Laboratorio Sesión 04: Instalación de paquetes y compilación del kernel

En esta sesión vais a compilar un nuevo kernel de Linux a partir de las fuentes y a instalar nuevos paquetes bajados de Internet. Como la compilación del kernel es larga, empezaremos compilando el kernel y a continuación, mientras este se compila, aprovecharemos que el sistema es multitarea para ir haciendo la práctica de instalación de paquetes. Si todo va bien, antes de acabar la práctica el kernel estará compilado y listo para usarse. Ánimo.

4.1. Un par de conceptos sobre paquetes:

Antes de empezar debéis conocer unos pocos conceptos simples sobre paquetes. Un paquete en Linux no es más que un conjunto de ficheros (que suelen formar un programa o aplicación completo) agrupados en un solo fichero. A diferencia de lo que suele pasar en otros sistemas operativos, esto no implica que los ficheros además estén comprimidos, aunque suele ser habitual. Podríamos distinguir entre dos tipos de paquetes:

- Paquetes de instalación: son paquetes con información específica para instalarse en algún tipo de sistema concreto, como por ejemplo `.deb` (Debian) o `.rpm` (Red Hat) Estos paquetes deben utilizarse con las herramientas de instalación de cada distribución (`apt-get` o `rpm`). En general son muy sencillos de instalar y no suelen dar ningún problema de dependencias ya que el propio sistema operativo se encarga de gestionarlas.
- Paquetes genéricos: el sistema de agrupación de ficheros más común en los sistemas *nix es `tar`. Este comando (podéis hacer un `info tar`) permite simplemente agrupar y desagrupar ficheros. El fichero paquete se distingue porque suele tener un nombre terminado en `.tar`. Si queremos obtener los ficheros que hay dentro de un fichero `.tar` deberemos ejecutar la orden: `tar xvf nombre.tar`. Este sistema es mucho más simple y genérico pero también puede dar lugar a muchos más problemas ya que es totalmente manual y no está controlado por el sistema.

Pregunta 4.1: *Cread 3 ficheros con un contenido cualquiera (Importante, si queréis ver que se comprimen poned varias lineas de texto, de otra forma los índices de compresión ocuparán más que los ficheros y al comprimirlos creceran en tamaño). Agrupadlos en un fichero llamado "Grupo.tar". ¿Qué ordenes habéis usado? ¿Que espacio ocupan los ficheros por separado? ¿Y el fichero* `.tar`*?*

Además una vez agrupados los ficheros originales, el fichero resultante puede comprimirse con el sistema de compresión que queramos. Los más típicos suelen ser los sistemas `gzip`, `bzip2` o el nuevo `xz`. Los ficheros comprimidos mediante este sistema suelen tener como extensión final `.gz`, `.bz2` o `.xz`. Para descomprimir y desempaquetar un fichero `.tar.gz` o `.tar.bz2`, deberéis ejecutar: `tar xvzf nombre.tar.gz` o `tar xvjf nombre.tar.bz2`.

Pregunta 4.2: *¿Qué orden desagrupa el fichero "Grupo.tar"? ¿Qué orden comprime "Grupo.tar" y lo transforma en "Grupo.tar.bz2"? ¿Y en "Grupo.tar.gz"? ¿Cúal de los dos ficheros comprimidos ocupa menos espacio?*

Pregunta 4.3: *¿Qué orden descomprime el fichero "Grupo.tar.bz2" y devuelve "Grupo.tar"? ¿Qué orden agrupa y comprime directamente los tres ficheros originales en "Grupo.tar.gz"?*

Pregunta 4.4: *¿Qué orden descomprime el formato "xz"? ¿Qué opción de "tar" trabaja con la extensión "xz"?*

A lo largo de esta práctica mucha de la información que os bajaréis está en ficheros `.tar`, evidentemente antes de trabajar con los ficheros que contienen deberéis descomprimirlos.

4.2. Compilación del kernel:

El kernel de Linux es un paquete como todos los demás, por lo tanto lo habitual en un sistema de escritorio (por ejemplo en vuestra casa) es actualizarlo con el sistema de actualización de paquetes normal (de hecho se suele hacer de modo automático) que acostumbréis a usar. Estos sistemas (como veremos luego) tienen la ventaja de requerir muy poco conocimiento (o ninguno) del sistema, pero a cambio solo trabajan con el kernel estándar, es decir, menos actualizado y/o adaptado a nuestro sistema concreto. Si nosotros mismos bajamos las fuentes del kernel, las configuramos para nuestro sistema y las compilamos, obtendremos un kernel que, por el hecho de ser mucho más específico, funciona mejor en nuestro sistema. Esto puede ser importante en servidores de alto rendimiento, en sistemas móviles específicos, en sistemas antiguos y por lo tanto escasos de hardware o, simplemente, porque debemos incorporar una funcionalidad muy especial.

