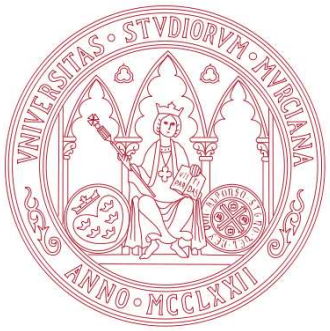

Universidad de Murcia

Facultad de Informática

TÍTULO DE GRADO EN
INGENIERÍA INFORMÁTICA

Fundamentos de Computadores

Tema 6: Introducción a las redes de ordenadores

Boletines de prácticas

CURSO 2020 / 21

Departamento de Ingeniería y Tecnología de Computadores

Área de Arquitectura y Tecnología de Computadores

Índice general

Boletines de prácticas

B6.1. Boletín 1: Configuración básica de red y servicios de comunicaciones

B6.1.1. Objetivos

Esta sesión de prácticas girará en torno a dos servicios fundamentales de la red Internet como son el DNS y el Web. El enfoque seguido para conocer el funcionamiento de estos dos servicios será mediante el uso de algunas herramientas fundamentales del sistema operativo Linux.

En lo que a servicios de red se refiere, un primer objetivo fundamental es conocer la naturaleza y el funcionamiento del sistema de DNS para la traducción de nombres y direcciones IP. De esa manera conoceremos cuál es el proceso que subyace a la mayoría de las conexiones establecidas en Internet en base a nombres.

En segundo lugar, se estudiarán los elementos constituyentes del Web para entender las bases de su funcionamiento y comprender mejor cuál es la función de un protocolo de comunicación. Mediante el análisis de HTTP, el alumno podrá comenzar a entender cómo se desarrollan los distintos servicios de Internet y cuál es la esencia del paradigma cliente-servidor.

Para la realización de la práctica, se asume que el alumno posee los conocimientos mínimos del manejo del sistema operativo Linux, así como de los navegadores Web. Por tanto no es necesario proporcionar otro tipo de detalles que no estén exclusivamente relacionados con los objetivos de la sesión.

B6.1.2. Plan de trabajo

El plan de trabajo de esta sesión será el siguiente:

1. Lectura del boletín por parte del alumno.
2. Presentación de los programas mediante ejemplos.
3. Realización de los ejercicios propuestos en el boletín.

B6.1.3. Consulta IP en equipos basados en Linux

En primer lugar estudiaremos cómo llevar a cabo la consulta de la dirección IP asignada de interfaces de red en el sistema operativo Linux. Aunque lo aquí descrito se refiere principalmente a la distribución Ubuntu, que es la que se encuentra instalada en el laboratorio, los conceptos son fácilmente trasladables a otras distribuciones.

En el caso de Linux, está la orden `ifconfig` que nos permite conocer la dirección IP de un equipo. Así mismo, veremos la orden `ping` que permite analizar el funcionamiento correcto o no de la red.

En la mayoría de los casos las interfaces de red se crean automáticamente por los controladores de dispositivo mientras se inicia y localiza el hardware. El nombre depende de la interfaz a la que representa y de la tecnología de red subyacente a dicha interfaz. Por ejemplo, el controlador Ethernet crea interfaces eth[0..n] secuencialmente según va encontrado tarjetas Ethernet. La primera tarjeta que encuentra es eth0, la segunda eth1, etc. Las interfaces inalámbricas suelen denotarse de modo similar, usando la notación wlan[0..n]. Sin embargo, hay que tener en cuenta que el nombre asignado depende muchas veces del sistema Linux usado.

