres, mientras que un maestro secundario realiza una transferencia de zona (y sigue siendo autoridad para la zona).

#### 4.4.4.4 Dominio

Un dominio es sólo un índice dentro de la base de datos de DNS. Un dominio puede ser una máquina o puede ser un nodo del cual pueden partir otros dominios (o ambas cosas a la vez).

El siguiente árbol invertido me muestra un dominio típico de DNS:

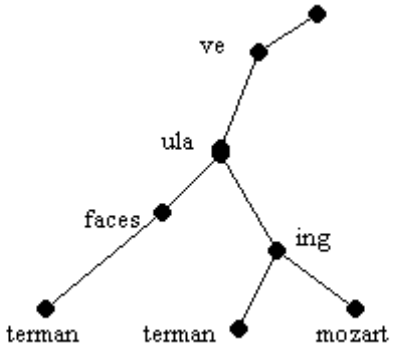


*Figura 33 Dominio DNS*

Cada unidad de datos en DNS está indexada por nombre (hay que recordar que es una base de datos). Estos nombres son esencialmente rutas en un gran árbol invertido, llamado el espacio de dominio de nombres. El árbol de DNS puede ramificarse de diversas maneras en cada punto de intersección, llamado nodo. Cada nodo puede tener hasta 63 caracteres de longitud. El dominio raíz tiene una etiqueta (de tamaño cero), la cual es reservada. El nombre completo de un nombre de dominio es una secuencia de etiquetas en la ruta desde ese nodo hasta la raíz. Cuando el dominio raíz aparece por si mismo es denominado por un punto. De manera que cuando alguien escribe una dirección terminada con un punto ésta es interpretada como una dirección absoluta.

En el gráfico vemos que hay dos nodos debajo del nodo ing. en el árbol invertido: Herman y mozart. Terman y Mozart son dos máquinas las cuales pertenecen al dominio ing.ula.ve. (Note el punto al final del dominio). Es decir, sus nombres completamente calificados son terman.ing.ula.ve. y mozart.ing.ula.ve. Vemos también como es posible con DNS tener dos máquinas con el mismo nombre pero en dominios diferentes (terman.ing.ula.ve y terman.faces.ula.ve).

#### 4.4.4.4.1 División de dominios

Los dominios en Internet están divididos de dos maneras:

**Dominios genéricos*:***

Son conocidos como dominios internacionales u organizacionales. Son organizados conceptualmente y son los dominios básicos de Internet.
Son los creados inicialmente por el *InterNIC,* como los .com (comerciales), .edu (universidades, instituciones educativas), .net (organizaciones relacionadas con la red) etc.

**Dominios geográficos*:***

Son conocidos como geográficos o *ISO3166*. Son mantenidos por cada país y territorio en el mundo. Estos dominios son organizados por localidad y son útiles para organizaciones y negocios que desean operar en otros países o que quieren proteger su compañía.

#### 4.4.4.4.2 Delegación de Dominios

Una de las metas de DNS es la descentralización de la administración. Esto se logra utilizando delegación de dominios. Delegación significa repartir tareas y responsabilidades de un subdominio a un grupo de personas u organización distinta a la que controla el dominio principal.

Lo que ocurre en realidad es la asignación de autoridad de sus subdominios a diferentes servidores de nombres (Estas son las máquinas que corren una implementación de DNS, por lo general *BIND)*. En sus archivos de datos, en vez de contener información sobre un subdominio, se incluye un puntero a servidores de nombres que son autoridad para ese subdominio. De esa manera, si el servidor de nombres de un dominio superior (como ula.ve) es interrogado acerca de una máquina perteneciente a un subdominio (digamos terman.ing.ula.ve) devolverá la dirección (o direcciones) de servidores que pueden darle esa información (mozart.ing.ula.ve, por ejemplo) en vez de tratar de responder el mismo.

#### 4.4.4.5 Registro de Recursos (RR)

Los datos asociados con nombres de dominios deben ser guardados de alguna manera en los servidores de nombres. La forma utilizada por Bind es guardarlos en archivos de texto simple con etiquetas llamadas registros de recursos (*Resource Records* o RR).

Cada clase de registro pertenece a un tipo de red o software. Actualmente hay clases para Internet (y cualquier red basada en *TCP/IP*), redes basadas en el Protocolo *Caosnet* (Red vieja, de importancia histórica) y redes que usan el protocolo *hesiod.* En este proyecto (ya que es así en la vida real) solamente trabajaremos con registros tipo *IN (Internet).*

Aparte de estos registros existen otros registros que guardan información útil sobre las máquinas pertenecientes a un dominio. Algunos de éstos son:

- NS (*Name Server*, especifican qué máquinas son servidores de nombres)
- MX (*Mail Exchangers*, especifican qué máquinas intercambian correos)
- PTR (*Pointer*, permiten la conversión de una dirección IP a nombre)
- A (*Address*, permiten la conversión de un nombre a dirección IP)
- CNAME (*Canonical Name*, se utilizan para hacer un alias).

La forma como se cargan los datos también depende mucho del servidor. Si el servidor es un *servidor maestro* guarda su información en archivos de bases de datos (*database files*).

Los archivos de datos contienen registros fuentes que describen la zona y manejan la delegación de subdominios.

En el caso de un servidor secundario estos tienen sólo los archivos de configuración básicos (para la red Loopback y las pistas de los Root Name Servers) y toman sus datos del servidor primario (Es decir hacen una transferencia de zona).

#### 4.4.4.6 Resolvers

Es un conjunto de bibliotecas de las aplicaciones clientes (es decir aquellas que solicitan información acerca de un espacio de dominios de nombres).

Un *resolver* tiene como tareas:

- Interrogar al servidor de nombres
- Interpretar respuestas (Que pueden ser registros RR o errores)
- Devolver información al programa que la solicita.

#### 4.4.4.7 ¿Cómo trabaja?

El protocolo DNS trabaja en la capa de aplicación. Si el segmento a enviar es menor que 512 Bytes utiliza el protocolo UDP, de lo contrario utiliza el protocolo TCP. El número de puerto que utiliza el protocolo DNS para comunicarse con la capa de aplicación es el número 53.

##### 4.4.4.7.1 La cabecera

Todos los mensajes generados por el protocolo DNS utilizan un único formato de cabecera el cual se muestra en la siguiente figura.