Para empezar deberéis bajaros las fuentes del kernel de una dirección fiable. La mejor para empezar es `www.kernel.org`, que como su nombre indica es la página de una organización que se dedica a almacenar los kernels. Buscad por los directorios hasta encontrar las fuentes de la última versión de kernel disponible. Tened en cuenta que las versiones del kernel tienen una numeración principal compuesta de 3 cifras separadas por puntos: "x.y.z". La "x" es el número de versión, que actualmente se modifica cada relativamente poco tiempo (en realidad a partir de la versión 2.6 cambiaron la numeración a 3.0 para celebrar los 20 años de Linux y a partir de ahí la diferencia entre versión y subversión ya no ha sido tan clara). La "y" es la subversión y la "z" es la actualización. Cada "z" más avanzada significa que contienes al menos un bug menos que la anterior de forma que siempre es buena idea trabajar con la última "z" existente. Además en algunas distribuciones (como Fedora) existe una cuarta cifra que es la versión de compilación con correcciones menores. En este caso también es buena idea ir actualizando el kernel con las nuevas subsubsubversiones. Puede sonar a mucho trabajo, pero pensad que la versión nueva del kernel os garantiza estar libres de virus y que con los sistemas automáticos actualizar el kernel da el mismo trabajo que actualizar el antivirus con otros SO (de hecho mucho menos).

Pregunta 4.5: *¿Qué versión del kernel tenéis instalada en el ordenador? ¿Qué versión es la última estable que podéis descargaros de "kernel.org"? ¿E inestable?*

Una vez os hayáis bajado las fuentes del kernel deberéis descomprimirlas.

Pregunta 4.6: *¿Qué orden habéis usado para descomprimir el kernel?*

Ahora entrad en el directorio donde están las fuentes descomprimidas. El siguiente paso es configurar el kernel para adaptarlo a nuestras necesidades. Hoy no vamos a hacer grandes cosas, pero esta bien que veáis el sistema de configuración para que vosotros podáis hacerlas en el futuro :-). El kernel se puede configurar de varias formas, por ejemplo mediante línea de comandos (`make config`, probadlo).

Pregunta 4.7: *¿Cuál es la segunda pregunta en la configuración mediante línea de comandos? ¿Para que sirve?*

La configuración mediante línea de comandos es bastante lenta así que, en realidad, no suele usarse. Una opción mejor sería configurarlo automáticamente con las mínimas opciones por defecto (con la orden `make defconfig`). Esto tampoco es muy útil, ya que en ese caso es mejor usar un kernel precompilado. La forma más sencilla y útil de configurar el kernel es mediante el interfaz gráfico. Para arrancarlo ejecutad: `make gconfig` (o `make xconfig` si vuestro entorno es kde).

Pregunta 4.8: *¿Qué error da Linux si ejecutáis* `make gconfig`*?*

En este caso, como supongo que ya habéis visto al ejecutar `make gconfig` (sino, probadlo ahora), el sistema gráfico no se arranca debido a que faltan librerías. Esto es muy típico de los sistemas Linux, ya que todos los paquetes, menos los más básicos, dependen unos de otros. Esto es debido a que como el software es todo libre y conocido se usan mucho las librerías dinámicas (al contrario que en otros sistemas operativos donde cada aplicación suele

estar cerrada en si misma). Esto hace que en general el código sea más reutilizado y también más eficiente, aunque por otro lado puede generar más problemas de inconsistencias entre distintos paquetes. Para solucionar el problema que tenéis en este momento vamos a instalar manualmente, y luego ya veremos que hay detrás. Ejecutad:

```
sudo apt-get update
sudo apt-get install libglade2-dev
```

Como habréis podido comprobar si os habéis fijado en los mensajes, los sistemas de paquetes son muy cómodos ya que si un paquete depende de otros se instala todo lo necesario automáticamente. Ahora ya podéis arrancar el menú de configuración con la orden que antes os había fallado.

