

**Propuesta de un Mecanismo de Seguridad para el Intercambio de Datos de  
Usuario en Redes de Próxima Generación**



**Universidad  
del Cauca**

**Jaime Andrés Oliva Ortega  
Fabio Joaquín Fuertes Montenegro**

**ANEXO A  
ENDURECIMIENTO DEL KERNEL DE LINUX EN DEBIAN**

**Universidad del Cauca  
Facultad de Ingeniería Electrónica y Telecomunicaciones  
Departamento de Telemática  
Popayán, Octubre de 2009**

## TABLA DE CONTENIDO

|   |   |
|---|---|
| 1. Introducción .....                     | 1 |
| 2. Requerimientos preliminares .....      | 1 |
| 3. Parchar el Kernel .....                | 3 |
| 4. Configurar el Kernel Endurecido .....  | 4 |
| 5. Compilación del Kernel endurecido..... | 6 |
| 6. Instalación del Kernel Endurecido..... | 9 |

## LISTA DE FIGURAS

|            |  |    |
|------------|--|----|
| Figura 1.  | Instalación de paquetes necesarios para la compilación del Kernel .....        | 1  |
| Figura 2.  | Instalación del paquete <i>libncurses5-dev</i> .....                           | 1  |
| Figura 3.  | Verificación de la instalación del paquete <i>initramfs-tool</i> .....         | 2  |
| Figura 4.  | Descarga de las fuentes del Kernel.....  | 2  |
| Figura 5.  | Descarga del parche Grsecurity .....   | 3  |
| Figura 6.  | Mover el parche Grsecurity al directorio del Kernel .....                      | 3  |
| Figura 7.  | Descomprimir y parchar las fuentes del Kernel .....                            | 4  |
| Figura 8.  | Lanzar el menú de consola .....  | 4  |
| Figura 9.  | Escoger Security Options.....  | 5  |
| Figura 10. | Escoger Grsecurity .....   | 5  |
| Figura 11. | Marcar Grsecurity e ir a Security Level .....                                  | 6  |
| Figura 12. | Marcar el nivel Medium.....  | 6  |
| Figura 13. | Realizar limpieza de la configuración previa a la compilación del Kernel ..... | 7  |
| Figura 14. | Compilación del Kernel endurecido .....  | 8  |
| Figura 15. | Terminación del proceso de compilación del Kernel endurecido .....             | 9  |
| Figura 16. | Instalación del Kernel endurecido .....  | 10 |
| Figura 17. | Verificación del archivo <i>vmlinuz</i> generado .....                         | 10 |
| Figura 18. | Verificación de la versión actual del Kernel antes del reinicio .....          | 10 |
| Figura 19. | Verificación de la versión actual del Kernel después del reinicio.....         | 11 |
| Figura 20. | Verificación de la versión actual del Kernel en el host <i>xeratul</i> .....   | 11 |
| Figura 21. | Verificación de la versión actual del Kernel en el host <i>dragoon</i> .....   | 11 |
| Figura 22. | Verificación de la versión actual del Kernel en el host <i>fenix</i> .....     | 11 |

## 1. Introducción

Endurecer un software implica incrementar la protección, prevención y detección de amenazas de seguridad, en este caso propias de la ejecución de código malicioso sobre el Núcleo o Kernel del sistema operativo Linux, que pueden causar desde la obtención de privilegios administrativos sobre la máquina hasta el bloqueo de la misma. Para realizar esta tarea, se hará uso del parche Grsecurity el cual brinda las características de seguridad mencionadas.

Todos los pasos seguidos en este documento son ejecutados con privilegios de usuario *root* sobre el host *zealot*, el cual, cuenta con sistema operativo Debian 5 y Kernel 2.6.26-2 Instalado con la distribución. Cabe anotar que no es necesario volver a realizar todo el proceso para los otros equipos, ya que como resultado esta guía se obtendrá un paquete genérico de Debian, que puede instalarse fácilmente en otros equipos con este sistema operativo.

