# Tema 1. Introducción a los servicios de red e Internet.

# 1. Tema 1. Introducción a los servicios de red e Internet.

## 1.1. Introducción.

En la actualidad la manera en que los usuarios usan Internet está evolucionando muy rápidamente debido al desarrollo de nuevos servicios que están simplificando su uso. Esto, junto a las capacidades de los dispositivos móviles está posibilitando el acceso a la información de una manera más eficiente.

Sabemos que Internet es una red que está enfocada al intercambio de información entre usuarios y equipos. La información disponible es enorme por lo que debe estar organizada y estructurada para que sea accesible y sobre todo útil. Esta información se encuentra principalmente estructurada en páginas de hipertexto (páginas web), con enlaces a otras páginas, incluyendo imágenes, vídeos, servicios web, etc.

En general, las aplicaciones que hacen uso de Internet se basan en transferencia de información, aunque se distinguen unas de otras por el formato o los tipos de datos que manejan. Algunos de los servicios más importantes que incluye Internet son:

- Transferencia de archivos.
- Correo electrónico.
- Conexión remota a equipos.
- Acceso a información de hipertexto (también llamadas páginas web).

## 1.2. Estructuras de las redes de comunicaciones.

Una red de comunicación o de transmisión de datos es una estructura formada por medios *físicos* (equipos y dispositivos) y *lógicos* (programas de transmisión y control) que permite la comunicación en una determinada zona geográfica.

La red de comunicación permite el intercambio de información entre un emisor y un receptor mediante señales de naturaleza física. A la señal recibida por el receptor siempre le acompaña un componente de ruido que se suma durante su circulación a través de la red. Esto obliga a introducir mecanismos software para la de detección y corrección de errores.

Hay que tener en cuenta que una línea de comunicación no es solamente un cable, sino que también entran en juego otros elementos, como el *sistema de conmutación*, que decide la ruta que va a seguir la información hasta su destino y el *sistema de señalización*, que controla las comunicaciones establecidas.

## 1.3. Arquitectura TCP/IP.

Para interconectar sistemas, se necesitan una serie de programas que establezcan todos los pasos que se deben seguir para: establecer las conexiones, realizar transferencia de información, controlar los errores, etc. Sin este conjunto de normas no estarían disponibles los servicios necesarios.

La mayoría de las redes actuales, incluida Internet, usan la arquitectura TCP/IP para las comunicaciones. La especificación **TCP/IP** (*Transmission Control Protocol/Internet Protocol* o *Protocolo de Control de la Transmisión/Protocolo de Interred*) se encuentra definida en documentos oficiales denominados RFC (*Request for Comments* o *Petición de Comentarios*), definidos desde el año 1983 por el **IAB** (*Internet Activities Board* o *Comité de Arquitectura de Internet*).

TCP/IP proporciona una estructura y una serie de normas de funcionamiento para interconectar sistemas. Para simplificar esta tarea se realiza una subdivisión del trabajo en niveles o capas relacionadas entre sí de manera que cada capa realiza una labor determinada.

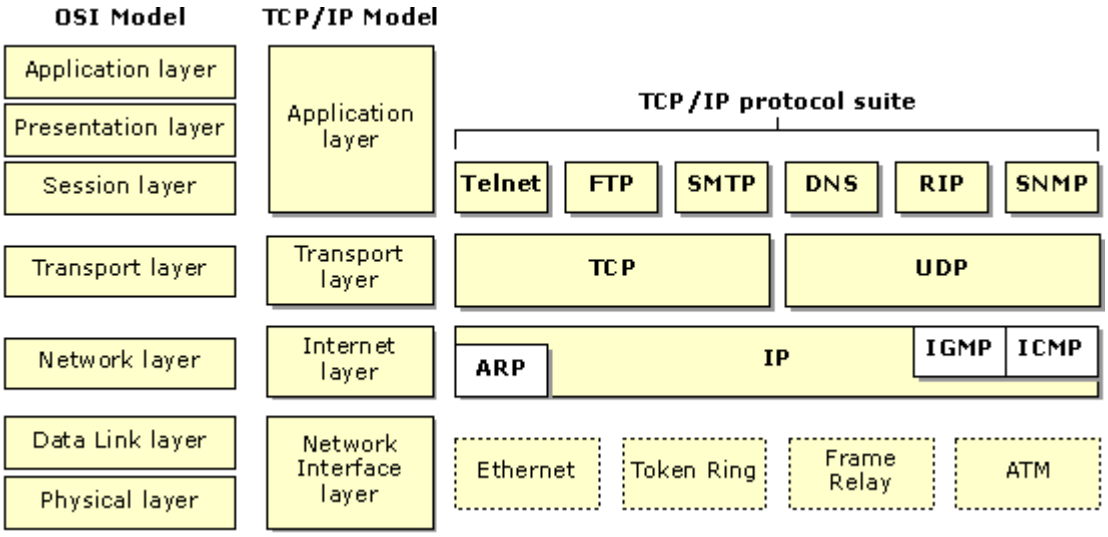
En cada capa existen una serie de protocolos que fijan unas series de normas para establecer la comunicación entre los sistemas. Cada protocolo se apoya en los protocolos de las capas inferiores para realizar su tarea y ofrece sus servicios a las capas superiores.

Las reglas que definen el protocolo TCP/IP son independientes del ordenador y del sistema operativo instalado, por lo que cualquier ordenador se puede comunicar con otro a través de la red. Ésta es una de las características que más ha contribuido al éxito y expansión de Internet.

En el caso de los sistemas operativos como Unix o Linux, TCP/IP está perfectamente integrada fundamentalmente por cuestiones históricas. (La red ARPANET, precursora de la actual Internet se diseñó inicialmente para comunicar máquinas Unix.) Otros, como Windows, la adaptación se realizó de forma distinta. Microsoft desarrolló inicialmente una serie de protocolos y normas que definen una red de

comunicación -la Red Microsoft- pero que era totalmente incompatible con Internet. Desde hace tiempo, Windows incluye TCP/IP; y muchos de los protocolos diseñados inicialmente para la Red Microsoft, se han ido sustituyendo por otros para conseguir una integración completa con TCP/IP.

Por otro lado el modelo OSI (***O**pen **S**ystem **I**nterconnection*) es el modelo de red descriptivo creado por la Organización Internacional para la Estandarización (ISO). Fue propuesto como una aproximación teórica y como una primera fase en la evolución de las redes de ordenadores. Pero el modelo TCP/IP es el que realmente se usa.




Comparativa de arquitecturas OSI y TCP/IP

### 1.3.1. El modelo cliente-servidor.

La mayoría de los servicios ofrecidos por una red de comunicación de ordenadores se basan en el modo cliente-servidor, consistente en que un proceso cliente solicita un servicio y un proceso servidor presta el servicio al cliente que lo solicitó.

Otro modelo es el modelo entre pares o P2P (*Peer to Peer*), donde todos los equipos de la red son responsables por igual en la comunicación de las aplicaciones sin que existan clientes y servidores fijos. Todos los nodos de la red se comportan como iguales entre sí.

En el modelo cliente-servidor existen dos procesos que interactúan entre sí:

- El proceso cliente, que normalmente inicia la comunicación, envía una petición a un proceso servidor quedando a la espera de una respuesta.
- El proceso servidor que suele ser un proceso que permanece a la espera escuchando las posibles conexiones de los clientes. Suelen ser procesos complejos, que necesitan de mecanismos de autenticación, autorización y seguridad; y robustos ya que habitualmente deben gestionar peticiones simultáneas de varios clientes.

## 1.4. Servicios de red.

Un servicio de red es una función que ofrecen las aplicaciones y los protocolos directamente a los usuarios o bien a otras aplicaciones. Las aplicaciones se comunican e intercambian información con otras aplicaciones apoyándose en los protocolos TCP/IP, tanto en los protocolos de nivel de aplicación como en los de niveles inferiores.

Por ejemplo, en el servicio web, una aplicación instalada por el usuario (un navegador) se comunica con una aplicación en un servidor web (Apache), las cuales se sirven del protocolo HTTP (protocolo a nivel de aplicación de TCP/IP) para realizar la comunicación entre ambas aplicaciones.

### 1.4.1. Servicios de alto nivel.

Los servicios de alto nivel son aquéllos que manejan directamente los usuarios. Por ejemplo, en una red de telefonía, el servicio fundamental para los usuarios es la transmisión de voz; en una red de ordenadores, sería el intercambio de datos (archivos, mensajes, vídeos, etc.).

Los servicios de alto nivel más importantes que ofrece actualmente Internet son:

**Transferencia de archivos entre equipos**. Era uno de los servicios de transmisión de datos más usados en una red. El protocolo **FTP** (*File Transfer Protocol* o Protocolo de Transferencia de Ficheros) permite la transferencia de archivos entre ordenadores en redes de arquitectura TCP/IP.

**Correo electrónico y mensajería instantánea**. El servicio de **correo electrónico** (*e-mail*) consiste en el envío y recepción de mensajes de texto (además de un conjunto de *archivos adjuntos*) desde un usuario origen a otro destino, sin necesidad de que el destinatario se encuentre conectado y disponible para su recepción. Los servicios de **mensajería instantánea** se diferencian del correo electrónico en que los usuarios deben estar conectados y "en línea". Existen muchos sistemas de mensajería instantánea que funcionan actualmente en Internet, aunque en la actualidad los más usados son los ofrecidos a través de páginas web dinámicas y por aplicaciones en dispositivos móviles.

