Hackeando la Matriz

Guía de compilación del núcleo Linux

Por Quique

Martes 1 abril 2003.

Una de las grandes ventajas de usar sistemas operativos libres tipo Gnu/Linux o las distintas versiones de los Unix libres BSD, es la posibilidad de modificar el "corazón" de nuestro sistema a nuestro gusto.

Ello nos permite tener un "núcleo" de nuestro sistema a la medida de nuestra computadora y nuestras necesidades. Desde hacer que trabaje con dispositivos obsoletos o avanzados , hasta añadir un cortafuegos, pasando por incorporarle opciones de seguridad. Las posibilidades son amplias frente a otros sistemas ("de cuyo nombre no quiero acordarme") que no nos dejan ni tan siquiera ver el "motor" de nuestra máquina.

Antes de empezar

Recordemos que:

- la mayoría de los pasos necesitan ser realizados como usuario root.
- debemos tener instalados algunos paquetes de desarrollo: gcc, libc6−dev, make

Instrucciones para compilar, paso a paso

1. Antes de nada comprobaremos si tenemos la última versión del núcleo (si no es así aprovecharemos la ocasión para actualizarlo). Para saber que versión del núcleo tenemos instalada usamos la orden **uname −sr** (ó **uname −a**, que proporciona información adicional). Supongamos que observamos que nuestro sistema usa la versión 2.2.17.

   Podemos obtener el código fuente de la última versión del núcleo de Internet (http://www.kernel.org). Dado el tamaño del núcleo, las revistas sobre GNU/Linux lo suelen incluir en el CD que habitualmente las acompaña. En esta ocasión instalaremos la versión 2.2.19, que copiamos al directorio /usr/src (**cp /cdrom/linux−2.2.19.tar.bz2 /usr/src/**).

2. Vamos al directorio /usr/src (**cd /usr/src**. Si aquí hay ya un directorio llamado linux, debemos renombrarlo (a linux−2.2.17, por ejemplo: **mv linux linux−2.2.17**). No es conveniente borrar nada del núcleo anterior hasta que comprobemos que el nuevo funciona correctamente. Descomprimimos el

tarball con la orden **tar −jvxf linux−2.2.19.tar.bz2** (o **tar −zxvf linux−2.2.19.tar.gz**, según el formato).

3. Aparece un directorio llamado linux. Una sana costumbre es renombrarlo a linux−2.2.19 (**mv linux linux−2.2.19**), y crear un enlace simbólico llamado linux a dicho directorio con la orden **ln −s linux−2.2.19 linux**. Una vez descomprimido ya no vamos a necesitar el código fuente comprimido, así que podemos borrarlo (**rm linux−2.2.19.tar.bz2**).

4. Una vez instalado el código fuente, entramos en el directorio linux (**cd linux**. Si ya habíamos compilado anteriormente un núcleo con estas fuentes, ejecutamos la orden **make mrproper**, que elimina todos los binarios y ficheros de configuración que puedan quedar.Al hacerlo volvemos a tener las fuentes del kernel tal y cual las bajamos, así que al volver a compilar hay que hacer de nuevo todos los pasos, incluyendo **make dep**.

   Si se trata de un Power Macintosh (PPC) y vamos a compilar el núcleo estándar, ejecutamos la orden **make pmac_config**, que selecciona el fichero de configuración especial para Mac.

   Ahora podemos pasar a configurar el núcleo con la orden **make xconfig**, que nos proporciona un interfaz gráfico basado en las bibliotecas Tcl/Tk. Otras alternativas son **make menuconfig** (la configuración se hará mediante un sistema de menús ncurses) y **make config** (pregunta sucesivamente todas las posibles opciones, muy engorroso).

5. Configuramos las opciones del kernel que necesitemos. Las posibilidades son demasiadas como para explicarlas en este pequeño tutorial, pero cada opción viene acompañada de una breve explicación. En general, pondremos en el núcleo las funcionalidades que necesitemos habitualmente, y cargaremos como módulos las que usemos ocasionalmente.