La orden `ifconfig` (*interface configure*) es el programa usado más comúnmente para asignar direcciones y otros parámetros a las interfaces de red en Linux (y también UNIX). Esta orden permite configurar numerosos parámetros de las interfaces de red. Pero si ejecutamos `ifconfig` sin parámetros ni argumentos muestra la configuración vigente de las interfaces de red activas, en concreto dirección IP asignada, e incluso tráfico de red producido. Nosotros en este curso, la usaremos sin parámetros. A continuación se muestra un ejemplo de ejecución de la orden en uno de los laboratorios de prácticas:

```
alumno@lab13:~$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:c4:db:13:a7  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 155.54.223.216  netmask 255.255.255.224  broadcast 155.54.223.223
        inet6 fe80::a5a8:925b:a97a:1b93  prefixlen 64  scopeid 0x20<link>
        ether 54:a0:50:7a:f8:8c  txqueuelen 1000  (Ethernet)
        RX packets 279  bytes 111490 (111.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 263  bytes 37573 (37.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Bucle local)
        RX packets 244  bytes 19354 (19.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 244  bytes 19354 (19.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 52:54:00:c8:1c:d2  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Esta orden nos muestra que el ordenador tiene activas cuatro interfaces de red: `docker0`, `eth0`, `lo` y `virbr0`. Las interfaces `docker0` y `virbr0` se usan para temas de virtualización y quedan fuera del ámbito de esta asignatura. La interfaz `eth0` es la que configura la red de ese equipo y da salida a Internet. En concreto nos dice que la dirección IP de ese equipo es la 155.54.223.216, que su máscara de red es 255.255.255.224 y la dirección de broadcast de la red es 155.54.223.223. Con estos datos podemos saber que el equipo pertenece a la subred 155.54.223.192/27. Otro dato interesante es la dirección MAC (Media Access Control) de esa interfaz, también conocida como dirección hardware, que es la 54:a0:50:7a:f8:8c. Los demás datos quedan fuera del ámbito de esta asignatura. Finalmente, la interfaz de red `lo` es una interfaz de red virtual y se conoce como dispositivo de red loopback. Esta dirección especial, normalmente 127.0.0.1, es usada por los equipos para enviar tráfico a ellos mismos.

Por otro lado, la orden `route` nos permite visualizar/manipular las tablas de enrutamiento de nuestro sistema. En este curso, nos centraremos sólo en visualizar esta información.

```
alumno@lab13:~$ route -n
Tabla de rutas IP del núcleo
Destino         Pasarela        Genmask         Indic Métric Ref    Uso Interfaz
0.0.0.0         155.54.223.221  0.0.0.0         UG    100    0        0 eth0
```

```
155.54.223.192  0.0.0.0         255.255.255.224 U     100    0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 virbr0
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 virbr0
```

Este resultado nos indica que la subred 155.54.223.192 con máscara de red 255.255.255.224 (por tanto la referente a las IPs que abarcan desde la 155.54.223.192 a la 155.54.223.223) se conectan entre sí sin necesidad de router (significado de *) y que para acceder a otro host distinto de ese rango de IPs se accede por el router con IP 155.54.223.221.

También hay tres entradas para los interfaces que se usan para la virtualización, y que tal como hemos dicho quedan fuera de la explicación de esta asignatura.

La orden `ping`

La orden `ping` es una utilidad presente en la mayoría de los sistemas operativos actuales. Se utiliza principalmente para analizar el funcionamiento de la red y para verificar la configuración de los parámetros de red en un host o un servidor. El propósito de dicha orden es sencillo, pues consiste básicamente en el envío de un paquete de datos al host indicado y en la espera de la correspondiente respuesta a dicho paquete por parte del host. Por tanto, puede utilizarse tanto para averiguar si un host concreto se encuentra activo y a la vez también proporciona información acerca del tiempo total transcurrido entre el envío de la solicitud y la recepción de la respuesta. Un ejemplo de la ejecución de dicha orden es el siguiente:

```
[user@host ~]$ ping 155.54.1.200
PING 155.54.1.200 (155.54.1.200) 56(84) bytes of data.
64 bytes from 155.54.1.200: icmp_seq=1 ttl=255 time=0.371 ms
64 bytes from 155.54.1.200: icmp_seq=2 ttl=255 time=0.367 ms
64 bytes from 155.54.1.200: icmp_seq=3 ttl=255 time=0.395 ms
--- 155.54.1.200 ping statistics ---
3 packets transmitted, 3 received, no packet loss, time 2448ms
rtt min/avg/max/mdev = 0.367/0.377/0.395/0.025 ms
```

En este caso vemos como se han enviado tres paquetes de 64 bytes al host 155.54.1.200 que han generado sus correspondientes respuestas. Tanto el número de paquetes a enviar, como el tamaño u otros parámetros de funcionamiento pueden ajustarse mediante opciones específicas de la orden ping. Para consultar dichas opciones basta con introducir `ping -h`, aunque en este boletín no emplearemos ninguna de ellas.

B6.1.4. Servicios de comunicaciones 1: DNS (Domain Name System)

Al igual que HTTP (Web) o SMTP (correo electrónico), el protocolo DNS es un protocolo de nivel de aplicación ya que se ejecuta entre sistemas finales siguiendo el paradigma cliente-servidor y utiliza un protocolo de transporte para transferir los mensajes DNS entre los sistemas finales. Sin embargo, el papel del DNS es muy distinto del Web o el correo electrónico. Al contrario que dichas aplicaciones, el DNS no es una aplicación con la cual interactúa directamente el usuario. En su lugar, el DNS proporciona una función clave en el funcionamiento de Internet: traducir nombres en direcciones IP.

Al igual que los seres humanos, los hosts de Internet pueden ser identificados de muchas formas. Cada identificador de host se conoce como *nombre de host* o *hostname*. Algunos de ellos son muy conocidos, como `www.google.es` o `www.youtube.com`. Sin embargo, dichos nombres proporcionan poca información acerca de la localización exacta de dicho host. Además, dado que estos nombres pueden estar formados por combinaciones de caracteres alfanuméricos de longitud variable, serían muy difíciles de procesar por parte de los routers. Ya sabemos que para el direccionamiento se hace uso de direcciones IP, no de nombres, pero para los seres humanos resulta mucho más fácil recordar los nombres en lugar de las direcciones IP.

Por tanto, se necesita un servicio de traducción que convierta los nombres de host en direcciones IP y esta es la tarea principal del DNS. Técnicamente, el DNS es una base de datos distribuida implementada mediante una jerarquía de servidores DNS y mediante un protocolo de consulta para acceder a los datos almacenados en dicha base de datos. Este protocolo utiliza UDP para transmitir sus mensajes y tiene asignado el puerto 53.

El DNS es un servicio del que hacen uso normalmente las aplicaciones para traducir los nombres en direcciones IP. Vamos por ejemplo a considerar qué sucede cuando le proporcionamos a un navegador (que al fin y al cabo es un cliente HTTP) la dirección `www.um.es`. Para que el host donde se ejecuta el navegador pueda enviarle un mensaje HTTP al servidor `www.um.es` es necesario primero obtener la dirección IP de dicho servidor. Esto se realiza de la siguiente manera:

1. El navegador proporciona el nombre de host `www.um.es` al DNS cliente que se ejecuta en nuestro host.
2. El cliente DNS envía una consulta sobre dicho nombre al servidor DNS que tenga configurado el host como servidor DNS primario[1].
3. El cliente DNS obtendrá una respuesta del servidor DNS que incluya la dirección IP asociada a dicho nombre. Si no obtiene respuesta probará a realizar la consulta al servidor DNS secundario.
4. Una vez que finalmente el navegador obtiene la dirección IP de parte del cliente DNS, puede establecer una conexión TCP con el servidor HTTP localizado en el puerto 80 de dicha dirección IP.

Si bien el proceso puede parecer sencillo, es conveniente aclarar que no siempre el servidor DNS primario o secundario sabrán traducir el nombre proporcionado, lo que implicaría reenviar la consulta a su vez a otros servidores DNS que tengan configurados. De ahí que el mantenimiento de la jerarquía de servidores DNS de Internet sea un aspecto crítico de su funcionamiento.

Además de para el propósito arriba indicado, el servicio DNS puede utilizarse también para realizar traducciones inversas (obtener el nombre en función de la IP), para la gestión de alias (distintos nombres para una misma máquina) u otros aspectos relacionados con el correo electrónico o el balanceo de carga.

La herramienta host

La herramienta de Linux denominada `host` es una utilidad que permite realizar consultas a servidores DNS. Normalmente se utiliza para convertir un nombre en una dirección IP y viceversa. La forma más habitual de uso es como la del siguiente ejemplo:

```
user@localhost$ host www.um.es
www.um.es is an alias for wwwclu.um.es.
wwwclu.um.es has address 155.54.212.103
```

Esto hace que se utilice el sistema DNS para acabar obteniendo la dirección IP asociada a dicho nombre (en este ejemplo 155.54.212.103). Además, si dicho nombre es un alias de otros nombres distintos, como en este caso, la herramienta `host` también nos lo indica. Sin embargo, también podemos introducir la siguiente orden para obtener un nombre (y posibles alias) asociado a una dirección IP:

```
user@localhost$ host 155.54.212.103
103.212.54.155.in-addr.arpa domain name pointer um.es.
103.212.54.155.in-addr.arpa domain name pointer editum.es.
103.212.54.155.in-addr.arpa domain name pointer wwwclu.um.es.
```

[1] Tanto la IP del servidor DNS primario como de los posibles DNS secundarios se pueden consultar usando el comando `nmcli`.

B6.1.5. Servicios de comunicaciones 2: WWW (World Wide Web)

La Web es la aplicación de Internet que mayor aceptación y más relevancia tiene hoy en día. Se trata de un servicio que funciona bajo demanda, dado que los usuarios reciben lo que quieren cuando quieren. Algunas claves de su éxito son la facilidad para publicar nuevos contenidos por parte de prácticamente cualquier usuario y la posibilidad de enlazar los distintos contenidos mediante los hipervínculos o los buscadores de información. En el corazón de este servicio se encuentra el sistema de direccionamiento mediante URLs (Uniform Resource Locators), el protocolo HTTP (HyperText Transfer Protocol) y el lenguaje HTML (HyperText Markup Language) para la especificación de contenidos Web.

URLs

Las URLs son secuencias de caracteres utilizadas para identificar contenidos, como texto o multimedia, en Internet. Estas secuencias asignan una dirección única a cada uno de los contenidos existentes en Internet, de tal forma que existe una URL única para cada página Web, para cada elemento de imagen dentro de la misma, o incluso para otros recursos no sólo accesibles mediante Web, como ficheros distribuidos mediante FTP.

El formato general de un URL es `protocolo://máquina/directorio/archivo`, aunque también podrían contemplarse otros datos adicionales. Por ejemplo, `http://www.um.es/estudios/`, sería un ejemplo de URL que haría referencia a los contenidos accesibles mediante HTTP ubicados en el directorio `estudios` de la máquina `www.um.es`. Como veremos a continuación, el sistema de direccionamiento basado en URLs es un aspecto central del protocolo HTTP.

HTTP

El protocolo HTTP es el núcleo del Web. Está definido en el documento público RFC 2661 y sigue el paradigma cliente-servidor. HTTP define la estructura de los mensajes intercambiados entre estas dos entidades. Básicamente HTTP define cómo los clientes Web, los navegadores, deben solicitar páginas Web a los servidores y cómo estos deben ir proporcionando la información solicitada por los navegadores de acuerdo a los parámetros especificados en la solicitud.

HTTP utiliza TCP como protocolo de transporte, así que lo primero que debe realizar el cliente es iniciar una conexión TCP con el servidor. Una vez establecida se procede al intercambio de información entre las dos entidades. Es interesante conocer que el protocolo HTTP es un protocolo sin estado, o dicho con otras palabras, el servidor no guarda ningún tipo de información sobre las consultas ya procesadas, lo que le impide por tanto saber, por ejemplo, si la página que le está solicitando el cliente ya ha sido enviada previamente a esa misma entidad. Cada conexión es completamente nueva.

El protocolo HTTP define principalmente dos mensajes distintos: mensajes de solicitud (HTTP Request) y mensajes de respuesta (HTTP Response). Vamos a analizar un poco el contenido de ambos mensajes puesto que resulta interesante conocer el funcionamiento básico de HTTP. Lo que aparece a continuación es un ejemplo de mensaje HTTP Request:

```
GET /~pedroe/documentos/prueba.html HTTP/1.1
Host: ditec.um.es
Connection: close
```

Mediante este mensaje es posible solicitar (`GET`) la página `prueba.html`, que se encuentra en la ruta especificada `/~pedroe/documentos/` del servidor `ditec.um.es`, indicando que una vez que dicha página ha sido servida la conexión debe cerrarse (`Connection: close`). Se indica además que se está utilizando la versión 1.1 del protocolo HTTP. La línea en blanco del final del mensaje es importante, puesto que es la que indica que el mensaje ha terminado y no hay más parámetros de la solicitud.

La respuesta a dicha solicitud se codifica mediante un mensaje HTTP response. Dicho mensaje tendrá una cabecera donde se especificará información acerca de cuál ha sido el resultado de la consulta y un cuerpo en el que irá incluida la página solicitada. A continuación se muestra un ejemplo de lo que podría obtenerse como respuesta:

```
HTTP/1.1 200 OK
Date: Fri, 07 Jan 2011 09:29:47 GMT
Server: Apache/2.2.11 (Fedora)
Last-Modified: Sun, 26 Sep 2010 16:42:34 GMT
ETag: "7ec0a8-141-4912c4c727280"
Accept-Ranges: bytes
Content-Length: 321
Connection: close
Content-Type: text/html; charset=es_ES.UTF-8

<HTML>
 <HEAD>
  <TITLE>Ejemplo</TITLE>
  <META NAME="AUTHOR" CONTENT="Fundamentos de Computadores">
 </HEAD>
 <BODY>
  <P>Esta palabra est&aacute; en <B>negrita</B>.</P>
  <P>Tambi&eacute;n la puedo poner en <I>cursiva</I>.</P>
  <BR>
  <P>Un <A HREF="http://www.um.es/">enlace</A> al Web de la UMU.</P>
 </BODY>
</HTML>
```

Como se puede apreciar, la respuesta está expresada también mediante HTTP 1.1. La primera línea de la cabecera indica también que la solicitud se ha procesado correctamente (código 200 OK). Se indica el instante en el que ha sido servida la página, el software de servidor Web que ha construido el mensaje, la fecha de la última modificación de la página solicitada, el tipo de información que contiene la página solicitada (texto HTML), la longitud de dicha página (321 bytes) y la codificación de caracteres utilizada (UTF-8). La información que aparece a partir de la línea en blanco es la propia página HTML de resultado.

Hay varias posibilidades a la hora de tramitar una consulta por parte de un servidor Web. El resultado de dicha tramitación se indica en el campo de estado, el cual puede tomar habitualmente algunos los siguientes valores[2]:

1. 200 OK. La solicitud se procesó correctamente y la información se devuelve en la respuesta.
2. 301 Moved permanently. El contenido solicitado ha cambiado de localización. En dicho caso la nueva URL se incluye en un campo de la cabecera del mensaje correspondiente. La mayoría de los navegadores solicitarán automáticamente el contenido de la nueva URL.
3. 400 Bad Request. Este código genérico se utiliza para informar de que la solicitud recibida no estaba bien formada y por tanto no ha podido ser entendida.
4. 404 Not Found. El contenido solicitado no se encuentra en el servidor.

Cualquiera puede entablar una conversación HTTP con un servidor, no sólo los navegadores Web. Tenemos una forma muy sencilla de hacerlo mediante el uso del programa `telnet`. Dado que el primer paso para comunicarnos mediante HTTP es establecer una conexión TCP con el puerto 80 del servidor, lo primero que deberemos realizar es lo siguiente:

[2]Naturalmente, la lista no es exclusiva.

```
$ telnet www.ditec.um.es 80
```

Esto hará que la conexión TCP quede establecida. A partir de ese instante será sólo tenemos que teclear el contenido del mensaje HTTP Request que nos interese y una vez introducido esperar la respuesta por parte del servidor.

B6.1.6. La orden netstat

La orden `netstat` permite identificar las conexiones TCP que están activas en la máquina en la que se ejecuta el comando. Normalmente se utiliza con las opciones `-tupan`, donde cada argumento significa:

- `-t` Visualiza las conexiones TCP.
- `-u` Visualiza las conexiones UDP.
- `-a` Visualiza todas las conexiones y puertos TCP y UDP, incluyendo las que están 'en escucha' (listening).
- `-n` Se muestran los puertos con su identificación en forma numérica y no de texto.
- `-p` Visualiza el identificador del proceso (PID) y el nombre del programa que abrió la conexión.

El siguiente ejemplo muestra algunas líneas de una ejecución de `netstat -tupan` en uno de los laboratorios:

```
alumno@lab13:~$ netstat -tupan
...
Proto  Recib Enviad Dirección local         Dirección remota       Estado       PID/Program name
....
tcp        0      0 155.54.223.216:34020    93.184.220.29:80        ESTABLECIDO 3424/firefox
tcp        0      0 155.54.223.216:58640    151.139.237.11:443      ESTABLECIDO 3424/firefox
tcp        0      0 155.54.223.216:39288    216.58.211.38:443       ESTABLECIDO 3424/firefox
tcp        0      0 155.54.223.216:42802    172.217.168.170:443     ESTABLECIDO 3356/chrome --type=
tcp        0      0 155.54.223.216:42498    172.217.168.170:443     ESTABLECIDO 3424/firefox
tcp        0      0 155.54.223.216:34324    93.184.220.29:80        ESTABLECIDO 3356/chrome --type=
tcp        0      0 155.54.223.216:56076    99.86.163.60:443        ESTABLECIDO 3424/firefox
tcp        0      0 155.54.223.216:55162    172.217.17.2:443        ESTABLECIDO 3424/firefox
.....
udp        0      0 192.168.122.1:53        0.0.0.0:*                           -
udp        0      0 0.0.0.0:161             0.0.0.0:*                           -
udp        0      0 0.0.0.0:32993           0.0.0.0:*                           -
udp        0      0 224.0.0.251:5353        0.0.0.0:*                           3356/chrome --type=
udp        0      0 224.0.0.251:5353        0.0.0.0:*                           3317/chrome
```

En concreto se muestra el protocolo usado, las direcciones IP del equipo local y del equipo remoto, el estado de la conexión y el nombre del programa que estableció esa conexión.

B6.1.7. Ejercicios a realizar durante la sesión

Por grupos de hasta dos personas, y haciendo uso cada grupo de un PC con sistema operativo Linux, llevar a cabo los siguientes ejercicios:

1. Comprobar la configuración IP del PC utilizado. Averiguar la dirección IP asignada y la máscara de red (usando el comando `ifconfig` sin parámetros). Comprobar también el router por defecto (comando `route`) y los servidores DNS asignados (esto último con el comando `nmcli`, que da la información resumida más completa).
2. Verificar la conexión de red, para ello utilizar el programa `ping`. Comprobar que se consigue llegar tanto a un equipo de la propia red como al router y a los servidores `www.um.es` y `www.google.es`.
3. Haciendo uso de la herramienta `host`:
   a) Comprobar la dirección IP asociada a `www.carm.es`.

b) Comprobar si `www.wikipedia.es` es un alias de otro nombre.

c) Averiguar cuántas direcciones IP tiene asociadas `www.hotmail.com`.

d) Averiguar cuál es el nombre asociado a la dirección IP 155.54.212.102.

4. Utilizar el navegador `firefox` para obtener la página principal de `slashdot.org`.

5. Volver a realizar el ejercicio anterior, pero introduciendo ahora la dirección IP en lugar de su nombre.

6. Haciendo uso del programa `telnet`, realizar los siguientes apartados:

   a) Obtener la página principal del servidor `www.ditec.um.es`.

   b) Realizar una consulta al servidor `www.ditec.um.es` de una página Web que no exista, como por ejemplo `~pedroe/documentos/fallo.html` .

   c) Obtener la página `~pedroe/documentos/prueba-con-imagen.html` del servidor `www.ditec.um.es` y a continuación solicitar la imagen referenciada en dicha página.

7. Vamos a ahora a actuar como un servidor Web. Para ello, vamos a realizar lo siguiente:

   a) Abrir una nueva ventana con un terminal.

   b) Utilizar el programa `nc` para escuchar conexiones en el puerto 30001. Para ello basta con ejecutar `nc -l 30001` (o, dependiendo de la versión, podría tener que ejecutarse el comando en la forma `nc -l -p 30001`).

   c) Utilizar el navegador para solicitar el contenido de la URL `http://<IP>:30001`, donde <IP>hace referencia a la IP del equipo que se está utilizando.

   d) Una vez que se vea en la ventana del terminal el mensaje HTTP Request enviado por el navegador, contestar mediante un mensaje HTTP response cuyo tipo de contenido sea `text/plain` y que contenga la palabra *Hola*. Para ello, escribir en la ventana un mensaje similar al siguiente:

   ```
   HTTP/1.1 200 OK
   Content-Type: text/plain
   Content-Length: 4

   Hola
   ```

   e) Comprobar en el navegador que aparece dicha palabra (podéis, por supuesto, cambiar el mensaje para que sea más largo y tenga varias líneas, o incluso contestar con un pequeño texto en HTML, cambiando adecuadamente los campos *Content-Type* y *Content-Length* de la cabecera).

8. Haciendo uso de `netstat -tupan`, probar a repetir algunos de los ejercicios anteriores, intentando investigar el comportamiento de puertos y conexiones.

9. Utiliza el comando `route` y `route -n` para visualizar cuál es la tabla de enrutamiento establecida. Interpreta el resultado obtenido.