**Acceso remoto a equipos**. El acceso vía **terminal remoto** a un ordenador ha sido el modo más frecuente de comunicaciones en red. Un programa *emulador de terminal* envía las órdenes que escribe el usuario en el *terminal* para que se ejecuten en el servidor, y este último devuelve los resultados al emulador de terminal para que aparezcan en la pantalla del usuario.

Para el acceso remoto, uno de los primeros protocolos aunque ya casi en desuso es **Telnet** (*Telematics Network*). Es un protocolo simple de terminal remoto que permite establecer una conexión entre un usuario y un servidor. El usuario realiza pulsaciones sobre el teclado del terminal que son enviadas al servidor por la red y procesadas por éste, tras lo cual, se realiza el envío de nuevo al usuario del resultado de la ejecución de las órdenes que aparece sobre la pantalla del terminal. Por eso las órdenes que reconoce el protocolo *telnet* son las mismas que se manejan en el sistema operativo del servidor.

También existen otros protocolos para el acceso remoto a un equipo, como **SSH** (*Secure Shell* o *Interfaz Segura*). Este protocolo, a diferencia del Telnet, ofrece mecanismos avanzados de seguridad de forma que la información se transmite cifrada.

Existen otros protocolos de acceso remoto que permiten la conexión pero a través del entorno gráfico, dando al usuario la sensación de que está trabajando directamente con el escritorio del equipo remoto.

**Consulta de información en hipertexto**. La **WWW** (*World Wide Web* o *Telaraña Mundial*) se utiliza para acceder a información distribuida a través de servidores Web de Internet. Se accede a través de documentos llamados **páginas**. Estas páginas son en realidad documentos de hipertexto. Para poder ver correctamente estos documentos, se necesita un programa adecuado llamado **visor** o **navegador**.

Cuando un cliente escribe una dirección de una página web, el servidor correspondiente recibe una petición a través del protocolo TCP por el puerto 80 solicitando la página concreta. Este protocolo se llama **HTTP** (*HyperText Transfer Protocol* o *Protocolo de Transferencia de Hipertexto*).

Algunos de los servicios comentados anteriormente como la transferencia de archivos o el correo electrónico, se están adaptando a la WWW, de forma que es posible descargar un archivo, consultar el correo o publicar un mensaje en un foro desde una página de hipertexto. Esto ha sido posible gracias al desarrollo de lenguajes que permiten desarrollar nuevas aplicaciones: las llamadas aplicaciones web.

Los navegadores actuales son capaces de interpretar código escrito en otros lenguajes que extienden la funcionalidad del protocolo, como XML, JavaScript, applets escritos en Java o PHP.

### 1.4.2. Servicios de bajo nivel.

Para que una red pueda ofrecer una serie de servicios de alto nivel, necesita de una compleja infraestructura que incluye a una serie de programas y servicios que realizan tareas más simples.

Cuando un usuario descarga una página de hipertexto mediante un navegador, está utilizando un servicio de alto nivel (el servicio web). Éste servicio requiere, a su vez, de la realización de otras operaciones más sencillas que tienen que ver con la forma en la que la red de comunicación subyacente transfiere los datos; como por ejemplo la comprobación de que el otro equipo está listo, la selección de la mejor ruta que debe seguir la información hasta alcanzar el destino, la confirmación de que el otro equipo acepta la conexión, la división de la información en fragmentos más pequeños para ser enviados individualmente, la ordenación y fusión de los mensajes recibidos en el destino, la comprobación de errores, etc. Estas operaciones se realizan gracias a la existencia de **servicios de bajo nivel**.

En Internet se utilizan dos servicios muy importantes que dan apoyo a otros servicios de alto nivel: **DHCP** (*Dynamic Host Configuration Protocol* o Protocolo de Configuración Dinámica de Equipos) y **DNS** (*Domain Name System* o Sistema de Nombres de Dominio). DHCP se usa para facilitar la

configuración automática de los equipos, de forma que los usuarios no tengan que configurar manualmente sus equipos. DNS se utiliza para ocultar el esquema de direcciones IP que usa Internet y así los usuarios puedan trabajar de una forma más cómoda con nombres de equipos.

## 1.5. El protocolo IP.

Las dos funciones principales del protocolo IP es establecer un *direccionamiento* para cada nodo de la red y determinar las rutas o *encaminamientos* que se ejecutan en los dispositivos que interconectan las redes.

La comunicación a nivel IP se realiza mediante unidades de datos denominadas *datagramas* las cuales tienen un formato que está especificado en el protocolo IP.

IP dispone de dos versiones: la versión 4 (IPv4), todavía la más utilizada actualmente, y la versión 6 (IPv6) aunque desarrollada hace tiempo, es ahora cuando parece que empieza realmente a implantarse.

### 1.5.1. Direccionamiento IP.

IP permite la conectividad extremo a extremo en la comunicación. Esto implica que todos los dispositivos deben tener una dirección única dentro de la red a la que pertenecen. El direccionamiento es independiente del dispositivo físico asignado y se puede modificar cuando se necesite.

El direccionamiento se establece mediante la *dirección IP*. Realmente, una dirección IP no identifica a un ordenador en la red, sino que identifica a una *interfaz de red*. Un ordenador con varias interfaces puede por tanto conectarse a diferentes redes. Incluso una misma interfaz de red puede tener varias direcciones IP; es lo que se conoce como direccionamiento virtual.

En IPv4, una dirección IP es un número binario de 32 bits, lo que permite un espacio de direcciones de $2^{32}$; es decir (4.294.967.296) direcciones diferentes. Para un manejo más cómodo, las direcciones se dividen en 4 grupos de 8 bits, donde cada grupo se escribe en decimal y separados por un punto. Por ejemplo 80.36.234.12.

Para conseguir que las direcciones IP expresen tanto información de direccionamiento como de encaminamiento se desglosan en dos partes:

- Identificador de red, que determina la red en que se encuentra el dispositivo.
- Identificador de equipo o *host* dentro de la red.

De esta manera, todos los equipos de una misma red comparten la parte de identificación de red, siendo éste un valor que dependerá del tamaño de la red. Las redes grandes usarán un identificador de red pequeño, y las redes pequeñas usarán uno grande.

Con la *máscara de red* se permite diferenciar el prefijo de la dirección IP que se corresponde con el identificador de red, de la parte correspondiente al identificador de equipo o *host*. La máscara de red es un número de 32 bits en donde las posiciones a "1" definen la parte correspondiente a la identificación de red, y las posiciones a "0" la parte que corresponde al identificador de equipo. Realizando la operación binaria AND entre la dirección de host y la máscara de red, se puede conocer a que red pertenece dicho equipo. Por ejemplo la dirección de host 192.168.30.73/255.255.255.192 pertenece a la red 192.168.30.64.

| | | |
|---|---|---|
| | 11000000 . 10101000 . 00011110 . 01001001 | Dirección de host 192.168.30.73 |
| AND | 11111111 . 11111111 . 11111111 . 11000000 | Máscara de red 255.255.255.192 |
| | 11000000 . 10101000 . 00011110 . 01000000 | dirección de red 192.168.30.64 |

Existen ciertas máscaras de red que se usan para las divisiones más comunes, las llamadas red con clase. Dichas máscaras son 255.0.0.0 (clase A), 255.255.0.0 (clase B) y 255.255.255.0 (clase C).

En la tabla siguiente se exponen algunas de direcciones de equipos junto a la máscara de red:

| **Dirección IP y máscara** | **Dirección de red** | **Número de equipo** |
|---|---|---|
| 80.36.234.12<br>255.0.0.0 | 80.0.0.0 | 36.234.12 |
| 10.245.132.65<br>255.255.0.0 | 10.245.0.0 | 132.65 |
| 192.168.30.73<br>255.255.255.192 | 192.168.30.64 | 73 |

La máscara de red también puede expresarse mediante la notación CIDR (*Classless Inter-Domain Routing)* situando a continuación de la IP del interfaz, un sufijo que indica cuántos bits de la máscara de red están a "1". Así, el último ejemplo de la tabla anterior se podría expresar de la forma:

192.168.30.73/255.255.255.192 -> 192.168.30.73/26

De todo el conjunto de direcciones IP, algunas de ellas tienen un significado especial:

- **Dirección de red**. Identifica a toda una red donde la parte correspondiente al identificador de host tiene todos sus bits a cero. Por ejemplo, la IP 122.76.211.89/24 tiene la dirección de red 122.76.211.**0**/24
- **Dirección de difusión limitada**. Se usa para mandar un mensaje de difusión o *broadcast* a todos los dispositivos de la propia red. Para todas las redes es la misma: 255.255.255.255
- **Dirección de difusión dirigida (*multicast*)**. Se usa para mandar un mensaje de difusión o *broadcast* a todos los dispositivos de una red concreta. Esta dirección se compone por el identificador de red en la que se va a realizar la difusión, seguida del identificador de equipo con todos sus bits a "1". Por ejemplo la dirección 122.76.211.**255**/24 permite realizar difusión dirigida en la red 122.76.211.**0**/24
- **Dirección de bucle local**. Permite referenciar de forma interna una interfaz. Esta dirección se usa para los procesos TCP/IP que se generan dentro del equipo. De esta forma, aunque un equipo no tenga ninguna interfaz de red física, puede usar la del bucle local para procesos TCP/IP. Se usa cualquier dirección de la red 127.0.0.0/8, como por ejemplo la dirección 127.43.121.55/8.