## 2. Requerimientos preliminares

Para compilar el Kernel, primero es necesario instalar algunos paquetes específicos como se muestra en la Figura 1.



```
ffuertes@fenix: ~  
zealot:~# apt-get install patch bin86 kernel-package build-essential  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias
```

Figura 1. Instalación de paquetes necesarios para la compilación del Kernel

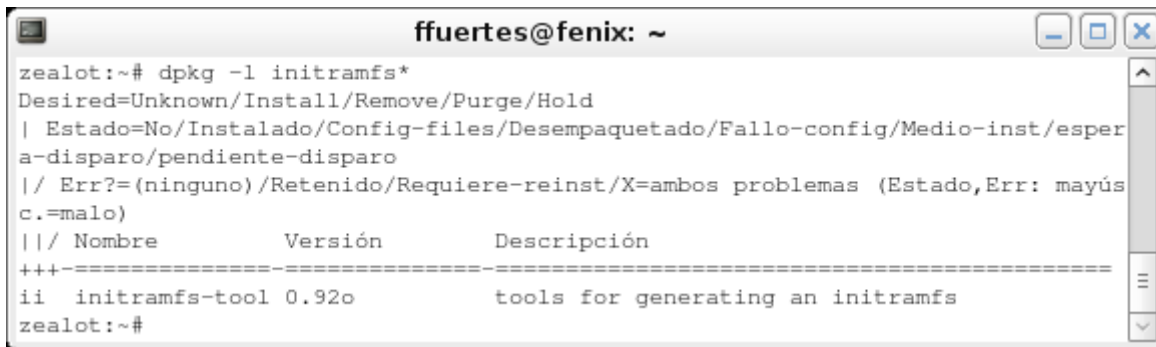
Para realizar una configuración del Kernel en una consola grafica (*make menuconfig*), se instala el paquete mostrado en la Figura 2.



```
ffuertes@fenix: ~  
zealot:~# apt-get install libncurses5-dev  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias
```

Figura 2. Instalación del paquete *libncurses5-dev*

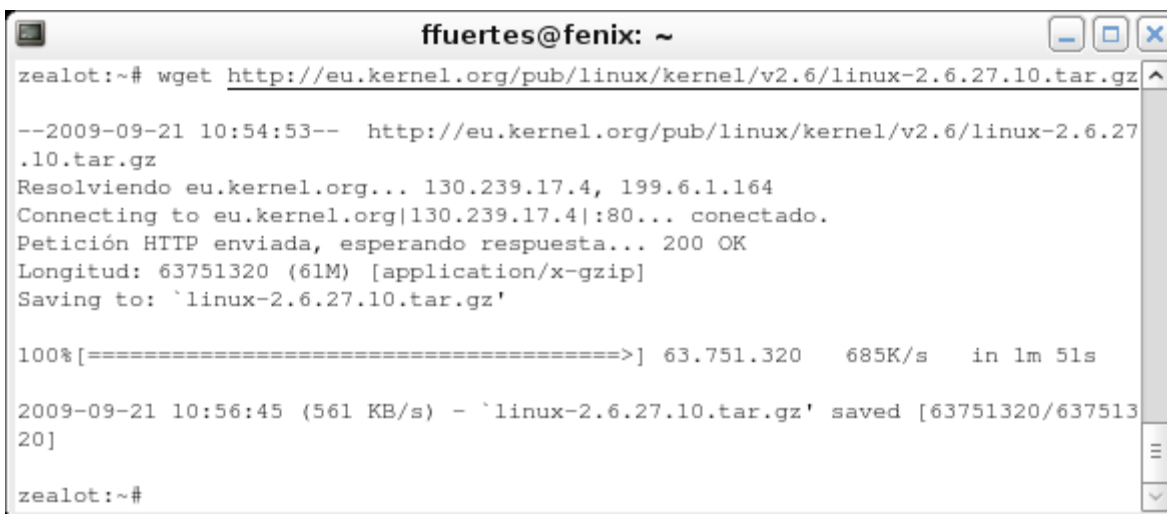
Seguidamente, es necesario verificar si el paquete *initramfs-tool*, se encuentra instalado en el equipo (ver Figura 3).



```
ffuertes@fenix: ~
zealot:~# dpkg -l initramfs*
Desired=Unknown/Install/Remove/Purge/Hold
| Estado=No/Instalado/Config-files/Desempaquetado/Fallo-config/Medio-inst/esper
a-disparo/pendiente-disparo
|/ Err?=(ninguno)/Retenido/Requiere-reinst/X=ambos problemas (Estado,Err: mayús
c.=malo)
||/ Nombre          Versión          Descripción
+++-----
ii  initramfs-tool    0.92o           tools for generating an initramfs
zealot:~#
```

Figura 3. Verificación de la instalación del paquete initramfs-tool

Ahora se descargan las fuentes del Kernel 2.6.27, las cuales en el momento de llevar a cabo este procedimiento corresponden a la versión estable actualizada (ver Figura 4). Igualmente se descarga el parche Grsecurity para dicho Kernel (ver Figura 5).