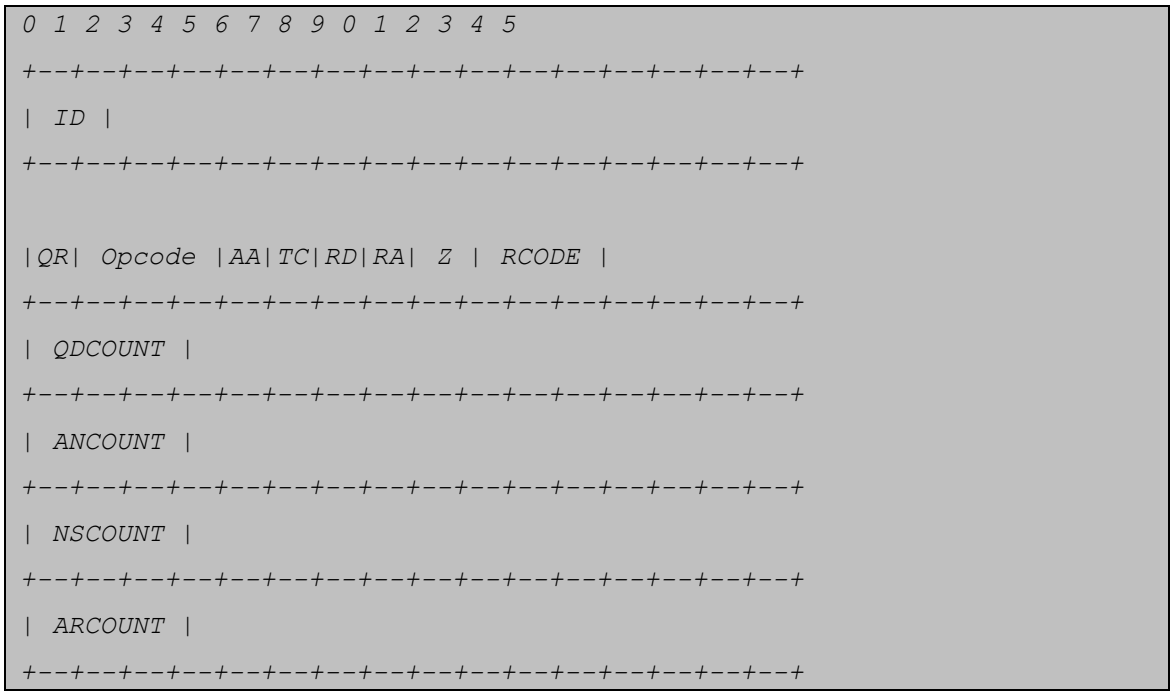


*Figura 34 Cabecera del protocolo DNS*

-. ID. Es un identificador de 16 bits asignado por el programa. Este identificador se copia en la respuesta correspondiente del servidor de nombres y se puede usar para diferenciar respuestas cuando concurren múltiples consultas.

-. QR. Flag que indica consulta (0) o respuesta (1).

-. AP code. Campo de 4-bit que especifica el tipo de consulta:

-. 0 consulta estándar (QUERY).

-. 1 consulta inversa (IQUERY).

-. 2 solicitud del estado del servidor (STATUS).

-. Se reservan los otros valores para su uso en el futuro.

-. AA. Flag de respuesta autoritativa. Si está activo es una respuesta, especifica que el servidor de nombres que responde tiene autoridad para el nombre de dominio enviado en la consulta.

-. TC. Flag de truncado. Activo si el mensaje es más largo de lo que permite la línea de transmisión.

-. RD. Flag de recursividad. Este bit se copia en la respuesta e indica al servidor de nombres una resolución recursiva.

-. RA. Flag de recursividad disponible. Indica si el servidor de nombres soporta resolución recursiva.

-. Z. 3 bits reservados para uso futuro. Deben ser cero.

-. Rcode. Código de respuesta de 4 bits. Los posibles valores son:

-. 0. Ningún error.

-. 1. Error de formato. El servidor fue incapaz de interpretar el mensaje.

-. 2. Fallo en el servidor. El mensaje no fue procesado debido a un problema con el servidor.

-. 3. Error en nombre. El nombre de dominio de la consulta no existe. Sólo válido si el bit AA está activo en la respuesta.

-. 4. No implementado. El tipo solicitado de consulta no está implementado en el servidor de nombres.

-. 5. Rechazado. El servidor rechaza responder por razones políticas. Los demás valores se reservan para su usuario en el futuro.

-. QDcount. Un entero sin signo de 16 bits que especifica el número de entradas en la sección "question"

-. ANcount. Un entero sin signo de 16 bits que especifica el número de RRs en la sección "answer".

-. NScount. Un entero sin signo de 16 bits que especifica el número de RRs en la sección "authority".

-. ARcount. Un entero sin signo de 16 bits que especifica el número de RRs en la sección "additional records".

Los servidores de nombres tienden a buscar datos de un espacio de dominio de nombre. Tienen que comportarse de esa manera, dada la poca inteligencia del *resolver*. No sólo pueden dar datos acerca de zonas de la que tienen autoridad sino que pueden buscar a lo largo de un espacio de dominio para encontrar datos sobre los que no tienen autoridad (a esto se le conoce como *resolución*).

La resolución comienza siempre desde los servidores de dominios superiores hasta llegar al servidor que tiene la información autorizada de un dominio en particular (Es decir comienza desde la parte superior del árbol invertido hasta llegar a la rama buscada).

En el proceso de resolución de nombres podemos identificar algunos elementos que veremos en los siguientes apartados.

#### 4.4.4.7.2 RNS Servidores de Raíz Primarios (Root Name Servers)

Los RNS saben que servidores de nombres tienen autoridad para los dominios superiores. Si se les hace una pregunta acerca de un subdominio, los servidores raíz maestros pueden al menos proveer los nombres y direcciones de los servidores de nombres con autoridad para el segundo nivel de dominios a los cuales un dominio pertenece. Cada servidor interrogado da al que pregunta información de cómo “estar más cerca” de la respuesta que está buscando o provee él mismo una respuesta. Lo que hacen los RNS es proveer punteros desde los dominios superiores a los servidores de nombres de los dominios inferiores.

Los RNS, así como los NS normales, son muy importantes en la resolución de un nombre dentro de un dominio particular. Debido a que son tan importantes, DNS provee mecanismos para asegurar siempre el servicio, utilizando redundancia (servidores secundarios) o aliviando la carga de los servidores primarios y root

(usando *caching*). Sin embargo, en ausencia de mecanismos como el *caching*, la resolución debe empezar en los servidores de raíz maestros.

#### 4.4.4.8 Métodos de búsqueda

Primero el servidor de nombres verifica sus tablas de máquinas a ver si allí consigue el nombre por el cual le están preguntando. Si es así, entonces retorna la dirección IP asociada con ese nombre. Si la información pertenece a otro dominio, entonces el servidor de nombres busca en su caché y si no está allí entonces comienza el proceso de resolución que se puede comportar de estas dos formas:

##### 4.4.4.8.1 Recursiva

Un servidor de nombres envía una respuesta recursiva cuando es el servidor y no el cliente el que pregunta a otros servidores de nombres por la información del dominio solicitada. Esto ocurre cuando el servidor de nombres sabe que el resolver no tiene la inteligencia de manejar una referencia a otro servidor de nombres (Es el resolver hace explícitamente una pregunta recursiva). A medida que un servidor de nombres pregunta (obtenga respuesta o no) va guardando los nombres encontrados en su caché para evitarse búsquedas innecesarias.

##### 4.4.4.8.2 Iterativa

El servidor de nombres da la mejor respuesta que ya sabe a quien preguntó (da una referencia al servidor de nombres más cercano a la información de dominio interrogado). Primero consulta sus datos locales, si no está allí busca entonces en su caché y si aún no encuentra nada entonces devuelve la respuesta al servidor más cercano al dominio buscado.

Las bibliotecas del resolver hacen búsquedas recursivas e iterativas, mientras que entre servidores de nombres solo se hacen búsquedas iterativas.

##### 4.4.4.8.3 Caching

Podría parecer que el proceso de buscar un nombre es sumamente lento, sin embargo no es así. Una de las razones de la rapidez es el uso de caching.

Este mecanismo trabaja de la siguiente manera: Un servidor de nombres procesando una búsqueda recursiva podría enviar unas cuantas preguntas para encontrar una

respuesta acerca de un dominio. Sin embargo, el servidor descubre información acerca del nombre del dominio a medida que explora. Cada vez que es referido a otro servidor, aprende que esos servidores son autoridades de las zonas interrogadas, y aprende esas direcciones. Si encuentra el dato buscado, lo guarda para usarlo en una futura referencia. La próxima vez que un resolver haga una pregunta acerca de un nombre de un dominio que el servidor conozca, el proceso es acortado un poco, ya que el servidor primero revisará en su caché para dar la respuesta.