Dentro del espacio de direcciones, algunas están reservadas para el ámbito privado. Estas *direcciones privadas* que no deben tener acceso a Internet se usan para redes privadas o *intranets* y deben pertenecer a las siguientes redes: 10.0.0.0/8, 172.16.0.0/16 y 192.168.0.0/16. En contra posición, las llamadas *direcciones públicas* permiten identificar a un dispositivo conectado a Internet.

También existe un conjunto de direcciones reservado para aquellas redes en donde no existe ningún tipo de direccionamiento definido. En este caso, son los propios equipos los que se asignan a sí mismos las direcciones usando las direcciones de la red 169.254.0.0/16.

A parte de la configuración IPv4 vista anteriormente es posible que se necesite una configuración IPv6. IPv6 admite un rango de direcciones mucho mayor, pero tiene un formato más complejo. Utiliza un formato de 128 bits, permitiéndose alrededor de 340 sextillones de direcciones.

La forma de especificar direcciones IPv6 es x:x:x:x:x:x:x:x donde cada x representa un número de 16 bits normalmente expresado con 4 dígitos hexadecimales. Por ejemplo:

2001:0db8:85a3:08d3:1319:8a2e:0370:7334

Las direcciones IPv6 también, se pueden representar de otras formas diferentes:

- Igual que la anterior, pero suprimiendo todos los ceros consecutivos en la dirección y sustituyéndolos por ':: '. Estos '::' solamente pueden aparecer una vez en la dirección.
- Utilizando una notación mixta formada por una parte de dirección v6 (seis números hexadecimales de 16 bits separados por dos puntos) y otra de v4 (cuatro números decimales de ocho bits).

Una dirección IPv6 también tiene varios campos: los primeros bits forman un **prefijo**, que indica el tipo de dirección; los bits centrales especifican un número de red (que puede no existir) y los bits finales especifican un número de host. La tabla siguiente muestras varios prefijos usuales.

| Prefijo | Descripción |
|---|---|
| 00 | Dirección IPv4. |
| 2 ó 3 | Dirección asignada por un proveedor de acceso a Internet. |
| de FE80 a FEBF | Direcciones privadas dentro del ámbito de la subred. |
| FF | Dirección de multidifusión. |

En IPv6 la representación de las máscaras de red es justamente la contraria a IPv4; por ejemplo un sufijo /50, indica que los 50 últimos bits de la dirección se reservan para numerar hosts.

Para mantener la coexistencia entre IPv4 e IPv6, los equipos actuales suelen tener instalados los dos protocolos a la vez, asignándose una dirección IPv4 y otra IPv6. Este mecanismo permite la coexistencia hasta que IPv6 se haya implantado definitivamente, pero aumenta la complejidad en la configuración de red de los equipos.

### 1.5.2. Encaminamiento IP.

El protocolo IP también es el responsable del encaminamiento de los datagramas, llevándolos desde una máquina origen a otra destino independientemente de la red en que se encuentren las máquinas.

Los *encaminadores* o *routers* son los dispositivos encargados de esta tarea, manteniéndose conectados al menos a dos redes y realizando el encaminamiento de los datagramas que pasen por él. Para ello se apoya en las *tablas de encaminamiento*.

Las tablas de encaminamiento almacenan la información necesaria para realizar el encaminamiento de los datagramas y están presentes tanto en los routers como en los propios equipos.

Los propios equipos también realizan tareas de encaminamiento en su tráfico saliente, de forma que cuando un equipo coloca un datagrama en la red destinado a una IP concreta, examinando la máscara de red decide si el datagrama lo envía directamente al equipo destino porque está en la misma red, o lo envía al encaminador para que éste lo dirija hacia otra red de destino.

Cuando un router recibe un datagrama, si éste se dirige a una dirección que pertenece a una red a la que está conectado, realiza una entrega directa a dicha red; en el caso de que vaya dirigida a una red a la que no está conectado directamente, lo reenviará al router que indique su tabla de encaminamiento. Este proceso se repetirá hasta que se alcance la red de destino o agotar el tiempo de vida del datagrama.

Cuando se arrancan los equipos, las tablas de encaminamiento de los routers y de los equipos se inicializan con las rutas correspondientes a sus redes adyacentes. A partir de este momento, la configuración del encaminamiento se puede realizar de dos formas: encaminamiento *estático* o *dinámico*.

- En el encaminamiento estático las tablas se configuran manualmente. No es muy recomendable esta estrategia ya que cualquier cambio en la estructura de red necesita de una actualización de las tablas.
- En el encaminamiento dinámico es el propio encaminador el que actualiza sus tablas usando para ello protocolos que permiten que los routers intercambien información entre ellos para mantener actualizadas sus tablas. Protocolos de encaminamiento son RIP (*Routing Information Protocol*), OSPF (*Open Shortest Path First*) y BGP (*Border Gateway Protocol*).

Los equipos de aquellas redes que no se conectan con otras, salvo a Internet, también necesitan conocer a dónde deben enviar los datagramas que no tienen como destino la propia red. A este parámetro se le denomina *puerta de enlace, pasarela por defecto* o *default gateway*. Su valor se establece con la dirección IP del dispositivo que conecta nuestro equipo con Internet (normalmente es la dirección del encaminador o el dispositivo propiedad del proveedor de acceso que conecta la red de comunicación de nuestro equipo con Internet).

## 1.6. Protocolos TCP y UDP.

IP permite comunicar dos máquinas haciendo que los datagramas puedan ir desde un origen a un destino, pero no permite mantener varias comunicaciones simultáneas entre dichos equipos, ya que sólo con la IP no se puede diferenciar qué datagramas pertenecen a cada una de las comunicaciones establecidas.

Mediante la capa de transporte se permite diferenciar y gestionar múltiples orígenes y destinos en una comunicación, y múltiples comunicaciones en cada equipo. Para ello se usa el concepto de *puerto de comunicaciones* para identificar cada unas de las comunicaciones que se establezcan y por lo tanto para identificar cada proceso de nivel de aplicación establecido por cada comunicación.

Todo proceso de la capa de aplicación usa uno o varios puertos a través de los cuales es accesible. Cada puerto se identifica por un número binario de 16 bits cuyos valores en decimal varían entre 0 y 65535.

Según los servicios que usan los puertos, se pueden diferenciar en tres clases:

- [0-1023]. Son los llamados puertos conocidos (*well known ports*) y están reservados para aplicaciones y servicios estándar: FTP, SMTP, HTTP, etc. Las aplicaciones clientes se conectan a estos puertos para acceder a los servicios.
- [1024-49151]. Puertos registrados usados para aplicaciones no estándar instaladas por el usuario que no tienen un puerto bien conocido asignado.
- [49152-65535]. Puertos dinámicos. Se emplean para iniciar conexiones desde el cliente y no suelen usarse en procesos de servidor.

La correspondencia entre los procesos y los puertos a usar se pueden establecer de dos maneras:

- Asignación estática: los puertos conocidos se reservan para aplicaciones estándar y sólo pueden emplearse por estos procesos.
- Asignación dinámica: cuando un proceso necesita un puerto, y éste no se asigna de forma estática, el sistema operativo le asigna un puerto de tipo dinámico que esté disponible.

Aunque a nivel de transporte existen dos protocolos TCP y UDP, ambos son independientes y los puertos que usan cada uno de ellos también lo son.

Es posible controlar la información que circula a través de los puertos usando *cortafuegos* o *firewall*. Se instalan en el propio equipo o bien como un dispositivo independiente entre el equipo y la red. También existen dispositivos encaminadores o *routers* que pueden realizar funciones de cortafuegos.

Cuando se instala un servicio de red en un equipo siempre es necesario que el puerto o puertos asociados se encuentren abiertos o activos. Por defecto, el equipo activa los puertos cuando se pone en marcha el servicio, de forma que el resto de ordenadores de la red pueden utilizarlo. La tabla muestra una lista con algunos de los puertos de transporte más utilizados

| Puerto | Servicio | Descripción |
|---|---|---|
| 7 | Echo | Puerto de "eco" |
| 20 | FTP Datos | Protocolo FTP de transferencia de archivos |
| 21 | FTP Control | Protocolo FTP de transferencia de archivos |
| 22 | SSH | Protocolo de terminal remoto seguro |
| 23 | Telnet | Protocolo de terminal remoto |
| 25 | SMTP | Protocolo de transferencia de correo |
| 42 | Nameserver | Servidor de nombres |
| 43 | WHOIS | Protocolo de información |
| 53 | DNS | Servidor de nombres |
| 67 | Bootp Servidor | Protocolo de arranque a través de la red |
| 68 | Bootp Cliente | Protocolo de arranque a través de la red |
| 69 | TFTP Protocolo | FTP trivial |
| 80 | HTTP | Protocolo de transmisión de hipertexto |
| 88 | Kerberos | Protocolo de autentificación seguro |
| 109 | POP2 | Protocolo de gestión de correo electrónico |
| 110 | POP3 | Protocolo de gestión de correo electrónico |
| 137-139 | NetBIOS | Protocolo de la red Microsoft |
| 143 | IMAP | Protocolo de correo electrónico |
| 161, 162 | SNMP | Protocolo de administración de red |
| 194 | IRC | Internet Relay Chat |
| 256 | SNMP | Protocolo de administración de red |
| 389 | LDAP | Protocolo de localización de servicios |
| 443 | HTTPS | Protocolo HTTP seguro (cifrado) |
| 445 | SMB CIFS | Protocolo de la red Microsoft |
| 513 | Rlogin | Protocolo de terminal remoto |
| 514 | Rshell | Protocolo de terminal remoto |
| 515 | Impresora TCP/IP | Protocolo de una impresora de red |
| 520 | RIP | Protocolo de encaminamiento |
| 524 | NCP | Protocolo de la red Novell NetWare |
| 1080 | Sockets | Servidor proxy |
| 989 | FTP Datos sobre SSL | Protocolo FTP seguro (cifrado) |
| 990 | FTP Controlsobre SSL | Protocolo FTP seguro (cifrado) |
| 992 | Telnet sobre SSL | Protocolo Telnet seguro (cifrado) |
| 993 | IMAP sobre SSL | Protocolo IMAP seguro (cifrado) |
| 1745 | Winsock-proxy | Servidor proxy de Microsoft |
| 3306 | MySQL | Base de datos relacional |
| 2049-4045 | NFS | Sistema de archivos de red de Linux |
| 6000-6007 | X-Window | Interfaz gráfica de Unix/Linux |
| 6667 | IRCD | Internet Relay Chat |

Para poder conocer el estado de actividad de los puertos de comunicación se utilizan diferentes herramientas que envían mensajes a un rango de puertos especificado y comprueban si éstos están activos. Software de este tipo son *nmap* para sistemas Linux, o *GFI LANguard* para Windows.