El interfaz gráfico, como veis consiste en señalar qué códigos del kernel (funciones) deseamos incluir y cuales no. Jugad un poco con el entorno y veréis que hay algunas opciones que tienen dos opciones y otras que tienen tres (cuando hacéis click en ellas alternan entre tres valores). Esto és así porque son opciones que se pueden incluir en el kernel (se marcan de una "v"), compilar como módulos (funciones externas que el kernel ejecuta cuando se necesitan, marcadas con color oscuro) o no incluir en absoluto (casilla en blanco). Un kernel ideal es aquel que tiene compilado solo aquello que necesita concretamente como parte del núcleo, pero esto es bastante largo y tedioso de conseguir. Además podéis ver que existe la opción de compilar el kernel para un procesador más o menos genérico.

Pregunta 4.9: *¿Para cuantas familias de procesadores se puede compilar el kernel que os habéis bajado? ¿Para qué familia se compila por defecto? ¿Es una opción adecuada para el entorno de laboratorio?*

Pregunta 4.10: *¿Qué ventajas y desventajas hay entre usar un procesador genérico (por ejemplo 386) y usar uno específico (como K8)?*

Pregunta 4.11: *Buscad la opción de incluir el soporte para AHCI SATA (un driver SATA) ¿Qué posibilidades de configuración tenéis para este módulo? ¿Cuál creéis que es la mejor opción?*

Pregunta 4.12: *¿Qué ventajas y desventajas creéis que tiene incluir muchos módulos en el sistema?*

Una vez que tengáis el kernel configurado (supuestamente con las opciones correctas para el ordenador destino), debéis compilarlo. Tened en cuenta que según las opciones que hayáis escogido luego no arrancará así que si queréis aseguraros de que funcione (aunque sea con errores) una buena forma es usando como punto de partida la configuración estándar de la herramienta gráfica (desde el `make gconfig`). En los ordenadores de laboratorio si compilamos con esta opción habitualmente conseguiremos arrancar en modo grafico. Dado que sólo vamos a hacer una prueba de compilación nos conformaremos con esta opción.

Finalmente, aunque no sea útil usarlas en el entorno de laboratorio ya que no estan instaladas para configurar un kernel que pueda arrancar, está bien que conozcáis dos opciones más para la compilación del kernel, bastante usadas en entornos mas "estables": `make oldconfig` y `make localmodconfig` (ambas usadas después del `make defconfig`).

Pregunta 4.13: *¿Qué hace cada una de las dos opciones anteriores?*

Pregunta 4.14: *Pregunta de control: ¿qué opción de configuración habéis usado? Aseguraros que la respuesta a esta pregunta es correcta si queréis llegar a arrancar el sistema.*

Para compilar deberéis ejecutar la orden: `make`. Si queréis que vaya más rápido podéis usar varios *hilos* para compilar (recomendado): `-j <hilos>`.

Pregunta 4.15: *¿Cuántos* `cores` *tiene vuestro ordenador?*

Veréis que la compilación da un error tan pronto empieza. Para poder compilar debéis ejecutar:

```
sudo apt-get install libssl-dev
```

Y volver a compilar. Ahora esto puede llevar casi una hora (depende del ordenador, de la configuración, también puede tardar menos), así que saltad a la parte de instalar paquetes e id vigilando esta compilación para continuar en cuanto acabe. Lo mejor es que enviéis la consola en la que acabáis de entrar la orden de compilar al segundo escritorio y abráis una nueva para hacer la parte de paquetes en el primer escritorio. Seguid por el siguiente punto de la práctica (la instalación de paquetes) y volved aquí cuando acabe la compilación.

Bien, si estáis leyendo esto es que ya está el kernel compilado. Ahora, si no lo ha hecho automáticamente la orden anterior (esto varía según la versión del kernel) deberéis compilar los módulos. Para ello basta con ejecutar: `make modules`. También puede tardar un buen rato así que si es el caso seguid con la instalación de paquetes por donde estabais. Por cierto, si queréis ver los módulos del sistema que ya tenéis instalado podéis hacer un: `ls /lib/modules`.

Para leer esto ya deberíais tener los módulos compilados. Si es así, el siguiente paso es instalarlos (copiar los ejecutables en el directorio donde los buscará el kernel) para ello basta con ejecutar: `make modules_install`. Podéis ver que cuando acabe esta orden aparecerá un nuevo directorio con los nuevos módulos en `/lib/modules`.