El caché solamente se guarda en memoria temporal la cual se borra cuando el servidor re-actualiza su memoria (por ejemplo, cuando se apaga la máquina en la que corre el servidor).

#### 4.4.4.8.4 TTL, Tiempo de vida (Time to Live)

Estos tiempos son los que le dicen a un servidor secundario cuanto tiempo debe mantener en memoria sus datos antes de buscar datos actualizados del servidor maestro.

Los servidores de nombres no mantienen los datos en caché por siempre (De ser así, las modificaciones hechas en un servidor maestro nunca se propagarían por la red). De esta manera, el administrador de una zona decide el tiempo de vida de los datos buscando un balance entre la veracidad de la información y la cantidad de tiempo perdido transfiriendo una zona. Una vez que el tiempo de vida expira, el servidor busca de nuevo los datos del dominio del cual es servidor secundario.

### 4.4.4.9 Instalación y configuración del servidor DNS. BIND9

Los objetivos de esta parte serán:

- Instalar un servidor DNS corporativo en una LAN sobre una máquina con sistema operativo Ubuntu Linux. Herramienta a utilizar, BIND9.
- Realizar la instalación remotamente, desde un PC cliente, utilizando el servicio SSH configurado anteriormente.
- Comprobar que nuestro servidor DNS resuelve correctamente dominios pertenecientes a dicha LAN como cualquier otro dominio de Internet de los cuales no somos autoritarios (peje. www.google.com).

La realización de esta parte se hará remotamente utilizando el servicio SSH instalado y configurado anteriormente. Se arranca el ordenador cliente con sistema operativo

WXp y se conecta al PC servidor con sistema operativo Ubuntu Linux mediante SSH con la aplicación Putty y se accede con el usuario privilegiado root. A partir de ahí se tienen todos los medios para realizar la instalación y la configuración del servidor DNS sin estar físicamente frente a él.

#### 4.4.4.9.1 Instalación de BIND9

BIND (*Berkeley Internet Name Domain*, anteriormente: *Berkeley Internet Name Daemon*) es el servidor de DNS más comúnmente usado en Internet, especialmente en sistemas UNIX. Fue creado originalmente por cuatro estudiantes de grado en la Universita Of. California, Berkeley.

Una nueva versión de BIND (BIND 9) fue escrita desde cero en parte para superar las dificultades arquitectónicas presentes anteriormente para auditar el código en las primeras versiones de BIND, y también para incorporar DNSSEC (DNS Security Extensions). BIND 9, que es el que utilizaremos, incluye entre otras características importantes: TSIG, notificación DNS, nsupdate, IPv6, rndc flush, vistas, procesamiento en paralelo, y una arquitectura mejorada en cuanto a portabilidad. Es comúnmente usado en sistemas Linux.

Existen cuatro tipos diferentes de servidores de resolución de nombres:

- **Master** (maestro o primario). Aloja los registros autoritarios de una zona, responde las peticiones de resolución de nombres como servidor de autoridad y delega copias a los servidores esclavo.
- **Slave** (esclavo o secundario). Responde a las peticiones de resolución de nombres como servidor de autoridad, pero la información es distribuida por los servidores primarios. Se considera que como medida de seguridad, se requiere al menos uno de estos, preferentemente independiente de la infraestructura del primario (red, energía eléctrica y ubicación geográfica).
- **Caching-only** (sólo de caché). Responde a las peticiones de resolución de nombres pero no es servidor de autoridad, las respuestas las guarda en memoria por un período determinado.
- **Forwarding** (de reenvío). Reenvía las peticiones a una lista de servidores de nombres.

#### 4.4.4.9.2 Tipos de registros

Para ofrecer suficiente flexibilidad en la configuración, se pueden declarar diversos tipos de registros, que hacen referencia a la función del host. Los más importantes son los siguientes.

- A (Address). Es el registro más usado, que define una dirección IP y el nombre asignado al host. Generalmente existen varios en un dominio.
- MX (Mail eXchanger). Se usa para identificar servidores de correo, se pueden definir dos o más servidores de correo para un dominio, siendo que el orden implica su prioridad. Debe haber al menos uno para un dominio.
- CNAME (Canonical Name). Es un alias que se asigna a un host que tiene una dirección IP valida y que responde a diversos nombres. Pueden declararse varios para un host.
- NS (Name Server). Define los servidores de nombre principales de un dominio. Debe haber al menos uno y pueden declararse varios para un dominio.
- SOA (Start Of Authority). Este es el primer registro de la zona y sólo puede haber uno en cada archivo de la zona y sólo está presente si el servidor es autoritario del dominio. Especifica el servidor DNS primario del dominio, la cuenta de correo del administrador y tiempo de refresco de los servidores secundarios

Lo primero es instalarlo en el sistema Linux para después ver como configurar BIND9 para disponer de un servidor DNS en una intranet que sea capaz de resolver tanto dominios internos de los que sea autoritario, como cualquier otro dominio de internet. Se utilizará uno de los siguientes comandos, cualquiera de los dos servirá:

```
aptitude install bind9
```

```
apt-get install bind9
```

Después de instalarlo se da paso a la configuración. En el directorio /etc/bind se encuentra todo lo necesario para configurar esta utilidad. Hay varios archivos, pero los más interesantes son:

- *named.conf*
- *named.conf.options*
- *named.conf.local*

La primera parte de la configuración va ser la relacionada con la resolución de los dominios de Internet, es decir, configurar lo que va a permitir navegar por la red.

Para esto se tiene que configurar el archivo *named.conf.options*. Se definen los servidores que van actuar de lo que se denomina como 'forwarders'. Se indica que servidores de DNS debe usar para poder resolver los dominios que no conoce. Dicho de otra manera, para resolver los dominios que el servidor no tiene definidos necesita 'preguntarle' a otros servidores DNS para dar una respuesta a esa petición. Esos son los llamados forwarders.

El proveedor de servicio que hay en la red local es Telefónica, así que se define en el *named.conf.options* el que es su DNS primario, IP: 80.58.61.250, como el DNS forwarder:

```
Forwarders{
 80.58.61.250;
};
```

*Figura 35 Configuración de los DNS 'forwarders'.*

De ésta manera Bind será capaz de resolver y cachear cualquier dominio de internet que se le pida.

La siguiente parte será configurarlo para que resuelva los dominios propios de la LAN. Se definirán dos PC's dentro del dominio *homeip.net* que es como va a llamarse.

*pc1pfc* será el nombre de la máquina servidor y *pc2pfc* el del cliente.

