Bisoños Usuarios de GNU/Linux de Mallorca y Alrededores | Bergantells Usuaris de GNU/Linux de Mallorca i Afegitons

Parcheando a la Debian (20181 lectures)

Per **Eduard Llull**, ***Daneel*** ()

Creado el 21/08/2002 16:15 modificado el 21/08/2002 16:15

*Una de las cosas que más identifican a **Linux** es la posibilidad de compilar el núcleo (kernel) del Sistema Operativo. Desde hace ya algún tiempo, la distribución **Debian** cuenta con unas herramientas que permiten compilarlo integrándolo en su sistema de gestión de paquetes. Veamos pues, cómo podemos parchear nuestro kernel para adaptarlo a nuestras necesidades, utilizando estas herramientas.*

1. Obteniendo las fuentes del kernel.

Lo primero que debemos hacer es obtener las fuentes del kernel. Esto se puede hacer con un simple:

```
host:~# apt-get install kernel-source-2.4.X
```

Pero para que esto funcione, debe existir un paquete con las fuentes. No suele haber problemas, pero hay un pequeño retardo con las últimas versiones. Desde que la ultima versión estable del kernel hasta que esta está disponible en forma de paquete pueden pasar unos días. Para saber que versiones están disponibles en los repositorios de paquetes podemos usar:

```
host:~# apt-cache search kernel-source
```

Una vez descargado el kernel, el `apt-get` nos deja un fichero comprimido con las fuentes en `/usr/src`, que debemos descomprimir. Además, es recomendable crear un enlace simbólico al directorio con las fuentes que se llame `linux`:

```
host:~# cd /usr/src
host:/usr/src# tar -jxvf kernel-source-2.4.X.tar.bz2
host:/usr/src# rm linux
host:/usr/src# ln -s kernel-source-2.4.X linux
```

2. Obteniendo los parches.

Igual que antes, para comprobar que parches podemos descargar como paquetes usaremos:

```
host:~# apt-cache search kernel-patch
```

Y si vemos uno que nos llama la atención y queremos averiguar para que sirve, podemos hacerlo con:

```
host:~# apt-cache show <nombre_del_paquete>
```

Por ejemplo:

```
host:~# apt-cache show kernel-patch-lowlatency-2.4
Package: kernel-patch-lowlatency-2.4
Priority: extra
Section: devel
Installed-Size: 124
Maintainer: Simon Law
Architecture: all
Version: 20020818-1
Depends: bash (>= 2.0), patch, kernel-patch-scripts (>> 0.99.12)
Suggests: kernel-package, kernel-source-2.4,
kernel-patch-preempt-2.4
Filename: pool/main/k/kernel-patch-lowlatency-2.4/
kernel-patch-lowlatency-2.4_20020818-1_all.deb
Size: 49574
MD5sum: 61f8fe047377ae1d8dba4df89da260e8
Description: Reduces the latency of the Linux kernel
 This patch to the Linux kernel reduces its scheduling latency. This
 makes the kernel more responsive, and potentially increases its
bandwidth;
 since threads waste less time waiting for execution.
 .
 It can be applied to the following kernel sources:
 2.4.17, 2.4.18, 2.4.19
```

Para instalarlo usaremos el `'apt-get install'` que nos dejará los parches y unos scripts para aplicarlos en el directorio `/usr/src/kernel-patches`.

---

3. Parcheando el núcleo.

Ahora que ya tenemos las fuentes del núcleo y los parches en nuestro sistema, debemos aplicar los cambios. Como casi todo en Linux se puede hacer de diversas formas.

3.1 Automáticamente.

Podemos hacer que la misma aplicación que nos compila el kernel y nos genera el paquete, nos aplique también los parches. Entonces el proceso de compilar el núcleo sería:

```
host:~# cd /usr/src/linux
host:/usr/src/linux# make menuconfig (o config, o xconfig)
host:/usr/src/linux# make-kpkg clean
host:/usr/src/linux# make-kpkg --added-patches <parches> --revision
1.0-con_parches kernel_image
```

Para que el `make-kpkg` aplique los parches que le indicamos con el parámetro `--added-patches`, la variable de entorno `PATCH_THE_KERNEL` debe tener el valor AUTO. Esto lo podemos hacer a mano antes de ejecutar el `make-kpkg kernel-image`:

```
host:/usr/src/linux# PATCH_THE_KERNEL=AUTO; export PATCH_THE_KERNEL
host:/usr/src/linux# make-kpkg --added-patches <parches> --revision
1.0-con_parches kernel_image
```

De esta manera, siempre se aplicarán los parches que le indiquemos.