Pregunta 4.16: *¿Qué nuevo directorio ha aparecido en* `/lib/modules`*?*

Ahora hay que instalar el kernel propiamente dicho. Para ello deberéis ejecutar `make install`. Esto copia los ficheros del kernel en `/boot/`. Fijaros que si miráis en este directorio veréis los ficheros de arranque de todos los núcleos que tenéis en el sistema y unos ficheros genéricos de arranque que son enlaces a los ficheros reales. Para distinguirlos basta con que hagáis: `ls -l /boot/`.

Finalmente, dependiendo de si ya está creado o no, deberéis crear el fichero `initrd` de vuestra versión del kernel. Este fichero monta un sistema de ficheros en RAM mientras no se acaba de cargar la raíz del sistema. Para crearlo deberéis ir al directorio `boot` y ejecutar:

```
update-initramfs -c -k <numerosdeversiondelkernel, ej: 2.6.85>
```

Pregunta 4.17: *¿Tenéis ya un fichero* `initrd` *correspondiente a la versión nueva del kernel que habéis compilado? ¿Qué orden en concreto debéis/deberíais ejecutar para crear el* `initrd` *si no existiera?*

Pregunta 4.18: *¿Cuántos ficheros nuevos se han creado en el directorio* `/boot/` *en total?*

Ya está, si ahora quisierais arrancar con el nuevo kernel bastaría con que cambiarais la configuración del gestor de arranque para que apunte al nuevo núcleo que habéis instalado. Si queréis que el sistema lo haga de forma automática podéis teclear `update-grub` (a veces esto también lo hace directamente el `make install`). De todas formas no perdáis de vista que un kernel genérico se puede instalar mucho más rápida y fácilmente con las instrucciones de paquetes normales. Este proceso manual en realidad solo se usaría si se configura un núcleo específico debido a unas necesidades concretas.

Pregunta 4.19: *¿Que versión del kernel teníais instalada? ¿Cual acabáis de instalar?*

Podéis probar a reiniciar el sistema y arrancar con la nueva versión del kernel (acordaros de **cargar el sistema desde disco y NO desde la red**, sino se borrará vuestra instalación). Si no os arranca el sistema gráfico probad a cambiar a modo consola (recordad, `ctrl + alt + F1`).

Pregunta 4.20: *¿Arranca el sistema que acabáis de compilar? ¿En modo gráfico o solo texto? ¿Da errores? ¿A qué creéis que se deben los errores?*

4.3. Instalación de paquetes

Existen 3 formas principales de instalar paquetes:

- A partir de paquetes genéricos compilados y ya preparados.
- A partir de los fuentes (así estamos instalando el kernel).
- A partir del sistema de paquetes de nuestra distribución (deb para las debian o rpm para las redhat).

A lo largo de la práctica vamos a instalar un paquete de cada tipo y de paso veremos las ventajas y desventajas de cada uno.

4.3.1. Instalar un paquete mediante el sistema de paquetes

En primer lugar vamos a usar el sistema de paquetes para instalar un nuevo paquete. Esta es la forma más cómoda de hacerlo (como ya habéis visto), y en general la que menos problemas da. Eso sí, tenéis que tener en cuenta que hay varios sistemas de paquetes y cada distribución usa uno u otro. De todas formas la idea es la misma, así que no debería daros ningún problema pasar de una a otra. Aquí veremos el sistema de Debian: `apt-get`, que instala paquetes `.deb`. Otro de los sistemas más típicos es el `rpm` de Redhat (el comando se llama `rpm` e instala paquetes `.rpm`). De Suse mejor no hablamos aunque OpenSuse actualmente usa básicamente `rpm`. Este tipo de instalación puede además daros problemas con sistemas muy antiguos ya que buscan automáticamente las fuentes por internet y los repositorios pueden estar demasiado actualizados. Por ello, este sistema es bueno para usarlo con sistemas actualizados y paquetes de uso común.

Es importante que sepáis que el núcleo también puede instalarse mediante este sistema, usando los paquetes adecuados, incluso podemos actualizar toda la distribución así (en Ubuntu, en mi experiencia, esto suele funcionar bien).

Vamos a intentar instalar un programa llamado `fish fillets`.

La primera ventaja del sistema `apt-get` es que se conecta directamente a Internet a las direcciones especificadas en el fichero `sources.list`. Podéis ver este fichero y comprobar las direcciones que contiene, además de ver qué opciones indica (podéis comprobar que significan las opciones mediante `man sources.list`). Fijaros que en realidad Ubuntu, básicamente utiliza las fuentes "inestables" de Debian.