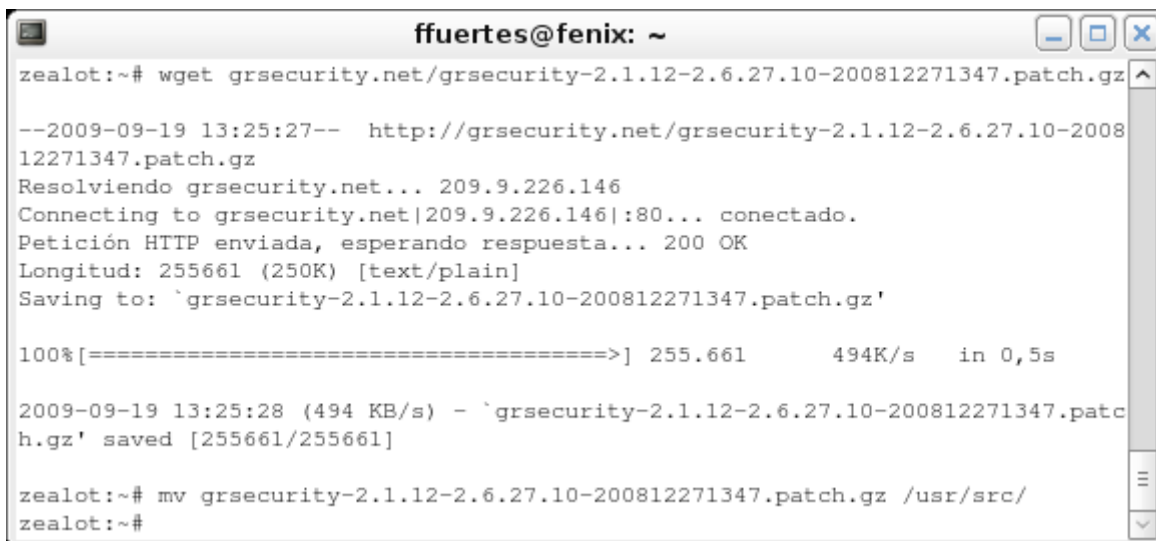
```
ffuertes@fenix: ~
zealot:~# wget http://eu.kernel.org/pub/linux/kernel/v2.6/linux-2.6.27.10.tar.gz
--2009-09-21 10:54:53-- http://eu.kernel.org/pub/linux/kernel/v2.6/linux-2.6.27
.10.tar.gz
Resolviendo eu.kernel.org... 130.239.17.4, 199.6.1.164
Connecting to eu.kernel.org|130.239.17.4|:80... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 63751320 (61M) [application/x-gzip]
Saving to: `linux-2.6.27.10.tar.gz'

100%[=====>] 63.751.320  685K/s  in 1m 51s

2009-09-21 10:56:45 (561 KB/s) - `linux-2.6.27.10.tar.gz' saved [63751320/637513
20]

zealot:~#
```

Figura 4. Descarga de las fuentes del Kernel



```
ffuertes@fenix: ~
zealot:~# wget grsecurity.net/grsecurity-2.1.12-2.6.27.10-200812271347.patch.gz
--2009-09-19 13:25:27--  http://grsecurity.net/grsecurity-2.1.12-2.6.27.10-2008
12271347.patch.gz
Resolviendo grsecurity.net... 209.9.226.146
Connecting to grsecurity.net|209.9.226.146|:80... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 255661 (250K) [text/plain]
Saving to: `grsecurity-2.1.12-2.6.27.10-200812271347.patch.gz'

100%[=====>] 255.661      494K/s   in 0,5s

2009-09-19 13:25:28 (494 KB/s) - `grsecurity-2.1.12-2.6.27.10-200812271347.patc
h.gz' saved [255661/255661]

zealot:~# mv grsecurity-2.1.12-2.6.27.10-200812271347.patch.gz /usr/src/
zealot:~#
```