Por regla general, cuantos más puertos de comunicación se encuentren activos o abiertos en nuestro equipo, mayor será el riesgo a padecer un problema de seguridad.

### 1.6.1. Protocolo UDP.

UDP (*User Datagram Protocol*) es un protocolo que no está orientado a la conexión. Esto implica que no se realiza una conexión previa a la transmisión y que no existe un control de flujo, de forma que se pueden entregar datagramas duplicados o desordenados. Hay que tener en cuenta que el receptor solo conoce la IP del emisor.

Se usa cuando es más importante la velocidad de la transmisión como es caso del *streaming* o vozIP, que la fiabilidad de la misma; o bien en protocolos de tipo petición respuesta, como son DHCP o DNS.

### 1.6.2. Protocolo TCP.

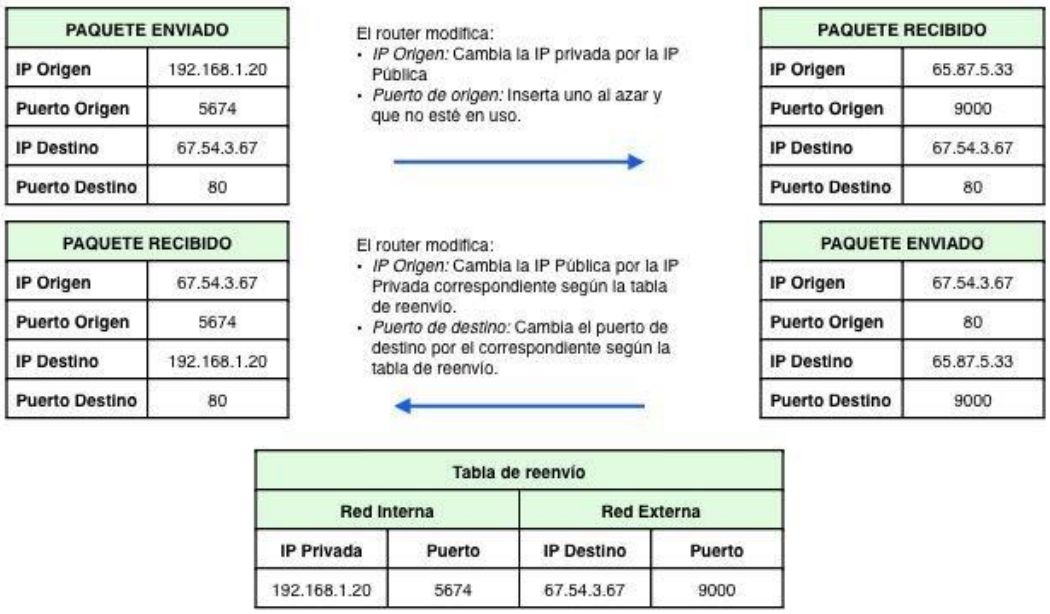
TCP (*Transmisión Control Protocol*) proporciona un servicio orientado a la conexión, lo que implica establecer una conexión antes de comenzar la transmisión y llevar un control de flujo y de errores. Con ello se consigue una transmisión fiable, aunque más lenta que con UDP.

Con TCP, cuando se desea conectar con un servicio de un equipo, hay que iniciar una conexión. Una vez establecida, cualquiera de los extremos puede empezar a transmitir y también terminarla cuando lo desee. La conexión se define de forma única indicando la dirección IP y número de puerto del equipo origen, y la dirección IP y número de puerto del equipo destino. A este par formado por dirección IP y número de puerto se le llama *conector* o *socket*. Se dice que un puerto de comunicación está *abierto* o *a la escucha* cuando existe un programa en el equipo que controla las comunicaciones que llegan a través del puerto.

## 1.7. Traducción de direcciones de red. NAT y PAT.

El aumento de la demanda de direcciones públicas para los equipos conectados a Internet, ha provocado que el espacio de direcciones IPv4 empiece a agotarse. Una de la técnicas más usadas para limitar el número de ordenadores con IP públicas conectadas a Internet es mediante la traducción de direcciones de red NAT (*Network Address Translation*).

NAT permite que varias direcciones IP privadas puedan acceder a Internet a través de una única IP pública. De esta forma los equipos de una intranet usan una única IP pública que estará asignada al router que les da salida a Intenet.



Para que este mecanismo funcione es necesario que el router que da acceso a Internet rescriba algunos datos en los datagramas que encamina. Según la información a modificar existen varios tipos de NAT:

- NAT (*Network Address Traslation*) básico: en este caso sólo se modifica la dirección IP. Es un NAT a nivel de red.
- NAPT/PAT (*Network Address Port Traslation*)/(*Port Address Traslation*). Se modifica la dirección de red y los puertos empleados en la comunicación. De hecho es el tipo NAT que más se utiliza y muchas veces se denomina simplemente NAT.

La información de los datagramas que ha de ser modificada al pasar por un router NAT (direcciones IP y puertos) se almacenan en una tabla NAT, de esta forma se pueden deshacer los cambios y recuperar los valores originales de las direcciones IP y puertos.

Cuando una máquina quiere establecer una conexión, el router NAT guarda su IP privada y el puerto de origen y los asocia a la IP pública y un puerto al azar. Cuando llega información a este puerto elegido al azar, el router comprueba la tabla y lo reenvía a la IP privada y puerto que correspondan.

## 1.8. Los Sistemas Operativos.

Un **sistema operativo** es un conjunto de programas que funcionan sobre una computadora y que se encarga de administrar sus recursos. Además, suelen incluir un conjunto de rutinas básicas que facilitan la tarea de desarrollo de aplicaciones, así como utilidades para los usuarios.

Desde el punto de vista de la gestión sobre los recursos de red, podemos dividir los sistemas operativos en dos tipos: **sistemas operativos de cliente** y **sistemas operativos de servidor**.

Un *sistema operativo de cliente* es aquél que no ha sido desarrollado con soporte para la administración de los recursos de la red aunque sí puede acceder a ellos como recursos de trabajo. Habitualmente requieren un menor espacio de almacenamiento en disco y menores requerimientos de memoria y capacidad de proceso. Por su parte, un *sistema operativo de servidor* es aquél especialmente diseñado para trabajar en una red y permitir la administración eficiente de sus recursos.

Aunque esta división todavía es aplicable a los sistemas operativos, resulta cada vez más confusa, ya que los sistemas operativos de cliente y servidor llegan a diferenciarse solamente por las aplicaciones y servicios que tienen instalados, no por el núcleo del sistema operativo.

### 1.8.1. Modelo cliente-servidor.

Cuando un equipo quiere acceder a los servicios disponibles en un servidor remoto, primero tiene que enviar un mensaje de solicitud y dirigirlo al puerto asociado a ese servicio. El servidor deberá tener activo ese puerto para recibir la solicitud, procesarla y enviar los resultados. En este modelo, el cliente debe conocer cuál es el número de puerto que tiene asociado ese servicio en el servidor, normalmente porque se trata de un *puerto bien conocido*.

Este modelo cliente-servidor se puede aplicar tanto a programas que se ejecutan en un mismo equipo como a equipos conectados a través de la red. Este modelo no restringe la función que desempeña cada equipo en la red, de forma que un equipo se puede comportar como cliente de unos determinados servicios y a la vez comportarse como servidor de otros.

El modelo funciona muy bien en redes donde los servicios se gestionan a través de servidores centralizados, y donde pueden existir varios servidores que repartan el trabajo de diferentes tareas.

Las características que definen a un equipo como cliente son:

- Requiere de una potencia de cálculo menor, aunque esto dependerá de los programas que ejecute.
- Es utilizado por los usuarios para realizar su trabajo diario.
- Es el encargado de iniciar las peticiones o solicitudes.
- Recibe las respuestas de los servidores, obteniendo la información que ha solicitado o un mensaje de rechazo.
- Puede realizar peticiones de varios servicios a diferentes servidores.

Las características que definen a un equipo como servidor son:

- Suele requerir una gran potencia de cálculo y memoria principal para poder atender todas las peticiones que recibe.