Para definir el dominio y los hosts que hay dentro de él se tiene que editar el fichero *named.conf.local.* En el directorio *bind* que está a su vez en /etc se encuentra dicho fichero:

*cd /etc/bind*

Se edita named.conf.local y se añade la zona "pc1pfc.homeip.net", haciendo referencia a su fichero de configuración:

```
zone "pc1pfc.homeip.net" {

    type master;
    file "/etc/bind/db.pc1pfc";
}
```

*Figura 36 Configuración registro DNS en BIND 9*

Creamos el fichero de configuración “db.pc1pfc” a partir del ya existente “db.local”:

```
cp db.local db.pc1pfc
```

Se edita “db.pc1pfc”, reemplazando la palabra “localhost” por “pc1pfc.homeip.net”, se cambia la IP “127.0.0.1″ por la que se quiera asignar al dominio, en este caso la 192.168.1.33 y se añade al final del fichero todos los A, MX y CNAME que se desee, quedando:

```
; BIND data file for local loopback interface
;
$TTL    604800
@   IN  SOA pc1pfc.homeip.net. root.pc1pfc.homeip.net. (
                  1     ; Serial
             604800     ; Refresh
              86400     ; Retry
            2419200     ; Expire
             604800 )   ; Negative Cache TTL
;
@           IN      NS          pc1pfc.homeip.net.
@           IN      A           192.168.1.33
@           IN      MX      0   pc1pfc.homeip.net.
www         IN      A           192.168.1.33
pcserver    IN      CNAME       pc1pfc.homeip.net.
```

*Figura 37 Definición de los hosts.*

Hay que decir que para editar los ficheros se está utilizando, como durante todo el proyecto, el editor VI.

Se ve primeramente el dominio a resolver, ‘pc1pfc.homeip.net.’ y el segundo es la cuenta de correo del root, root.pc1pfc.homeip.net. (sustituyendo el primer punto por arroba, lo que dejaría ‘root@pc1pfc.homeip.net’). Debemos notar que al final de cada dominio viene un punto, que identifica la raíz de este. El resto de los parámetros son:

*Serial*: es un identificador del archivo, puede tener un valor arbitrario pero se recomienda que tenga la fecha con una estructura AAAA-MM-DD y un consecutivo.

*Refresco*: número de segundos que un servidor de nombres secundario debe esperar para comprobar de nuevo los valores de un registro.

*Reintentos*: número de segundos que un servidor de nombres secundario debe esperar después de un intento fallido de recuperación de datos del servidor primario.

*Expiración*: número de segundos máximo que los servidores de nombre secundarios retendrán los valores antes de expirarlos.

*TTL mínimo*: Significa Time To Live y es el número de segundos que los registros se mantienen activos en los servidores NS caché antes de volver a preguntar su valor real.

Cada vez que se cambia la configuración de BIND9, se debe reiniciar el demonio:

```
/etc/init.d/bind9 restart
```

Para que la máquina utilice el servidor de DNS que se ha configurado, se debe editar "/etc/resolv.conf" y dejar únicamente la línea:

*nameserver 127.0.0.1*

Se debe hacer lo mismo con el resto de máquinas de la intranet que vayan a utilizar el servidor, con la única diferencia que habrá que sustituir la IP 127.0.0.1 por la IP que tenga el servidor en la red, 192.168.1.33.

Para comprobar su correcto funcionamiento se puede utilizar el comando *nslookup.* Después de introducir el comando, se indica el nombre del host que se quiere saber su IP, si la respuesta es la correcta el servidor estará funcionando correctamente.

Si se desea también disponer de resolución de dominios inversa, es decir, que se pueda preguntar por la IP "192.168.1.33″ y el servidor DNS conteste que pertenece a pc1pfc.homeip.net, se deba añadir a "/etc/bind/named.conf.local":

```
zone "192.in-addr.arpa" {

    type master;

    file "/etc/bind/db.192";

};
```

*Figura 38 Definición inversa.*

Se crea el archivo de configuración "/etc/bind/db.192" a partir del fichero ya existente "/etc/bind/db.127":

```
cd /etc/bind/

cp db.127 db.192
```

Se edita "/etc/bind/db.192", se sustituye "localhost" por "pc1pfc.homeip.net" y se cambia la última línea:

```
; BIND reverse data file for local loopback interface
;
$TTL    604800
@   IN  SOA pc1pfc.homeip.net. root.pc1pfc.homeip.net. (
                  1     ; Serial
             604800     ; Refresh
              86400     ; Retry
            2419200     ; Expire
             604800 )   ; Negative Cache TTL
;
@   IN  NS  pc1pfc.homeip.net.
33.1.168   IN  PTR pc1pfc.homeip.net.
```

*Figura 39 Definición Inversa, db.192*

De forma que, la última línea indica que la IP [192.]168.1.33 (escrita a la inversa y omitiendo el 192 que ya se especificó en "named.conf.local") corresponde al dominio *pc1pfc.homeip.net*.

Se puede comprobar su funcionamiento reiniciando el demonio BIND9 y realizando una consulta con *nslookup.*

Con esto ya estará instalado, configurado y funcionando correctamente el servidor DNS.

### 4.4.5 Servidor web

#### 4.4.5.1 Introducción

Un servidor web es un programa que implementa el protocolo HTTP (hypertext transfer protocol). Este protocolo está diseñado para transferir lo que llamamos hipertextos, páginas web o páginas HTML (hypertext markup language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música.

Sin embargo, el hecho de que HTTP y HTML estén íntimamente ligados no debe dar lugar a confundir ambos términos. HTML es un formato de archivo y HTTP es un protocolo. Este protocolo está incluido en la Capa de Aplicación. La palabra servidor identifica tanto al programa como a la máquina en la que dicho programa se ejecuta.

##### 4.4.5.1.1 HTTP, HyperText Transfer Protocol

El protocolo de transferencia de hipertexto (*HTTP, HyperText Transfer Protocol*) es el protocolo usado en cada transacción de la Web (WWW). El hipertexto es el contenido de las páginas web y el protocolo de transferencia es el sistema mediante el cual se envían las peticiones de acceso a una página y la respuesta con el contenido. También sirve el protocolo para enviar información adicional en ambos sentidos, como formularios con campos de texto.

HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores. Al finalizar la transacción todos los datos se pierden. Por esto se popularizaron las cookies, que son pequeños ficheros guardados en el propio ordenador que puede leer un sitio web al establecer conexión con él y de esta forma reconocer a un visitante que ya estuvo en ese sitio anteriormente. Gracias a esta identificación, el sitio web puede almacenar gran número de información sobre cada visitante, ofreciéndole así un mejor servicio.

La versión actual de HTTP es la 1.1, y su especificación está en el documento RFC 2616. HTTP dispone de una variante cifrada mediante SSL llamada HTTPS. Es el protocolo que se usa para establecer conexiones seguras.

El protocolo HTTP está basado en el modelo cliente-servidor. Un cliente HTTP abre una conexión y envía su solicitud al servidor, el cual responderá con el recurso solicitado —si está disponible y su acceso es permitido— y la conexión se cierra.

Las solicitudes están formadas por tres campos que se separan con un espacio en blanco: "Método recurso versión-del-protocolo". Por ejemplo:

*"GET /path/to/file/index.html HTTP/1.0"*

La línea inicial de una respuesta tiene tres campos separados por un espacio: "versión-del-protocolo código-respuesta mensaje". Por ejemplo:

*"HTTP/1.0 200 OK"*

Los encabezados están normados en el protocolo, e incluyen, en el caso de una solicitud, información del navegador y eventualmente del usuario cliente; En el caso de una respuesta, información sobre el servidor y sobre el recurso. El cuerpo del mensaje contiene el recurso a transferir o el texto de un error en el caso de una respuesta. En el caso de una solicitud, puede contener parámetros de la llamada archivos enviados al servidor.

Se quiere obtener un recurso con la URL http://www.example/index.html. El diálogo http seguiría los siguientes pasos:

Se abre un socket con el host www.tuhost.example, puerto 80 que es el puerto por defecto para HTTP.