Pregunta 4.21: *¿Que significa que un paquete Debian sea* `testing`*? ¿Y* `unstable`*?*

Lo primero que debemos hacer antes de instalar un paquete es actualizar la lista de paquetes para que el sistema sepa donde buscar un paquete nuevo y cuales son las últimas versiones. Esto se hace con la orden `apt-get update` y su función (que hemos utilizado al principio de la práctica) es conectarse a los repositorios y saber que paquetes y versiones exactas existen en este momento de forma que si nos actualizamos descarguemos versiones coherentes (compatibles entre ellas).

Una vez actualizada la lista podemos intentar instalar el paquete mediante el comando: `apt-get install nombrepaquete`. Probad con el nombre "fish".

Fijaros que, cuando no tenemos mucha información, puede ser difícil localizar un paquete por la línea de comandos. Para facilitar esta tarea hay un comando de búsqueda de paquetes: `apt-cache search palabra`. Podéis probar a buscar substituyendo palabra por "fish", o "fillets" o ambas a la vez (separadas por un espacio).

Pregunta 4.22: *¿Cual es el nombre del paquete que contiene el programa* `fish fillets`*?*

Pues ya lo tenéis, instalad el paquete con `apt-get install` y el nombre. Fijaros que el sistema automáticamente resuelve las dependencias con otros paquetes y los instala si es necesario.

Pregunta 4.23: *¿El sistema ha instalado algún paquete adicional? ¿Cuál?*

Otras opciones interesantes de `apt-get` son: `update`, `remove`, `upgrade` y `dist-upgrade`.

Pregunta 4.24: *¿Para qué sirve, en pocas palabras, cada una de las anteriores opciones?*

Dado que la forma más cómoda de instalar programas es mediante el sistema de paquetes, una última opción interesante es añadir más repositorios remotos desde los que bajar nuevos programas. Vamos a probar a añadir un nuevo repositorio. En primer lugar intentad instalaros el paquete "ufoai". No debería funcionar porque no existe en los repositorios que tenéis configurados en vuestro fichero `sources.list`. Así pues, editad el fichero `sources.list` y añadir al final la línea:

```
deb http://archive.getdeb.net/ubuntu xenial-getdeb games
```

A continuación, deberéis decirle a vuestro sistema que confíe en esta fuente de software (ojo, si no queréis comprometer la seguridad de vuestro sistema aseguraros de CONFIAR en la fuente antes de hacerlo :-)

```
wget -q -O- http://archive.getdeb.net/getdeb-archive.key | sudo apt-key add -
```

Ya podéis probar a instalar el paquete "ufoai"... aunque ojo, lleva un ratito largo dependiendo de la carga de la red así que en todo caso podéis dejarlo para el final de la práctica :-). Si queréis saber el repositorio que habéis instalado visitad: `www.playdeb.net`.

Pregunta 4.25: *¿Para que sirve la orden* `wget`*? ¿Y el comando* `apt-key`*? ¿Qué significa el – final?*

4.3.2. Instalar un paquete ya compilado

Este sistema es el más directo y básicamente, consiste en bajarse el ejecutable (y todos los ficheros asociados) y copiarlo en algún sitio de nuestro ordenador. Tiene la ventaja de ser muy fácil y rápido y la desventaja de que es un sistema que no informa al SO del nuevo software para que lo gestione. Vamos a probarlo instalándonos el programa "Bos Wars".

Primero de todo debemos bajar el paquete que contiene el programa. Bajaros el paquete que dice que contiene el ejecutable para Linux (no las fuentes). Este pretende ser un ejecutable estático, esto quiere decir que intenta no depender de otros paquetes para ejecutarse, a cambio de ello ocupa más espacio ya que intenta incluir todas las librerías. El problema es que lo de "ejecutable estático y aislado" no deja de ser una declaración de intenciones cada vez más dificil de cumplir.

Una vez que lo tengáis descomprimidlo, acceded al directorio del programa y ejecutadlo. Si todo ha ido bien el sistema funcionará. Si falla algo, (dependiendo de la instalación concreta y el paquete) no. Si a continuación queréis desinstalar el programa deberéis borrar a mano tanto el fichero del paquete como todo el directorio del programa.