- Permanece a la espera de recibir peticiones. Cuando recibe una petición, la procesa y envía los resultados o un mensaje de rechazo.
- Suele aceptar un gran número de peticiones, aunque este valor puede limitarse.
- Es un equipo dedicado a atender peticiones y los usuarios no suelen trabajar con él directamente.

Entre las ventajas del uso del modelo cliente-servidor en los servicios de red podemos destacar:

- Se establece un mayor control de la seguridad y el acceso a servicios autorizados, ya que éste se realiza a través de cada servidor.
- Puede aumentarse fácilmente la capacidad de los equipos o su número.
- Permite un mantenimiento más sencillo y una división de responsabilidades entre los administradores, ya que los cambios solamente se deben realizar en los servidores involucrados.

El modelo cliente-servidor también tiene sus inconvenientes:

- Sobrecarga de los servidores cuando existen muchas peticiones.
- El mal funcionamiento de un servidor hace que no estén disponibles los servicios que ofrece.
- Los servidores requieren de sistemas operativos y programas muy estables.

## 1.8.2. Herramientas de administración de servicios.

Los sistemas operativos disponen de una serie de programas y herramientas que facilitan la instalación, configuración y administración de los servicios de que disponen.

### 1.8.2.1 Herramientas en Microsoft Windows

En los sistemas Windows Server se utilizan las denominadas *Herramientas administrativas*. Algunas de las opciones que aparecen por defecto en el menú de *Herramientas administrativas* no están disponibles por defecto y se añadien cuando se instalan los servicios correspondientes.

Algunas de las herramientas más importantes que incluye Windows Server son las siguientes:

**Administración de equipos**: se usa para la administración del equipo local o equipos remotos. Desde aquí se puede realizar la gestión completa de un equipo e incluye las herramientas del sistema (*visor de sucesos*, *administrador de dispositivos*, etc.), almacenamiento (*desfragmentador de discos*, *administrador de discos*, etc.) y servicios y aplicaciones instaladas (*DHCP*, *DNS*, *IIS*, etc.).

**Administrador de Internet Information Services (IIS)**: se usa para administrar el Servidor de Información de Internet (IIS) que incluye los servicios Web, FTP y correo saliente (SMTP).

**Administrador del servidor**: mediante Administre su servidor o el Asistente para configurar su servidor se puede instalar, desinstalar y configurar servicios en el equipo.

**DHCP**: se usa para configurar el servidor DHCP del equipo local.

**DNS**: se usa para configurar el servidor DNS del equipo local.

**Directiva de seguridad local**: se utiliza para establecer características de seguridad del equipo cuando éste no pertenece a ningún dominio.

**Directiva de seguridad de dominio**: permite establecer características de seguridad sobre el dominio.

**Directiva de seguridad del controlador de dominio**: se utiliza para establecer características de seguridad sobre el servidor del dominio.

**Servicios**: muestra una lista con todos los servicios iniciados en el sistema y los que están inactivos.

**Servicios de Escritorio Remoto**: incluye las herramientas para el acceso remoto a equipos.

**Programador de tareas**: se utiliza para definir qué programas se van a ejecutar en el equipo en una fecha o con una frecuencia determinada.

**Visor de eventos** o **sucesos**: muestra información sobre los sucesos, avisos o errores que se han producido mientras el equipo ha estado en funcionamiento.

En Windows Server es posible realizar tareas de administración sin necesidad de estar sentado delante del servidor. Para ello, se pueden utilizar los **Servicios de Escritorio Remoto** que permiten realizar una conexión al servidor desde un equipo remoto. El equipo desde donde se realiza la conexión puede ser cualquier versión de Microsoft Windows, Unix/Linux o Apple.

#### 1.8.2.2 Herramientas en GNU/Linux

Tradicionalmente, estas herramientas no han existido y todas las tareas se realizaban modificando los archivos de configuración correspondientes de los servicios o usando órdenes. Esta técnica tiene la ventaja de que permite modificar cualquier parámetro, está más estandarizada y se puede utilizar de la misma forma en diferentes distribuciones y versiones (normalmente los nombres y las ubicaciones de los archivos pueden variar).

No obstante, tiene el inconveniente de que hay que tener mucho cuidado a la hora de manejar las órdenes o modificar los archivos, ya que cualquier error de sintaxis puede hacer que el servicio afectado o todo el sistema deje de funcionar correctamente.

Actualmente, han aparecido una serie de herramientas que facilitan la configuración y administración del sistema. Algunas de ellas son muy dependientes de la distribución de Linux utilizada y puede que no incluyan todas las opciones de configuración.

Todas las modificaciones en la configuración del sistema deben realizarse utilizando la cuenta de usuario que tiene privilegios para realizar estas operaciones. Esta cuenta, que normalmente es **root**, dispone de acceso a las órdenes, archivos de configuración y herramientas gráficas de administración. Por seguridad, el uso de esta cuenta debe estar limitado a realizar este tipo de acciones y utilizar una cuenta de usuario normal para el resto de operaciones que no requieran de permisos especiales.

Si un usuario normal intenta acceder a cualquier herramienta que requiera privilegios de *root*, el sistema nos solicitará la contraseña de esta cuenta de administrador antes de acceder a ella. En caso de que necesitemos ejecutar una orden en la línea de órdenes con los privilegios de administrador, podemos cambiarnos a *root* utilizando la orden *su* (switch user) de la forma:

```
paquito@equipo:~$ su -
```

(Ojo, si no ponemos el guión cambiaremos a *root*, pero seguiremos en el "home" y con las variables de entorno del usuario activo de la sesión; en el ejemplo *paquito*).

A veces puede resultar más útil la posibilidad de ejecutar determinados programas con los privilegios de *root* mediante *sudo* (switch user do**)**. Aunque por seguridad Ubuntu trae desactivada la cuenta del *root*, para poder administrar el sistema existe un grupo de usuarios denominado *sudoers users* los cuales pueden obtener permisos de *root* mediante la utilización de *sudo*. Como el usuario con el que se instala Ubuntu se encuentra incluido en este grupo de administradores, podrá obtener permisos de *root*.

La forma usual de usar el comando sudo, es anteponiéndolo a la orden o comando a ejecutar:

```
paquito@equipo:~$ sudo comando
```

Algunas veces, cuando vamos a ejecutar muchos comandos como *root*, podemos cambiar al usuario *root*, para así ahorrarnos escribir el *sudo* en cada línea de órdenes. Pero atención, se debe usar con la opción -i para cambiar las variables de entorno y el "home" al de *root*.

```
paquito@equipo:~$ sudo -i
```

Cada distribución Linux suele incorporar unas herramientas específicas de configuración. Así RedHat dispone de **control-panel**. Fedora también incorpora este panel de control pero integrado en los menús. SUSE utiliza la herramienta **YaST.**

Además es posible utilizar otras herramientas de administración que pueden utilizarse en la mayoría de las distribuciones. **Linuxconf** permite administrar la configuración de la red, los usuarios y los grupos del sistema local, el acceso a particiones locales y remotas, el modo de arranque del sistema, etc. No obstante esta utilidad no dispone de algunas opciones básicas de configuración.

Una buena herramienta de configuración gráfica es **Webmin**. Es una aplicación web que funciona en más de 35 sistemas Unix/Linux diferentes y es multilenguaje. La forma de acceder a Webmin una vez instalada, es a través de un explorador web con dirección https://localhost:10000. Este método permite también acceder a Webmin desde otro equipo distinto al que se encuentra instalado usando la IP en la URL de conexión.

La herramienta Webmin está construida mediante módulos agrupados en categorías, lo que flexibiliza su configuración y actualización. Las categorías por defecto en Webmin son las siguientes:

- Webmin: incluye los módulos de configuración propios de Webmin. Desde esta opción se puede configurar el propio programa Webmin (como el idioma, las cuentas de usuario que van a tener acceso a este programa, etc.).

- Sistema: aquí se encuentran los módulos relacionados con la configuración del sistema operativo. A este grupo pertenecen las opciones de configuración de los programas que se inician automáticamente en el arranque, instalar o desinstalar aplicaciones, gestionar cuentas de usuario y grupos, administrar sistemas de archivos locales y en red, etc.
- Servidores: incluye los módulos de configuración de la mayoría de programas servidores que se pueden instalar en el equipo. Aquí se incluyen opciones para compartir archivos con equipos Windows mediante Samba, configurar un servidor de correo electrónico, servidor de páginas Web, servidor de bases de datos, etc.
- Trabajando en red: aquí podemos encontrar los módulos de configuración de red del sistema, además de otros servicios como NFS, programa cortafuegos, etc.
- Hardware: esta opción incluye los módulos relacionados con la configuración de dispositivos hardware del sistema, como impresoras, discos duros, grabadoras de CD y DVD, sistemas RAID, etc.
- Clúster: incluye los módulos de configuración y administración de un sistema clúster (varios equipos conectados en red que funcionan como si fuera uno solo).
- Otros: esta categoría incluye otros módulos adicionales, entre los que podemos encontrar un administrador de archivos, línea de órdenes, órdenes ejecutadas habitualmente, etc.

Cuando se instala Webmin, éste comprueba qué programas se encuentran instalados y activa aquellos módulos estándares relacionados. Si algún módulo estándar no es necesario porque la herramienta que configura no está instalada, entonces éste aparecerá en un apartado denominado "Módulos no utilizados" (*Un-used Modules*), dentro de la página principal de Webmin. Si Webmin se ha instalado con anterioridad a alguna herramienta que puede ser gestionada con un módulo estándar, entonces éste permanecerá en el grupo de módulos no utilizados hasta que se pulse en la opción "Refrescar módulos" (*Refresh Modules*), momento en el que pasará a formar parte de la categoría correspondiente.