Se envía un mensaje en el estilo siguiente:

```
GET /index.html HTTP/1.0
Host: www.example.com
User-Agent: HTTPTool/1.0
[Línea en blanco]
```

*Figura 40 Mensaje del cliente*

La respuesta del servidor está formada por encabezados seguidos del recurso solicitado, en el caso de una página web:

```
HTTP/1.0 200 OK
Date: Fri, 31 Dec 2003 23:59:59 GMT
Content-Type: text/html
Content-Length: 1221

<html>
<body>
<h1>Página principal de tuHost</h1>
(Contenido)
  .
  .
  .
</body>
</html>
```

*Figura 41 Mensaje del servidor*

### 4.4.5.2 Funcionamiento y aplicaciones de un servidor web

#### 4.4.5.2.1 Funcionamiento

Un servidor web se encarga de mantenerse a la espera de peticiones HTTP llevada a cabo por un cliente HTTP que suele conocerse como navegador (Internet Explorer, Mozilla Firefox…).

Estas peticiones se realizan al puerto 80 del servidor, por tanto el servidor web estará 'escuchando' las peticiones que le lleguen a ese puerto para dar una respuesta con el contenido que el cliente solicita. A modo de ejemplo:

Al teclear *www.gogle.es* en el navegador, éste realiza una petición HTTP al servidor de dicha dirección (por supuesto con la ayuda del servidor DNS que es quién traducirá el nombre de la web con su dirección IP real). El servidor responde al cliente enviando el código HTML de la página; el cliente, una vez recibido el código, lo interpreta y lo muestra en pantalla. En la figura 33 vemos como sucede todo esto:

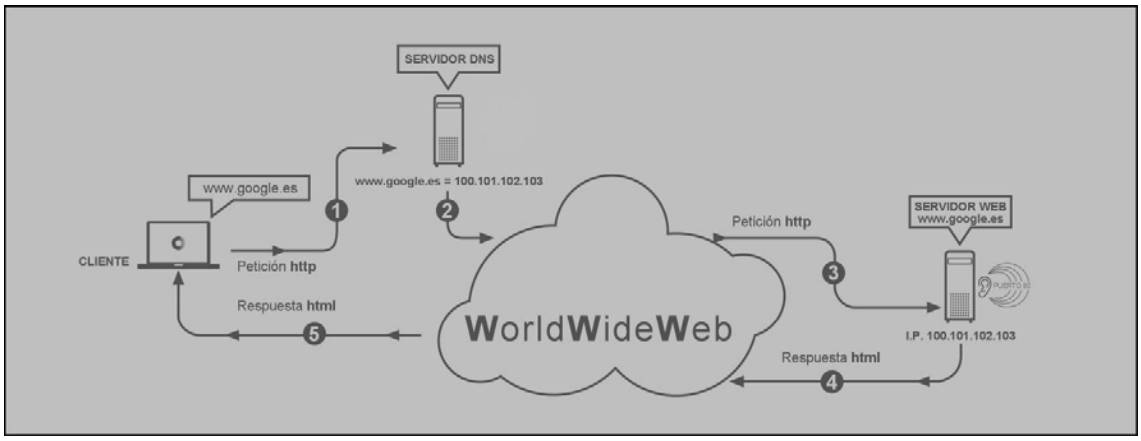


*Figura 42 Esquema de la visita a la web www.google.es*

El cliente es el encargado de interpretar el código HTML, es decir, de mostrar las fuentes, los colores y la disposición de los textos y objetos de la página; el servidor tan sólo se limita a transferir el código de la página sin llevar a cabo ninguna interpretación de la misma.

#### 4.4.5.2.2 Aplicaciones web

Sobre el servicio web clásico podemos disponer de aplicaciones web. Éstas son fragmentos de código que se ejecutan cuando se realizan ciertas peticiones o respuestas HTTP. Hay que distinguir entre:

· Aplicaciones en el lado del cliente: el cliente web es el encargado de ejecutarlas en la máquina del usuario. Son las aplicaciones tipo Java o

Javascript. El servidor proporciona el código de las aplicaciones al cliente y éste, mediante el navegador, las ejecuta. Es necesario, por tanto, que el cliente disponga de un navegador con capacidad para ejecutar aplicaciones (también llamadas scripts). Normalmente, los navegadores permiten ejecutar aplicaciones escritas en lenguaje javascript y java, aunque pueden añadirse más lenguajes mediante el uso de plugins.

· Aplicaciones en el lado del servidor: el servidor web ejecuta la aplicación; ésta, una vez ejecutada, genera cierto código HTML; el servidor toma este código recién creado y lo envía al cliente por medio del protocolo HTTP.

Las aplicaciones de servidor suelen ser la opción por la que se opta en la mayoría de las ocasiones para realizar aplicaciones web. La razón es que, al ejecutarse ésta en el servidor y no en la máquina del cliente, éste no necesita ninguna capacidad adicional, como sí ocurre en el caso de querer ejecutar aplicaciones javascript o java. Así pues, cualquier cliente dotado de un navegador web básico puede utilizar este tipo de aplicaciones.

Algunos conceptos relacionados con las aplicaciones web son:

· PHP: Es un lenguaje de programación usado generalmente para la creación de contenido para sitios web. PHP es un acrónimo recurrente que significa "PHP Hypertext Pre-processor" (inicialmente PHP Tools, o, *P*ersonal *H*ome *P*age Tools), y se trata de un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios web. Últimamente también para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica usando la biblioteca GTK+.

· ASP: es una tecnología del lado servidor de Microsoft para páginas web generadas dinámicamente, que ha sido comercializada como un anexo a Internet Information Server (IIS). La tecnología ASP está estrechamente relacionada con el modelo tecnológico de su fabricante. Intenta ser solución para un modelo de programación rápida ya que programar en ASP es como programar en VisualBasic, por supuesto con muchas limitaciones ya que es una plataforma que no se ha desarrollado como lo esperaba Microsoft. Lo interesante de este modelo tecnológico es poder utilizar diversos componentes ya desarrollados como algunos controles ActiveX. Otros problemas que han hecho evolucionar esta tecnología es el no disponer de información "que

oriente a quienes desean aprenderla y resulta muy costosa en tiempo descubrir aquí y allá toda la información para volverla altamente útil".

· PERL: Perl, Lenguaje Práctico para la Extracción e Informe (ver abajo) es un lenguaje de programación diseñado por Larry Wall creado en 1987. Perl toma características del C, del lenguaje interpretado shell (sh), AWK, sed, Lisp y, en un grado inferior, muchos otros lenguajes de programación. Estructuralmente Perl está basado en un estilo de bloques como los del C o AWK, y fue ampliamente adoptado por su destreza en el procesado de texto y no tener ninguna de las limitaciones de los otros lenguajes de script.

· CGI: Common Gateway Interface (en castellano «Interfaz Común de Pasarela», abreviado CGI) es una importante tecnología de la World Wide Web que permite a un cliente (explorador web) solicitar datos de un programa ejecutado en un servidor web. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor web y una aplicación externa. Las aplicaciones CGI fueron una de las primeras maneras prácticas de crear contenido dinámico para las páginas web.

En una aplicación CGI, el servidor web pasa las solicitudes del cliente a un programa externo. La salida de dicho programa es enviada al cliente en lugar del archivo estático tradicional. CGI ha hecho posible la implementación de funciones nuevas y variadas en las páginas web, de tal manera que esta interfaz rápidamente se volvió un estándar, siendo implementada en todo tipo de servidores web.