Pregunta 4.26: *¿Qué ordenes habéis introducido para instalar y ejecutar el programa? ¿Ha funcionado? ¿Se puede instalar mediante el sistema de paquetes? ¿Funciona ahora?*

Pregunta 4.27: *¿Que instrucciones deberíais usar para eliminar totalmente el programa?*

4.3.3. Instalar un paquete a partir de los fuentes

Esto es básicamente lo que estamos haciendo/hemos hecho con el kernel. Básicamente consiste en los siguientes pasos:

- Bajarse las fuentes de internet.
- Descomprimirlas.
- Configurar el sistema (con la orden `./configure`).

- Compilarlo (con la orden `make`).
- Instalarlo (con la orden `make install`, o en todo caso `sudo make install` ya que hace falta que seáis "root").

El gran problema de este sistema es que cada distribución puede mover las cosas de sitio y por lo tanto esto solo funciona realmente bien en sistemas totalmente comprometidos con el estándar Debian (como Knoppix) o con paquetes que no dependen de otros.

Un segundo inconveniente es que esto no instala los paquetes de los que dependen los anteriores automáticamente. Así pues cuando el `configure` o el `make` dan un error hay que buscar las dependencias a mano (usualmente leyendo los mensajes) e instalarlas (acordaros de lo que nos ha pasado con el `make gconfig` del núcleo). Como un paquete puede depender fácilmente de 5 o 6 más y estos a su vez pueden depender de otros... pues la tarea puede prolongarse bastante (fijaros que os he hecho instalar `libglade` mediante el sistema de paquetes y no a mano).

Ultimo problema, por si los anteriores eran pocos. Si os fijáis al compilar el núcleo no hemos usado exactamente las órdenes que os he puesto un poco más arriba. Esto es debido a que una cosa es el sistema en general y otra cada caso en concreto. Cuando los que han hecho el paquete no siguen exactamente el criterio habitual lo único que se puede hacer es empezar a leer los ficheros "README", "INSTALL", etc. que suelen acompañar a las fuentes.

Dicho todo esto, podéis intentar la instalación del programa FreeCiv. Id a su página web en `sourceforge`, bajaros los fuentes de la **versión 2.4.3** y seguid los pasos que os he dado más arriba para compilarlo e instalarlo.

Pregunta 4.28: *¿Qué error da cuando intentais configurar* `freeciv`*? ¿Cómo se puede solucionar? Como veis la solución no es inmediata pero la cantidad de opciones es lo suficientemente limitada como para ir probando. Hacedlo y volved a configurar hasta que funcione. ¿Cuántos paquetes parecen susceptibles de solucinar el problema? ¿Qué paquete lo ha solucionado realmente?*

Cuando acabéis la instalación probad a ejecutar `freeciv-gtk2`. Si todo ha ido bien funcionará.

Pregunta 4.29: *¿Qué ordenes habéis usado para instalarlo? ¿Aparece el nuevo programa en los menús del sistema? ¿Se encuentra en el* `PATH`*?*

4.3.4. Sistemas gráficos de instalación de paquetes

Finalmente, tenéis que pensar, que como casi siempre pasa en el mundo del software libre, las opciones vistas aquí no son ni las únicas, ni quizás las más cómodas (eso si, son las más básicas). A partir de estas hay otras herramientas de instalación que con los mismos principios hacen la misma faena pero que os pueden resultar más agradables (o no, esto ya depende de cada uno). Algunas de estas herramientas son: `synaptic`, `kpackage`, `tasksel`, `yast`... y un largo etcétera. Podéis probar a ejecutar el centro de software de Ubuntu y navegar un poco por él. Como podéis comprobar es tan solo un front-end de `apt-get` que puede gustaros más que este, pero fijaros en que usa los mismos principios. La configuración se realiza a partir de la opción de herramientas del sistema.

Pregunta 4.30: *¿Qué opción de* `herramientas` *hay que abrir y cómo está organizado el contenido del fichero* `sources.list` *en el sistema gráfico? ¿Qué opción os da más opciones, el fichero texto o el configurador gráfico?*

Finalmente podéis comprobar que todos los sistemas actuales de Linux vienen con un sistema que detecta automáticamente actualizaciones a través de internet. Es muy útil que mantengáis el sistema actualizado ya que en Linux esta es la mejor protección antivirus que hay, así que es recomendable que lo dejéis activado si tenéis una buena conexión en casa. En caso contrario lo mejor es desactivarlo y que periódicamente os acordéis de actualizar la distribución.