Además de estas herramientas, las distribuciones de Linux incluyen otras muchas que agilizan las tareas de administración del sistema, aunque su utilidad se reduce a parámetros específicos

## 1.8.3. Instalación de programas.

Cuando se necesita que un equipo ofrezca algún servicio, es necesario instalar en él todos los programas necesarios, que suelen estar disponibles a través de uno o varios archivos. Estos archivos suelen contener un programa de instalación que guía paso a paso, una herramienta de configuración del servicio, el proceso demonio encargado de atender las peticiones, y varios archivos de configuración o valores del registro del sistema.

A la hora de instalar un determinado servicio en el sistema, podemos hacerlo de varias formas: abriendo el programa ejecutable de instalación o mediante la herramienta de instalación de programas.

### 1.8.3.1. Instalación de componentes en Microsoft Windows.

En Windows existen dos tipos de programas que pueden instalarse: los *componentes de Windows* (programas y utilidades que se incluyen en el disco de instalación del sistema operativo) y las *aplicaciones*.

Las **aplicaciones** se instalan y desinstalan en Windows, dependiendo de la versión, mediante "Agregar o quitar programas" o "Aplicaciones y características", que se encuentra disponible desde el icono *Programas* del *Panel de control*. Sin embargo la forma más común de instalar una aplicación es ejecutar directamente el programa de instalación de la propia aplicación.

Los **componentes** de Windows se instalan o desinstalan desde varias ventanas del sistema dependiendo de su tipo. Los componentes de red de Windows (servicios, protocolos y clientes) se gestionan desde las propiedades del icono "Red" del *Escritorio* o del *Menú de Inicio* (Windows 2000/XP/2003) o desde el *Centro de redes y recursos compartidos* (Windows Vista/7/2008/10/2012). El resto de componentes se instalan, al igual que las aplicaciones, desde el icono "Agregar o quitar programas" (Windows 2000/XP/2003) o desde "Aplicaciones y características" (Windows Vista/7/2008/10/2012), accesibles desde el *Panel de control*.

### 1.8.3.2 Instalación de paquetes en GNU/Linux.

A la hora de instalar o desinstalar programas y aplicaciones en Linux, es conveniente conocer una serie de términos:

- Paquete: está formado por un conjunto de archivos o ficheros que se distribuyen conjuntamente, de forma que es la unidad mínima de instalación en un sistema operativo. Para evitar complejidad y facilitar las tareas de administración, no se permite que un paquete se instale parcialmente en el sistema. Es posible que un programa esté distribuido en varios paquetes y que un paquete contenga varios programas. El paquete se almacena en un archivo con una extensión concreta para identificar su tipo (por ejemplo, ".rpm" o ".deb") y contiene, además de los archivos a copiar, las rutas donde se copian, información sobre la persona que los ha desarrollado, una clave cifrada que asegura su autenticidad, la suma de verificación para comprobar si ha sido alterado, las operaciones para configurar el programa, etc.
- Repositorio o fuente de instalación: se trata de un gran almacén que guarda los archivos con los paquetes para instalar. La fuente de instalación más común siempre fue el soporte CD o DVD, pero actualmente se usan los repositorios oficiales y no oficiales disponibles en servidores de Internet.
- Dependencia: en muchas ocasiones, un paquete puede requerir de otro para poder funcionar correctamente (porque dispone de alguna librería o archivo que necesita). En este caso, es necesario que el paquete requerido sea instalado antes, para poder mantener la estabilidad del programa que queremos instalar e incluso la del sistema.
- Conflicto: se producen cuando queremos instalar un paquete que no puede coexistir al mismo tiempo con otros paquetes a instalar o ya instalados. Ocurren normalmente porque se trata de programas que realizan funciones similares, con lo que su uso simultáneo puede acarrear problemas.
- Gestor de paquetes: es el programa encargado de gestionar los paquetes de instalación en el sistema, ofreciendo operaciones como: instalar, desinstalar y actualizar paquetes. Sus tareas son: gestionar una base de datos con información sobre los paquetes instalados y los disponibles para instalar, la selección de las fuentes de instalación, la consulta de información sobre los paquetes instalados y no instalados, gestionar correctamente las dependencias, etc.

Existen básicamente dos métodos para instalar programas en Linux:

**A través de los paquetes fuente**: consiste en utilizar paquetes que contienen los archivos con el código fuente del programa. Por lo tanto, en la operación de instalación del paquete es necesario copiar los archivos, compilarlos a código ejecutable y configurarlos convenientemente. Esto requiere que el usuario tenga conocimientos sobre el sistema, aunque la operación de compilación y configuración se realiza ejecutando una orden o un programa. Sin embargo, tiene la ventaja de que el paquete no depende de la distribución o versión de Linux

**A través de paquetes precompilados**: se utilizan paquetes que previamente han sido compilados en código ejecutable, lo que facilita enormemente las tareas de instalación automática. Sin embargo, es necesario disponer del paquete específico que ha sido diseñado para funcionar en una distribución concreta sobre un equipo con una arquitectura concreta.

Algunos de los gestores de paquetes más utilizados actualmente son *dpkg* (utilizado por Debian y otras distribuciones derivadas), *RPM* (es el sistema más extendido que se ha convertido en un estándar para las distribuciones Red Hat, Fedora, SuSE, Mandriva, etc.), *tgz* (usado por Slackware), *Pacman* (usado en la distribución Arch) y *Ebuild* (usando en la distribución Gentoo).

Una herramienta que se puede utilizar para la gestión de paquetes es *Webmin*, que tiene la ventaja de que funciona en cualquier versión de Linux.

Las distribuciones Debian y sus derivadas, utilizan un sistema de gestión de paquetes almacenados en archivos con extensión ".deb". Los paquetes ".deb" se gestionan a través de dos herramientas que funcionan en niveles diferentes: por un lado está **dpkg**, que funciona en un nivel más bajo, mientras que por otro lado está **apt**, que funciona en un nivel más alto.

Esta jerarquía hace que *dpkg* proporcione todo lo necesario para manipular paquetes, mientras que *apt* utiliza la anterior para permitir que el usuario trabaje de una forma más cómoda, ofreciendo funciones para obtener paquetes desde diferentes lugares o resolver dependencias complejas.

El proyecto Debian mantiene para cada una de las versiones que publica, tres tipos de distribuciones que se utilizan en las fases de desarrollo: la denominada **Estable** (*stable*), es la distribución que se recomienda, ya que se han depurado todos los errores (o por lo menos todos los conocidos). **De prueba** (*testing*), es utilizada por los que quieren disponer cuanto antes de la última versión, aunque todavía no se han corregido algunos pequeños errores detectados. **Inestable** (*unstable*), es la distribución utilizada por los programadores que forman parte del proyecto Debian durante todo el desarrollo activo del sistema.

La herramienta **apt** conforma un entorno completo de gestión de paquetes que realiza de forma automática la mayoría de operaciones complejas para el usuario. Esta herramienta dispone de muchas órdenes que se pueden ejecutar en el intérprete:

| Orden | Descripción |
|---|---|
| *apt-cache* | Manipula los archivos de los paquetes disponibles en cache. |
| *apt-cdrom* | Permite incluir la unidad de CD-ROM como fuente de instalación de paquetes. |
| *apt-config* | Permite acceder a la configuración de las herramientas apt (archivo apt.conf). |
| *apt-file* | Muestra a qué paquetes pertenecen los archivos indicados. |
| *apt-get* | Permite la instalación, desinstalación y actualización de paquetes. |
| *apt-key* | Gestiona la lista de claves utilizada para autentificar los paquetes descargados para instalar. |
| *apt-rpm* | Se trata de una versión modificada de la herramienta apt que pretende funcionar con el gestor de paquetes RPM. (En desarrollo). |
| *auto-apt* | Instala paquetes de forma automática cuando se ejecuta un programa que necesita un determinado paquete. |
| *localepurge* | Elimina de forma automática aquellos archivos de la documentación del sistema que pertenecen a idiomas no utilizados por los usuarios. |

Ejemplos de utilización de la herramienta **apt**:

- Para actualizar la base de datos de paquetes disponibles para instalación, nuevas versiones y actualizaciones de paquetes ya instalados, se usa esta orden con privilegios de administrador:

  ```
  # apt-get update
  ```

- Para actualizar con nuevas versiones los paquetes ya instalados

  ```
  # apt-get upgrade
  ```

- Para instalar el paquete "manolete" (si existieran dependencias, se resolverían confirmando el mensaje de advertencia que aparece):

  ```
  # apt-get install manolete
  ```

- Para reinstalar el paquete "manolete" dañado, o actualizarlo con una nueva versión disponible:

  ```
  # apt-get  --reinstall install manolete
  ```

- Para instalar el paquete "manolete" en la última versión todavía de prueba:

  ```
  # apt-get install manolete /testing
  ```

- Para eliminar el paquete "manolete" y mantener los archivos de configuración. (*apt* se encargará de eliminar también los paquetes dependientes, después de recibir confirmación):

  ```
  # apt-get remove manolete
  ```

- Para eliminar el paquete "manolete" y todos los archivos de configuración:

  ```
  # apt-get purge manolete
  ```

- Para actualizar a una nueva versión de Linux disponible (debe estar disponible la fuente donde se encuentran los paquetes de la nueva versión):

  ```
  # apt-get dist-upgrade
  ```