· JSP: JavaServer Pages (JSP), en el campo de la informática, es una tecnología para crear aplicaciones web. Es un desarrollo de la compañía Sun Microsystems y su funcionamiento se basa en scripts, que utilizan una variante del lenguaje java. La JSP es una tecnología Java que permite a los programadores generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo. Las JSP's permiten al código Java y a algunas acciones predefinidas ser incrustadas en el contenido estático del documento web. En las JSP se escribe el texto que va a ser devuelto en la salida (normalmente, código HTML) incluyendo código java dentro de él, para poder modificar o generar contenido dinámicamente.

### 4.4.5.3 El servidor web APACHE


#### 4.4.5.3.1 Introducción

El servidor HTTP Apache es un software (libre) servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etcétera), Windows y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que originalmente Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. Era, en inglés, a patchy server (un servidor "parcheado"). Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation.

Apache presenta entre otras características mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

Apache tiene amplia aceptación en la red: Apache es el servidor HTTP más usado, siendo el servidor HTTP del 70% de los sitios web en el mundo y creciendo aún su cuota de mercado.

El núcleo 2.x de Apache tiene varias mejoras clave sobre el núcleo de Apache 1.x. Estas mejoras incluyen threads de UNIX, mejor soporte para plataformas no Unix (como Windows), un nuevo API, y soporte de IPv6.

#### 4.4.5.3.2 Arquitectura de APACHE

El servidor Apache es un software que esta estructurado en módulos. La configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo. Los módulos del Apache se pueden clasificar en tres categorías:

· Módulos Base: Módulo con las funciones básicas del Apache.

· Módulos Multiproceso: son los responsables de la unión con los puertos de la máquina, acepando las peticiones y enviando a los hijos a atender a las peticiones.

· Módulos Adicionales: Cualquier otro módulo que le añada una funcionalidad al servidor.

Las funcionalidades más elementales se encuentran en el módulo base, siendo necesario un módulo multiproceso para manejar las peticiones.

Se han diseñado varios módulos multiproceso para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código. El resto de funcionalidades del servidor se consiguen por medio de módulos adicionales que se pueden cargar. Para añadir un conjunto de utilidades al servidor, simplemente hay que añadirle un módulo, de forma que no es necesario volver a instalar el software.

#### 4.4.5.3.2.1 Módulos base y Módulos multiproceso

mpm_common: Colección de directivas que se implementan en más de un módulo multiproceso.

beos: Módulo de multiproceso optimizado para BeOS.

leader: Variable experimental de MPM.

mpm_netware: Módulo de multiproceso que implementa un servidor web optimizado para Novell NetWare.

mpmt_os2: MPM híbrido, multiproceso y multihilo para OS/2 .

perchild: Módulo multiproceso que permite a los procesos demonio servir las peticiones que se asignan a distintos id de usuario.

prefork: Implementa un servidor sin hilos.

threadpool: Variante experimental del módulo estándar de MPM .

mpm_winnt: Módulo multiproceso optimizado para Windows NT.

worker: Módulo multiproceso que implementa un híbrido multihilos y multiprocesos de servidor Web.

#### 4.4.5.3.2.2 Módulos adicionales

mod_access: proporciona control de acceso basándose en el nombre del host del cliente, su dirección IP u otras características de la petición del cliente.

mod_actions: este módulo se utiliza para ejecutar Scripts CGI, basándose en el tipo de medio o el método de petición.

mod_alias: proporcionado para mapear diferentes partes del sistema de ficheros del servidor en el árbol de documentos del servidor, y para redirección de URL's.

mod_asis: envío de ficheros que tienen sus propias cabeceras http.

mod_auth: autentificación de usuario utilizando ficheros de texto.

mod_auth_anon: permite a usuarios anónimos acceder a áreas autentificadas.

mod_auth_dbm: proporciona autentificación utilizando ficheros DBM.

mod_auth_digest: autentificación de usuario utilizando MD5.

mod_auth_ldap: permite la utilización un directorio LDAP para almacenar la base de datos de autentificación.

mod_autoindex: muestra los contenidos de un directorio automáticamente, parecido al comando ls de Unix.

mod_cache: Caché de contenidos indexados por URI's.

mod_cern_meta: Semántica de etiquetas meta del CERN.

mod_cgi: Ejecución de Scritps CGI.

mod_cgid: ejecución de Scripts CGI utilizando un demonio CGI externo.

mod_charset_lite: para la especificación del juego de caracteres de las traducciones.

mod_deflate: comprime el contenido antes de ser enviado al cliente.

mod_dir: Proporcionado para redirecciones y para servir los ficheros de listado de directorios.

mod_disk_cache: Caché para almacenar contenidos identificados por URI.

Moho echo: Un servidor simple de echo para ilustrar los módulos del protocolo.

mod_env: modificación del entorno que se envía a los scripts CGI y las páginas SSI.

moho expires: Generación de las cabeceras http Expires, de acuerdo de los criterios especificados por el usuario.

mod_ext_filter: pasa el cuerpo de la respuesta a través de un programa antes de enviársela al cliente.

mod_file_cache: cachea una lista estática de ficheros en memoria.

mod_headers: personalización de las peticiones HTTP y las cabeceras de las respuestas.

mod_imap: proceso de imágenes en el lado del servidor.

mod_include: Documentos HTML generados por el servidor (Server Side Includes).

mod_info: proporciona una visión comprensiva de la configuración del servidor.

mod_isapi: Extensiones ISAPI en Apache para Windows.

mod_ldap: pool de conexiones LDAP y cacheo de resultados para la utilización de otros módulos LDAP.

mod_log_config: registro de las peticiones hechas al servidor.

mod_logio: registro del número de bytes recibidos y enviados en cada respuesta.

mod_mem_cache: Caché de contenidos identificados por URI.

mod_mime: asocia las extensiones de peticiones de los ficheros con el comportamiento del fichero (manejadores y filtros) y contenido (tipos mime, idioma, juego de caracteres y codificación).

mod_mime_magic: determina el tipo MIME de un fichero mirando unos pocos bytes del contenido.

mod_negotiation: se proporciona para la negociación del contenido.

mod_proxy: servidor HTTP/1.1 proxy/gateway.

mod_proxy_connect: extensión de mod_proxy para la gestión de las peticiones CONNECT.

mod_proxy_ftp: soporte FTP para mod_proxy.

mod_proxy_http: soporte HTTP para el módulo mod_proxy.

mod_rewrite: proporciona un motor de reescritura basado en reglas que rescribe las peticiones de URL's al vuelo.

mod_setenvif: permite la configuración de las variables de entorno basándose en las características de la petición.

mod_so: carga del código ejecutable y los módulos en al iniciar o reiniciar el servidor.

mod_speling: intenta corregir las URL mal puestas por los usuarios, ignorando las mayúsculas y permitiendo hasta una falta.

mod_ssl: criptografía avanzada utilizando los protocolos Secure Sockets Layer y Transport Layer Security.

mod_status: proporciona información en la actividad y rendimiento del servidor.

mod_suexec: permite a los scripts CGI ejecutarse con un nombre y grupo específico.

mod_unique_id: proporciona variables de entorno y un identificador único para cada petición.

mod_userdir: directorios específicos para usuarios.

mod_usertrack: registro de actividad de un usuario en el sitio.

mod_vhost_alias: Proporcionado para configurar muchos servidores virtuales dinámicamente.

#### 4.4.5.4 Instalación y configuración

Para empezar, comentar que la instalación de Apache viene acompañada de alguna 'cosa' mas, es decir, no se va a instalar solo Apache sino que lo que se instalará es un servidor LAMP.