   Cuando hayamos acabado, pulsamos sobre **"Save and Exit"** para guardar nuestra configuración y salir del programa. Las opciones seleccionadas se guardan en un fichero llamado .config, que podemos copiar en otro sitio si queremos recordar en el futuro esta configuración.

6. Ha llegado por fin el momento de compilar. Lanzamos la orden **make dep && make clean && make bzImage**, y nos vamos a tomar un café (según la potencia de nuestra máquina le puede tomar un buen rato).

   Este paso de **make dep** sólo hay que hacerlo una vez, al descomprimir las fuentes del kernel y después del primer 'make menuconfig'. A partir de ahora, si queremos hacer pruebas, añadir y quitar opciones, el **make dep** nos lo podemos saltar.

   Además de **make bzImage** otras opciones son **make zImage**, **make zdisk** y **make zlilo**. Si se trata de un PowerMac creo que debe ser **make vmlinux**.

7. Una vez compilado el núcleo, le toca el turno a los módulos. Para ello ejecutamos la orden **make modules && make modules_install**.

   En caso de que ya hubiéramos compilado esta versión del núcleo anteriormente, deberíamos previamente borrar (o mejor renombrar) el directorio /lib/modules/2.2.19 (**mv /lib/modules/2.2.19**

/lib/modules/2.2.19.bak).

8. Debemos copiar el fichero System.map y el núcleo obtenido (que se encuentra en /usr/src/linux/arch/i386/boot/bzImage, si nuestra arquitectura es i386) al directorio /boot , con los nombres System.map−2.2.19 y vmlinuz−2.2.19 (**cd /boot; cp /usr/src/linux/System.map System.map−2.2.19; cp /usr/src/linux/arch/i386/boot/bzImage vmlinux−2.2.19**). Es conveniente además que creemos en ese directorio unos enlaces simbólicos System.map y vmlinuz apuntando a ellos (**rm System.map; ln −s System.map−2.2.19 System.map; rm vmlinuz; ln −s vmlinuz−2.2.19 vmlinuz**).

9. Por último actualizaremos si es necesario el fichero de configuración de nuestro gestor de arranque (/etc/lilo.conf si es LILO y /boot/grub/menu.lst si es GRUB). Es conveniente añadir una entrada para nuestro kernel anterior, como medida de precaución por si hubiera algún problema con el nuevo núcleo.

   Si estamos en un PowerMac podemos usar BootX para arrancar GNU/Linux desde MacOS. Para ello tendremos que copiar el fichero vmlinux obtenido a la Carpeta del Sistema en la partición de MacOS.

10. Ya podemos reiniciar nuestra máquina (**shutdown −r now**), que deberá arrancar con el nuevo núcleo, como podemos comprobar con la orden **uname −sr**.

Truco

Debian tiene un paquete llamado kernel−package que facilita la compilación e instalación del núcleo. Una vez configurado, únicamente habrá que lanzar la orden **make−kpkg clean && make−kpkg −−revision fecha kernel_image && make−kpkg −−revision fecha modules_image**. Obtendremos unos paquetes .deb que nos servirán para instalar el núcleo que hemos configurado en cualquier máquina Debian (de la misma arquitectura), y que incluso modificarán la configuración de LILO automáticamente.

Más información

Si necesitamos instrucciones con más detalle, explicaciones de lasdiferentes opciones del núcleo, etc, podemos recurrir al **The Linux Kernel HOWTO**.

Si éste resulta demasiado extenso, lo tenemos resumido en el kernel−mini−howto. Otro texto más informal pero bastante completo es el **Kernel−Aaargh!** de runlevel0.

Si necesitamos instrucciones para configurar LILO o GRUB, consultaremos el **LILO mini−HOWTO** o el **GRUB Manual**, respectivamente.

Sobre este documento

El texto de la licencia se encuentra en **http://www.fsf.org/copyleft/fdl.html.**