- Para eliminar los paquetes temporales que han sido descargados al equipo:

  ```
  # apt-get clean
  ```

- Para eliminar los paquetes antiguos que han sido descargados al equipo:

  ```
  # apt-get autoclean
  ```

- Para solicitar a *apt* que realice una recomendación para instalar, desinstalar y eliminar paquetes, de forma que se tengan en cuenta las sugerencias incluidas con los paquetes:

  ```
  # apt-get –u dselect-upgrade
  ```

- Para obtener una lista de los paquetes que pueden ser actualizados a nuevas versiones:

  ```
  # apt-show-versions –u
  ```

- Para obtener una lista de paquetes disponibles para instalar cuyo nombre o descripción cuadre con la cadena de texto "mail":

  ```
  $ apt-cache search mail
  ```

- Si se desea obtener información específica sobre el paquete disponible para instalar "apache2" (se mostrará información de todas las versiones disponibles o instaladas):

  ```
  $ apt-cache show apache2
  ```

- Para obtener una lista con los paquetes de que depende "apache2":

  ```
  $ apt-cache depends apache2
  ```

- Para conocer qué paquetes instalados o disponibles para instalar contienen archivos con determinados nombres:

  ```
  $ apt-file archivo
  ```

- También se puede obtener una lista de los archivos que contiene el paquete "manolete" con esta orden:

  ```
  $ apt-file list manolete
  ```

La herramienta *apt* guarda en el archivo *sources.list* (normalmente ubicado en la carpeta */etc/apt*) una lista con las fuentes de instalación desde donde se obtienen los paquetes. Este archivo de texto contiene una línea por cada una de las fuentes disponibles y, para cada una de estas líneas, se indican los siguientes campos separados por espacios en blanco:

**Tipo de archivos**: indica el tipo de archivos que contiene la fuente de instalación: archivos binarios Debian (*deb*), archivos fuente Debian (*debsrc*), archivos RPM (*rpm*), archivos fuente RPM (*rpm-src*), etc.

**URL** : dirección de la fuente de instalación, que puede ser de varios tipos: *http*, *ftp*, *file*, *ssh*, etc.

**Argumentos**: las opciones indicadas en este campo dependen del tipo de archivos y de la distribución utilizada. Por ejemplo, se puede indicar si se trata de un paquete que pertenece a la distribución oficial y con soporte por parte de Ubuntu (*main*) o si el soporte lo ofrece la comunidad (*universe)*. También existen paquetes que no tienen una licencia libre (*restricted*) y los que no tienen ningún tipo de soporte (*multiverse*).

El archivo *sources.list* se puede modificar para especificar diferentes fuentes de instalación de los paquetes del sistema. También se pueden eliminar los comentarios "#" de las fuentes que disponen de paquetes en código fuente para instalar éstos en vez de los paquetes precompilados.

En Ubuntu se dispone del *Centro de software de Ubuntu*. Es un "front end" gráfico de alto nivel para el sistema de gestión de paquetes apt/dpkg. Permite buscar, instalar y desinstalar aplicaciones y añadir repositorios de terceros para instalar aplicaciones que no se encuentren en los repositorios oficiales.

Las distribuciones de Linux disponen de mecanismos muy potentes para la gestión de actualizaciones del sistema. Estas actualizaciones son muy importantes, ya que permiten instalar las nuevas versiones de los paquetes con correcciones a problemas y fallos de seguridad de versiones anteriores.

### 1.8.4. Gestión e inicio de servicios.

Cuando un sistema operativo se inicia, debe arrancar también determinados servicios, tanto para uso interno del sistema operativo, como para aquéllos que pueden ser accedidos desde otros equipos. Cada uno de estos servicios es controlado por uno o varios procesos que se inician en el arranque y permanecen en ejecución mientras que el equipo está encendido. Si uno de estos procesos no se inicia o termina, entonces el servicio que ofrece dejará de estar disponible.

Antes de entrar en detalles, recordemos cómo es el proceso de arranque de un sistema.

Al iniciar el ordenador, la BIOS busca en su configuración el dispositivo de arranque por defecto, el cual suele ser un disco duro, Entonces carga en memoria el primer sector del dispositivo de arranque y le transfiere el control. Cuando se trata de un disco duro, este primer sector es el Master Boot Record (*MBR*) conteniendo, además del código cargado por la BIOS, la tabla de particiones del disco. El código del MBR, lee en la tabla de particiones cuál es la partición activa y carga en memoria el primer sector de dicha partición, transfiriéndole el control.

Normalmente este sector estará ocupado por un gestor de arranque que nos permitirá elegir cual sistema operativo debe arrancar. (Opcionalmente, puede instalarse el gestor de arranque en el MBR,

tomando antes el control). Una vez cargado el gestor de arranque, éste se ejecuta, busca el kernel del sistema operativo en una posición conocida del disco, lo carga en memoria y le cede el control.

A partir de aquí, dependiendo de la arquitectura concreta de kernel, éste operará de una u otra manera, aunque la finalidad es básicamente la misma: iniciar una lista de comprobaciones y activar una serie de módulos internos del mismo para gestionar la memoria, el tiempo de procesador y el acceso a los periféricos.

La mayoría de los sistemas operativos ofrecen la posibilidad de arrancar con diferentes configuraciones y servicios, lo que permite recuperar el sistema de algunos fallos producidos.

#### 1.8.4.1 Modo seguro de Microsoft Windows.

El **modo a prueba de errores** que incorporan los sistemas Microsoft Windows permite iniciar el sistema operativo con diferentes configuraciones. Estas configuraciones cargan unos determinados controladores de dispositivos y servicios.

Para poder seleccionar los diferentes modos a prueba de errores hay que pulsar la tecla F8 en el inicio del proceso de arranque del sistema, antes de que comience la carga del sistema operativo. Los diferentes modos a prueba de errores que incluye Microsoft Windows son:

- Modo seguro.
- Modo seguro con funciones de red.
- Modo seguro con símbolo de sistema.
- Habilitar el registro de inicio o de arranque.
- Habilitar el modo VGA o vídeo de baja resolución.
- La última configuración válida conocida.
- Modo de restauración de servicios de directorio (Solamente en Windows Server 2003).
- Modo de depuración.
- Deshabilitar el reinicio automático en caso de error del sistema.
- Deshabilitar el uso obligatorio de controles firmados
- Iniciar Windows normalmente.

Existe otra herramienta que gestiona los procesos en ejecución, denominada **Servicios**. Desde ella se puede establecer si un determinado proceso instalado en el sistema se inicia o se detiene, haciendo que el servicio correspondiente esté disponible o no.

También se puede utilizar la combinación de teclas Ctrl+Alt+Supr para acceder al **Administrador de tareas**, que muestra una lista con los procesos en ejecución y las opciones para detenerlos.

#### 1.8.4.2 Niveles de ejecución en GNU/Linux.

Los sistemas Linux disponen de mecanismos para iniciar el sistema con diferentes configuraciones. Estas configuraciones están basadas en determinar qué procesos se activarán en el arranque.

Para ver la lista de programas en ejecución en el sistema se puede ejecutar la orden `ps -ef`

Si se modifican los programas que se inician en el arranque podemos conseguir que Linux trabaje de forma distinta y ofrezca un diferente conjunto de servicios.

Actualmente existen básicamente tres sistemas de inicialización (*init systems*) en Linux. Hasta hace unos años, **System V init** era el único, pero sus carencias lo han convertido en un sistema obsoleto en la mayoría de distribuciones. Ahora mismo, casi todas las distribuciones se están pasando a **systemd.** Todavía tenemos otro, **upstart**, que fue desarrollado por el personal de Ubuntu, pero después de que Debian optara por utilizar **systemd**, Ubuntu decidió abandonarlo.

Algunas distribuciones y los sistemas de inicialización utilizados por defecto son:

| **system V** | **upstart** | **systemd** |
|---|---|---|
| Debian 6 (y anteriores) | | Debian 7 → |
| Ubuntu 9 (y anteriores) | Ubuntu 10 → Ubuntu 14 | Ubuntu 15 → |
| CentOS 5 (y anteriores) | CentOS 6 | CentOS 7 → |
| Fedora 9 (y anteriores) | Fedora 10 → Fedora 14 | Fedora 15 → |

Actualmente **systemd** se ha convertido en el demonio de inicialización de facto para la mayoría de distribuciones Linux, y es compatible hacia atrás con los comandos y los scripts de inicialización de **System V init**, lo que significa que cualquier servicio **System V** correrá también bajo **systemd**.

**Sistema System V init**

Cuando se inician los sistemas GNU/Linux y Unix basados en el clásico System V, después de que los *drivers* se hayan cargado, el kernel ejecuta el proceso especial llamado **init**. La misión de init es ejecutar el resto de procesos del sistema: comprobación de discos, detección/configuración de hardware adicional, apertura de terminales, servidores, etc.

El proceso init se convierte en el proceso número 1 (PID 1) y se encarga de ejecutar el resto de programas que hacen que el sistema funcione. En realidad init no hace mucho, sino que se limita a ejecutar una serie de guiones o scripts que activan ordenadamente los diferentes servicios que hacen funcionar el sistema.

El proceso init permite definir diferentes configuraciones, cada una de ellas iniciará un conjunto distinto de procesos. A estas configuraciones se les conoce como **niveles de ejecución**. De esta forma, el usuario puede seleccionar sobre qué configuración o nivel de ejecución va a iniciar el sistema.