El acrónimo LAMP (Linux, Apache, Mysql y Php) se refiere a este conjunto de subsistemas de software necesarios para alcanzar una solución global, en este caso configurar sitios web o servidores dinámicos.

Las versiones que se van a instalar son: Apache 2.0, MySql 5.0 (gestión de bases de datos) y Php 5.

Lo primero que se hará es asegurar que están activados y actualizados los repositorios en Ubuntu. Un repositorio es una fuente de localidades en Internet para aplicaciones. Lo comprobamos abriendo el archivo sources.list:

```
sudo gedit /etc/apt/sources.list
```

Hay que descomentar las siguientes líneas:

```
deb http://security.ubuntu.com/ubuntu dapper-security universe

deb-src http://security.ubuntu.com/ubuntu dapper-security universe
```

*Figura 43 Actualización y activación de los repositorios.*

Y actualizar los repositorios:

```
sudo aptitude update
```

Una vez hecho esto se pasa a instalar los componentes de software del servidor web:

```
Sudo aptitude install apache2

Sudo aptitude install php5

Sudo aptitude intstall mysql-server-5.0
```

*Figura 44 Instalación de los componentes del servidor web Apache.*

Para permitir la interacción entre Apache, Mysql y Php se instalan las siguientes librerías:

```
Sudo aptitude install libapache-mod-auth-mysql php5-mysql
```

Seguidamente se reinician los servicios.

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

Sudo /etc/init.d/mysql restart
```

*Figura 45 Reinicio de los demonios.*

Para comprobar que el servidor web está funcionando correctamente se puede hacer lo siguiente:

En una ventana del navegador web (Internet Explorer, Mozilla Firefox…) se teclea: *http:///localhost.* También se puede probar tecleando el nombre que se había definido en el servidor DNS para esa máquina: http://pc1pfc.homeip.net y a la vez comprobar el buen funcionamiento del servidor DNS. Si todo ha ido correctamente se visualizará la página de presentación de Apache indicando que el servidor ha sido instalado correctamente

Para probar PHP en el servidor Apache se creará un archivo con esa extensión, .php, con el nombre prueba y se colocará en el directorio correspondiente:

```
Sudo gedit /var./www/prueba.php
```

Se edita escribiendo el mensaje que queramos (“probando mi servidor” por ejemplo) y se vuelve a teclear la dirección del servidor como se ha hecho anteriormente. Ahora la diferencia está en que en la página de presentación hay un link al que si se accede muestra el mensaje “probando mi servidor”.

Con esto ya estaría trabajando el servidor WEB, esperando a recibir peticiones al puerto 80 para poder atenderlas con lo que tenga alojado en sus directorios.

# 5. Guión de prácticas

# 5. Guión de prácticas

## 5.1 Introducción

Una vez se han instalado, configurado y comprobado que funcionan correctamente los diferentes servicios, se pasará a la redacción y realización de los ejercicios prácticos que se plasmarán en un guión para aplicar a la docencia de la asignatura de Telemática. Hay que resaltar la importancia, para los futuros ingenieros, que tiene conocer, saber configurar, diagnosticar y resolver posibles problemas de estos servicios ya que son los que se encuentran día a día en todas las redes informáticas con conexión a Internet. Será una práctica por servicio realizado, la cual constará de tres partes:

- **Guión de Prácticas:** Formado por la teoría expuesta en cada uno de los puntos donde ha sido tratado el servicio de Internet configurado.

- **Estudios Previos:** Una serie de preguntas teóricas referentes al servicio en cuestión para que el alumno consulte y entienda los fundamentos teóricos del servicio que se está trabajando.

- **Ejercicios Prácticos:** Un conjunto de actividades donde el alumno pondrá de manifiesto los conocimientos adquiridos con los estudios previos aplicándolos a las diferentes configuraciones y análisis que se solicitarán.

## 5.2 DHCP

### Estudios Previos

1. Explica en qué consiste el servicio DHCP. Enumera las diferencias que existen entre DHCP y BOOTP.
2. Enumera y explica, de forma detallada, las diferentes configuraciones que permite el protocolo DHCP.
3. Explica los pasos que sigue el cliente para que su petición sea atendida por el servidor DHCP.
4. ¿Para que es necesaria la dirección broadcast en el servicio DHCP? ¿Y la dirección MAC?
5. Enumera las ventajas y desventajas que tiene una red configurada con un servidor DHCP.

6. Define la función de los siguientes mensajes e indica si el mensaje lo envía el servidor DHCP o el cliente configurado con DHCP
   - DHCP Discover
   - DHCP Offer

## Ejercicios Prácticos

1. Configura tu máquina Windows para que obtenga una configuración de red de forma automática. Indica, de forma detallada, los pasos que has seguido.
2. Con el paquete dhcp3-server instalado en la máquina Linux, modifica el o los ficheros de configuración necesarios para que asigne correctamente una configuración de red a los clientes, con un rango cualquiera de 10 IPs y con todos los parámetros necesarios para que puedan navegar por Internet. Indica los ficheros que has modificado y razona la respuesta.
3. Con la configuración de red asignada, arranca el analizador de protocolos de red Ethereal y comienza la captura del tráfico de tu tarjeta de red. Libera y renueva tu IP y finaliza la captura de tráfico. Encuentra en dicha captura los mensajes de 'negociación' entre cliente y servidor DHCP e indica cuales son.
4. Identifica por cuanto tiempo tenemos el alquiler de nuestra IP por defecto y modifícalo para tenerla tan solo 1 minuto de arrendamiento. Indica el fichero que has modificado y las consecuencias que este tiempo de arrendamiento tiene en el tráfico de la red. Razona la respuesta.
5. ¿Por que si solo tenemos un minuto de alquiler de IP y transcurre mas tiempo no dejamos de tener conexión? Analiza esos pasos con ethereal identificando los mensajes.

## 5.3 SSH

### Estudios Previos

1. Define SSH, comenta sus principales características, puertos que utiliza y que ventajas tiene sobre otras herramientas que tiene la misma utilidad como pueden ser rsh (remote shell) o el telnet.
2. Enumera y comenta brevemente alguno de los algoritmos de encriptación que utiliza SSH.
3. Describe los pasos que realizan tanto el emisor y el receptor en una conexión SSH.
4. Dibuja y explica los campos del 'paquete' SSH.
5. Comenta los diferentes tipos de autenticación de usuario que existen en SSH.
6. ¿Que es SFTP?

### Ejercicios Prácticos

1. Realiza una conexión SSH a la máquina remota con un usuario no privilegiado y después con el usuario 'root'. Comprueba que permisos tienes para cada uno de los usuarios.
2. Desconecta la sesión, arranca el analizador de protocolos de red Ethereal para realizar capturas y vuelve a establecerla. Identifica e indica los paquetes del establecimiento de la sesión así como los de los algoritmos de encriptación.
3. Con el analizador funcionando y la sesión SSH establecida envía un PING desde el cliente al servidor y viceversa e identifica los paquetes relacionados con dicha herramienta en las dos direcciones.
4. Realiza una descarga de un fichero de prueba mediante SFTP y captura el tráfico que circula por la tarjeta de red durante la transmisión del fichero. Posteriormente, realiza el mismo procedimiento con la aplicación FTP. ¿Qué diferencias existen entre los dos procedimientos? Razona la respuesta.

## 5.4 DNS

### Estudios Previos

1. Realiza una pequeña descripción histórica de porqué y cómo se llegó a pensar y crear el servidor DNS.
2. Define el concepto de 'dominio' y describe los diferentes tipos que existen.
3. Describe el funcionamiento del protocolo DNS, puertos que utiliza así como el formato de su cabecera.
4. Comenta los diferentes métodos de búsqueda que utiliza el DNS.
5. Haz una descripción de BIND, el servidor DNS más usado en Internet y el que utilizaremos en la realización de las prácticas.

### Ejercicios Prácticos

1. Crea un dominio dentro de tu red y define una entrada para que sin introducir la IP del servidor SSH podamos realizar una conexión remota. Indica los pasos realizados y los ficheros modificados.
2. ¿Cómo podemos realizar la prueba de que la definición es correcta desde la ventana de comandos (terminal)?. Realiza dicha comprobación.
3. Configura el servidor DNS para que sea capaz de darnos respuesta a cualquier página de Internet a pesar de que no estén definidas en él, o lo que es lo mismo, nuestro servidor no sea el autoritario del domino que consultamos. Indica los pasos realizados y los ficheros modificados.
4. Configura el servidor DNS para que conociendo una dirección IP dada, responda con el nombre del host a la que está asociada. Indica los pasos realizados y los ficheros modificados.

## 5.5 Servidor web APACHE

### Estudios Previos

1. Describe la función del protocolo HTTP. ¿Qué diferencia existe entre HTTP y HTML?
2. ¿Qué son las 'cookies' y por qué se utilizan?
3. Explica brevemente el funcionamiento de un servidor WEB y enumera diferentes distribuciones existentes en el mercado

4. Explica los dos tipos de aplicaciones webs que hay en el servicio web clásico. Indica para que se utiliza cada una de ellas.
5. Enumera y comenta los diferentes módulos del servidor web Apache.
6. ¿Qué es un servidor LAMP?

## Ejercicios Prácticos

1. Arranca el servidor web Apache y comprueba que esté escuchando correctamente por el puerto 80. (Descubre la página de presentación de APACHE). Justifica la respuesta con una captura del tráfico HTTP realizada con el analizador de protocolos Ethereal.
2. Cuelga de dicha página un fichero de prueba en PHP y confirma que se accede y se visualiza lo que hayas introducido en el fichero. Indica los pasos que has seguido y los ficheros que has modificado.
3. Arranca el analizador de protocolos Ethereal y realiza capturas de los mensajes que se intercambian cliente y servidor web en la visita a la página de presentación. Indica dichos mensajes y descríbelos brevemente.
4. Arranca el analizador de protocolos Ethereal y realiza capturas de los mensajes que se intercambian cliente y servidor web en la visita a la página de prueba. Indica dichos mensajes y descríbelos brevemente. ¿Qué diferencia existe entre esta captura y la del ejercicio anterior?

# 6. Conclusiones

## 6. Conclusiones

Una vez realizado este proyecto podemos extraer las siguientes conclusiones:

Internet se ha convertido uno de los medios de comunicación con mayor expansión, crecimiento e importancia de la historia (podríamos considerarlo el más importante por encima de teléfono o televisión ya que estos medios podemos incluirlos hoy en día dentro de Internet). Día a día ha tomado y toma mayor fuerza e importancia en todas las áreas de nuestra vida, tanto a nivel de ocio como en el terreno laboral.

El movimiento GNU/Linux y del software libre es uno de los mayores trabajos y/o proyectos de cooperación en el campo de la Ingeniería de toda la historia, por todos los pasos que ha dado desde sus inicios, hace ya mas de 20 años, hasta la actualidad, con sistemas operativos gratuitos que nos ofrecen todas las prestaciones y proporcionan todas las herramientas necesarias para sacarle todo el rendimiento y provecho a un ordenador (trabajar en red, realizar documentos, hojas de cálculo navegar, descargar y reproducir música, videos, fotos…).

Con esto último, queda claro entonces, que cualquiera puede tener en su casa todo el software necesario para hacer uso de internet o de un PC sin necesidad de gastarse el dinero en caras licencias de software y mantenerse dentro de la ley sin utilizar la 'piratería', por otro lado un 'campo' en gran desarrollo gracias a la red.

El conocimiento de como funcionan los servicios básicos de Internet (DNS, servidor Web, DHCP…) así como el conocimiento de que existe 'algo más' que Windows me ha sido de gran utilidad tanto a nivel profesional como a nivel personal y sobre todo esto último (la utilización de Linux) es algo que tendría que ser mucho más habitual para los 'internautas' ya que es mucho mas sencillo de lo que a priori podría parecer. Además se dispone de una basta documentación, foros, listas de correo... a lo largo y ancho de la red para obtener siempre una respuesta a cualquier duda que se nos pueda plantear.

Resaltar también la importancia para los futuros ingenieros de conocer, configurar y resolver problemas sobre estos servicios ya que son los más habituales que se encontrarán en el día a día.

# 7. Líneas futuras

## 7. Líneas futuras

Las configuraciones que se han realizado en el proyecto son siempre dentro de la red de área local con ISP telefónica, por tanto, están dando servicio de forma privada. El servidor web no es accesible desde Internet, el servidor DNS no tiene definido ningún dominio visible para el resto del mundo, aunque si es capaz de dar respuesta a cualquier página que le preguntemos (gracias a los forwarders).

Debido a esto, las líneas futuras del proyecto podrían ser la publicación de nuestro dominio privado en Internet así como la visibilidad desde fuera del servidor web.

Se tendría que pensar como trabaja el router de nuestro ISP para realizar dichas publicaciones. El router es a su vez un cliente DHCP por lo que cada vez que se reinicia coge un IP diferente y aleatoria. Se debería contratar una IP fija o en su defecto instalar algún programa para tener un domino siempre asociado a la IP pública que coja el router. Este programa podría ser *DYNDNS*, una aplicación gratuita que asocia un nombre de dominio (ejemplo.dyndns.org) a la IP pública (sea la que sea y cambie las veces que cambie) que tenga el router. Una vez instalado este programa nuestra IP pública siempre estará asociada a ese dominio. Una vez hecho esto se tendría que configurar en el router los NATS (Network Adress Translation) para que redirigiera las peticiones que le llegan a la IP privada que corresponda. Un ejemplo:

Si se quiere publicar el servidor web en Internet para que estemos donde estemos tecleemos ‘miservidorweb.dyndns.org’ (por ejemplo) y nos responda mostrándonos la página que se tenga instalada en el servidor Apache. En el router tendremos que redirigir las peticiones que le lleguen al router (llegarán al dominio que está asociado a la IP pública) al puerto 80 a la IP privada 192.168.1.33 que es la que tiene el servidor web y así poder dar una respuesta a esas peticiones.

Lo mismo se debería aplicar con el resto de servicios teniendo en cuenta que trabajan con puertos diferentes.

# 8. Bibliografía

## 8. Bibliografía


- "Linux. Guía de instalación y administración"
  Autor: Vicente López Camacho
- Introducció al programari Lliure
  Autores: Jesús González Barahona y Gregorio Robles
- Sistema Operatiu GNU/Linux Basic
  Autores: Roger Baig Viñas i Francesc Aulí
- Xarxes de Computadors
  Autores: Jose Maria Barceló, Jordi Iñigo i Enric Peig
- Sybex CCNA Cisco Certified Network Asóciate
  Autor: Todd Lammle
- http://www.todolinux.org
- http://www.forosuse.org
- http://www.ubuntu-es.org
- http://www.ubuntu-es.org
- http://www.linux.es