Pero, ¿cómo configuramos la configuración del kernel de las modificaciones que acabamos de introducir? Por suerte, `make-kpkg kernel-image` hace un `make oldconfig` antes de empezar a compilar. En este momento será cuando podamos configurar las opciones que se han añadido con los parches.

3.2 Manualmente.

Si no nos acaba de gustar el anterior método, podemos parchear el núcleo manualmente, usando los scripts que vienen con los paquetes de los parches para su correcta y sencilla aplicación:

```
host:~# cd /usr/src/linux
host:/usr/src/linux# ../kernel-patches/<arquitectura>/apply/<parche>
```

Si los parches no dependen de la arquitectura, estarán en el directorio `/usr/src/kernel-patches/all`.

Ahora ya podemos configurar el kernel y compilarlo como siempre. Como hemos aplicado los parches manualmente no hace falta definir la variable `PATCH_THE_KERNEL` ni tampoco utilizar el parámetro `--added-patches`.

Personalmente, prefiero esta forma de aplicar los parches. Me da mayor sensación de control sobre las modificaciones que sufre el núcleo. Pero por eso nos gusta tanto Linux, cómo dijo Larry Wall: TMTOWTDI (*There's More Than One Way To Do It*, hay más de una forma de hacerlo)

---

4. Ejemplo.

Muchas veces un ejemplo puede servir de mucha ayuda, así que voy a explicar como preparar un kernel con los parches lowlatency y preemptive con las herramientas de Debian (además, fue una de las peticiones que se pudieron escuchar durante la cena de la última kedada):

1. Nos descargamos el kernel 2.4.19 y los parches lowlatency y preemptive:

   ```
   host:~# apt-get install kernel-source-2.4.19
   kernel-patch-lowlatency-2.4 kernel-patch-preempt-2.4
   host:~# cd /usr/src
   host:/usr/src# tar -jxvf kernel-source-2.4.19
   host:/usr/src# rm linux
   host:/usr/src# ln -s kernel-source-2.4.19 linux
   ```

2. Ahora aplicaremos los parches con el método manual:

   ```
   host:/usr/src# cd linux
   host:/usr/src/linux# ../kernel-package/all/apply/lowlatency
   host:/usr/src/linux# ../kernel-package/all/apply/preempt
   ```

3. A partir de aquí, lo de siempre. Configuramos las opciones de configuración el kernel, incluidas las opciones de los parches que acabamos de aplicar, y compilamos:

   ```
   host:/usr/src/linux# make menuconfig
   host:/usr/src/linux# make-kpkg clean
   host:/usr/src/linux# make-kpkg --revision 1.0.desktop
   kernel_image
   ```

4. Finalmente instalamos el paquete con el kernel:

```
host:/usr/src/linux# dpkg -i
../kernel-image-2.4.19_1.0.desktop_i386.deb
```

5. Conclusiones.

Puede parecer que nos estamos complicando la vida para hacer una cosa que ya hacíamos con herramientas "tradicionales" wget, tar y patch, pero lo bueno de tener los parches integrados en el sistema de gestión de paquetes es que si alguna vez aparece una nueva versión del parche (que añade nuevas funcionalidades, o que soluciona bugs), al hacer el periódico `apt-get update && apt-get dist-upgrade`, el sistema se descargará la nueva versión, ahorrandonos ese trabajo a nosotros, sus administradores. Además, podemos estar seguros de que el parche que nos hemos descargado se aplicará sin problemas a nuestro kernel.

El único inconveniente está en el retraso que suele haber desde que aparece un nuevo kernel hasta que están disponibles los paquetes de los parches adaptados a esa versión. Pero bueno, con un poco de paciencia se arregla todo.

E-mail del autor: daneel _ARROBA_ bulma.net
Podrás encontrar este artículo e información adicional en: http://bulma.net/body.phtml?nIdNoticia=1457