Los niveles de ejecución predefinidos en la mayoría de las versiones de Linux son los siguientes:

**0** (**Parada de sistema**): se utiliza para apagar el equipo.

**1** (**Modo monousuario**): se accede al sistema en modo monousuario (sólo puede entrar un único usuario al sistema en un momento dado) sobre una consola de línea de órdenes y sin servicios de red.

**S** (**Modo monousuario**): igual que el modo anterior pero con el teclado configurado en idioma inglés.

**2** (**Modo multiusuario local sin red**): se accede al sistema sobre una consola de línea de órdenes y sin servicios de red. Varios usuarios pueden acceder al mismo tiempo al sistema.

**3** (**Modo multiusuario completo con red**): se accede al sistema sobre una consola de línea de órdenes con todos los servicios.

**4** (**libre**, no utilizado): este nivel no está configurado, así que puede definirse a gusto del usuario.

**5** (**Modo multiusuario completo con red y entorno gráfico**): éste es el nivel por defecto que inicia el sistema con todos los servicios y en modo gráfico.

**6** (**Reiniciar el sistema**): se utiliza para reiniciar el equipo.

Existe también otro nivel de ejecución que no se comporta como tal y que se llama "B" (*Boot* o de arranque). Los programas que se inician en este nivel de ejecución solamente están activos cuando el sistema está arrancando y se desactivan una vez que éste es operativo. Por ejemplo, el proceso *boot.sysctl* se usa para establecer los parámetros por defecto que va a utilizar el sistema una vez que ha arrancado. Estos parámetros se establecen en el archivo de configuración /etc/sysctl.conf.

El proceso init mantiene su configuración básica en el archivo /etc/inittab. Para seleccionar el nivel de ejecución que se va a utilizar en el arranque se puede hacer de dos formas:

- Editar el archivo /etc/inittab, usando el usuario *root* y un programa de edición de textos, buscar una línea con el texto "id:5:initdefault:" y sustituir el número 5 por el nivel de ejecución que deseamos sea el nuevo nivel por defecto. La próxima vez que se inicie Linux arrancará en ese nivel de ejecución.
- Utilizando una herramienta de gestión de los niveles de ejecución.

Es posible cambiar en cualquier momento de un nivel de ejecución a otro, utilizando la orden init, pero con privilegios de *root*. Por ejemplo, la siguiente orden cambia el nivel de ejecución actual al 3:

```
# init 3
```

La parada del sistema se realiza con comando shutdown. Este comando soporta varios parámetros:

| | |
|---|---|
| -h | detener el sistema |
| -r | reiniciar el sistema |
| now | ahora |
| +5 | dentro de 5 minutos |
| 14:00 | a las 14:00 horas |

Existen varios enlaces simbólicos a shutdown para detener y reiniciar el sistema de forma inmediata:

```
halt        /sbin/shutdown -h now
reboot      /sbin/shutdown –r now
```

También podemos, directamente, ir al runlevel 0 para detener el sistema.

```
root@linux:~# init 0
```

**Sistema Upstart**

Algunas distribuciones de Linux como Ubuntu, modificaron el mecanismo de arranque sustituyendo el proceso init por el proceso Upstart. Es un modelo basado en eventos lo que permite responder a éstos de una forma asíncrona cuando se generan.

Upstart es compatible con System V de forma que puede ejecutar sus scripts, pero usa sus propios archivos de configuración ubicados en /etc/init para determinar los servicios que deberán arrancarse como eventos. Todos estos archivos usan como nombre servicio.conf.

Aunque con Upstart ya no existe el archivo /etc/inittab, debido a la compatibilidad con System V podemos crearlo y colocar en su interior

```
id:N:initdefault:
```

Donde N es el runlevel por defecto que queremos establecer para el arranque del sistema.

**Sistema systemd**

El mecanismo **systemd** ha sido creado para ofrecer un inicio más rápido y flexible que System V, permitiendo el arranque en paralelo de los servicios y su inicio basado en la detección de conexiones de nuevas unidades externas.

Hasta ahora con System V el primer proceso en ejecutarse era el programa **/sbin/init**, cosa que ha cambiado en systemd a favor de **/lib/systemd/systemd.**

Mediante systemd se inicia y supervisa todo el sistema y se basa en la noción de unidades (*units*). Existen diferentes tipos de unidades: servicio (***.service***), punto de montaje (***.mount***), dispositivo (***.device***) socket (***.socket***), target (***.target***), etc. Una unidad concreta está compuesta de un nombre (el nombre del demonio) y de una extensión. Será la extensión la que indique de que tipo de unidad se trata.

Cada unidad tiene su correspondiente archivo de configuración cuyo nombre es idéntico. Por ejemplo la unidad de tipo servicio **httpd.service** tiene como archivo de configuración *httpd.service*. Los diferentes archivos de unidades disponibles en nuestro sistema podemos encontrarlos en **/usr/lib/systemd/**, en **/etc/systemd/system/** y en **/lib/systemd/system** dependiendo de si la unidad ha sido instalada por un paquete o es propia del sistema.

La utilidad de administración de las unidades de systemd es **systemctl**, con la cual podremos arrancar, parar, recargar servicios, activar o desactivar servicios en el arranque, listar el estado de los servicios, etc.

En systemd existen varios tipos de unidades, pero hay una de ellas denominada *target* que es la que se corresponde con los *runlevels* (0123456) de System V. Cada *target* reciben un nombre (en vez de un número) para identificar un propósito específico. La siguiente tabla muestra la similitud entre algunos de los *target* con los *runlevels* de System V:

| System V | systemd | Uso |
|---|---|---|
| 0: | runlevel0.target o poweroff.target | Apaga el sistema |
| 1 o S: | runlevel1.target o rescue.target | Nivel mono-usuario |
| 2: | runlevel2.target o multi-user.target | Multi-usuario. Idéntico al nivel 3 |
| 3: | runlevel3.target o multi-user.target | Multi-usuario |
| 4: | runlevel4.target o multi-user.target | Multi-usuario. Idéntico al nivel 3 |
| 5: | runlevel5.target o graphical.target | Multi-usuario con servidor gráfico |
| 6: | runlevel6.target o reboot.target | Reinicia sistema |
| emergency | emergency.target | Intérprete de mandatos de emergencia. |

¿Cómo cambiar en systemd el *target (runlevel)* predeterminado en el arranque por otro distinto?. El *target* **/lib/systemd/system/default.target** es el *target* predeterminado de arranque, y en realidad es un enlace simbólico que por defecto apunta a **/lib/systemd/system/graphical.target** por lo que para cambiar de *target* de arranque, bastará con eliminar dicho link y crear uno nuevo apuntando al nuevo *target*.

Otra forma, es usar el comando systemctl. Por ejemplo para cambiar al nivel 3 haríamos

```
# systemctl enable multi-user.target
```

#### 1.8.4.3 Gestión de servicios en GNU/Linux Ubuntu.

Dependiendo del sistema de gestión de servicios, se usaran diferentes comandos. La siguiente tabla muestra las equivalencias en los diferentes sistemas system V, Upstart y systemd:

| | **System V (init)** | **Upstart** | **systemd** |
|---|---|---|---|
| **Listar servicios** | ls /etc/init.d/ | initctl list | systemctl list-unit-files --type=service |
| **Gestionar servicios** | /etc/init.d/nombre_servicio {status \| stop \| start \| restart \| reload \| …}<br><br>o bien<br><br>service nombre_servicio {status \| stop \| start \| restart \| reload \| …} | {status \| stop \|start \| restart \| reload \| … } nombre_servicio<br><br>o bien<br><br>service nombre_servicio {status \| stop \| start \| restart \| reload\| … } | systemctl {status \| stop \| start \| restart \| reload \| … } nombre_servicio |
| **Habilitar /deshabiltar servicios en el arranque** | update-rc.d nombre_servicio {enable\|disable} | Editar el archivo /etc/init.d/nombre_servicio.override con el texto "manual" | systemctl {enable \| disable} nombre_servicio |
| **Ubicación de los scripts** | /etc/init.d | /etc/init | /lib/systemd/system<br><br>/usr/lib/systemd |

Los servicios basados en System V se manipulan de la siguiente manera.

```
$ sudo /etc/init.d/NOMBRE_SERVICIO {stop | start | status | ...}
```

Los servicios basados en Upstart se manipulan mediante initcl

```
$ sudo {stop | start | status | ...}  NOMBRE_SERVICIO
```

No obstante, ambos tipos de servicios se pueden manipular con el comando service:

```
$ sudo service NOMBRE_SERVICIO {stop | start | status | ...}
```

**Los servicios basados en systemd se manipulan con systemctl**

```
$ sudo systemctl NOMBRE_SERVICIO {stop | start | status | ...}
```

Para que un servicio se inicie o no automáticamente en el arranque del sistema operativo, en el modo System V haríamos:

```
$ sudo update-rc.d NOMBRE_SERVICIO {enable | disable}   # activar o desactivar
```

Para los servicios basados en Upstart se puede editar el archivo `/etc/init/NOMBRE_SERVICIO.conf` y comentar la línea que empieza con start on. Para activar nuevamente el servicio será necesario eliminar el comentario al comienzo de la línea start on.

Otra forma es crear un archivo de texto con el nombre del servicio y extensión *override* y añadirle el texto "manual". Esto indica que el servicio ya solo arrancará de forma manual.

En el modo systemd se haría mediante:

```
$ sudo systemctl {enable | disable} NOMBRE_SERVICIO   # activar o desactivar en el arranque
```