Figura 5. Descarga del parche Grsecurity

Los dos archivos descargados se mueven al directorio `/usr/src`. Posteriormente se procede a descomprimir el archivo del Kernel mediante el siguiente comando:

```
# tar xzvf linux-2.6.27.10.tar.gz
```

Se crea un enlace simbólico llamado `linux`, apuntando al directorio del nuevo Kernel para hacer un poco más sencillas las tareas posteriores mediante el siguiente comando:

```
# ln -s linux-2.6.27.10 linux
```

### 3. Parchar el Kernel

Ahora que todos los requerimientos preliminares se han cumplido se procede a parchar el Kernel siguiendo los siguientes pasos:

Mover el parche Grsecurity al nuevo directorio y entrar en este (ver Figura 6).



```
ffuertes@fenix: ~
zealot:/usr/src# mv grsecurity-2.1.12-2.6.27.10-200812271347.patch.gz linux/
zealot:/usr/src# cd linux
zealot:/usr/src/linux#
```

Figura 6. Mover el parche Grsecurity al directorio del Kernel

Descomprimir y parchar las fuentes del Kernel (ver Figura 7).



```
ffuertes@fenix: ~
zealot:/usr/src/linux# gunzip < grsecurity-2.1.12-2.6.27.10-200812271347.patch.gz | patch -p1
```

Figura 7. Descomprimir y parchar las fuentes del Kernel

Una vez se ha realizado este proceso, se han modificado las fuentes para realizar una instalación con elementos de seguridad avanzados.

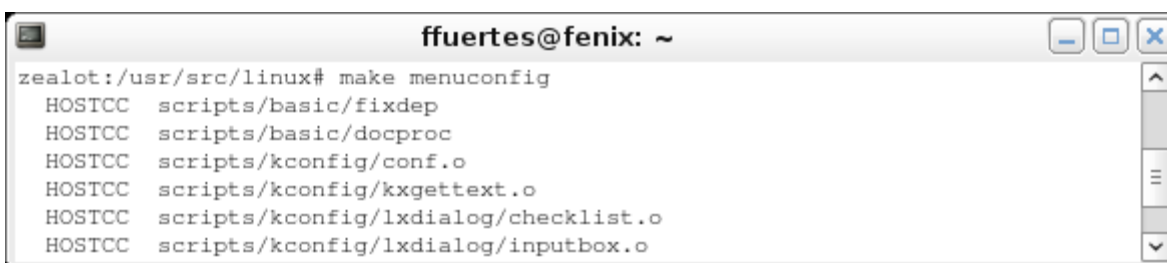
#### 4. Configurar el Kernel Endurecido

Para realizar la configuración, en este caso se utiliza el menú de consola (*make menuconfig*), utilizando el paquete *libncurses5-dev*. Sin embargo es posible realizar la configuración en una consola de solo texto (*make config*), o en interfaz gráfica (*make xconfig*).

Grsecurity tiene niveles de seguridad predefinidos: bajo, medio y alto, también puede ser configurado en un nivel personalizado donde se puede escoger habilitar o no, cada una de las opciones.

Para este caso se utilizará una configuración en nivel medio que asegura la protección contra las principales amenazas de ejecución de código malicioso existentes actualmente. Además, el nivel medio no es tan restrictivo como el alto, pues este puede deshabilitar algunos servicios necesarios en el prototipo. Los pasos necesarios para la configuración son los siguientes:

Lanzar el menú de consola (Figura 8):



```
ffuertes@fenix: ~
zealot:/usr/src/linux# make menuconfig
HOSTCC scripts/basic/fixdep
HOSTCC scripts/basic/docproc
HOSTCC scripts/kconfig/conf.o
HOSTCC scripts/kconfig/kxgettext.o
HOSTCC scripts/kconfig/lxdialog/checklist.o
HOSTCC scripts/kconfig/lxdialog/inputbox.o
```

Figura 8. Lanzar el menú de consola

Las figuras 9 a 11 muestran el proceso de configuración del nivel medio de Grsecurity en el menú de consola durante la puesta a punto de un Kernel a la medida.

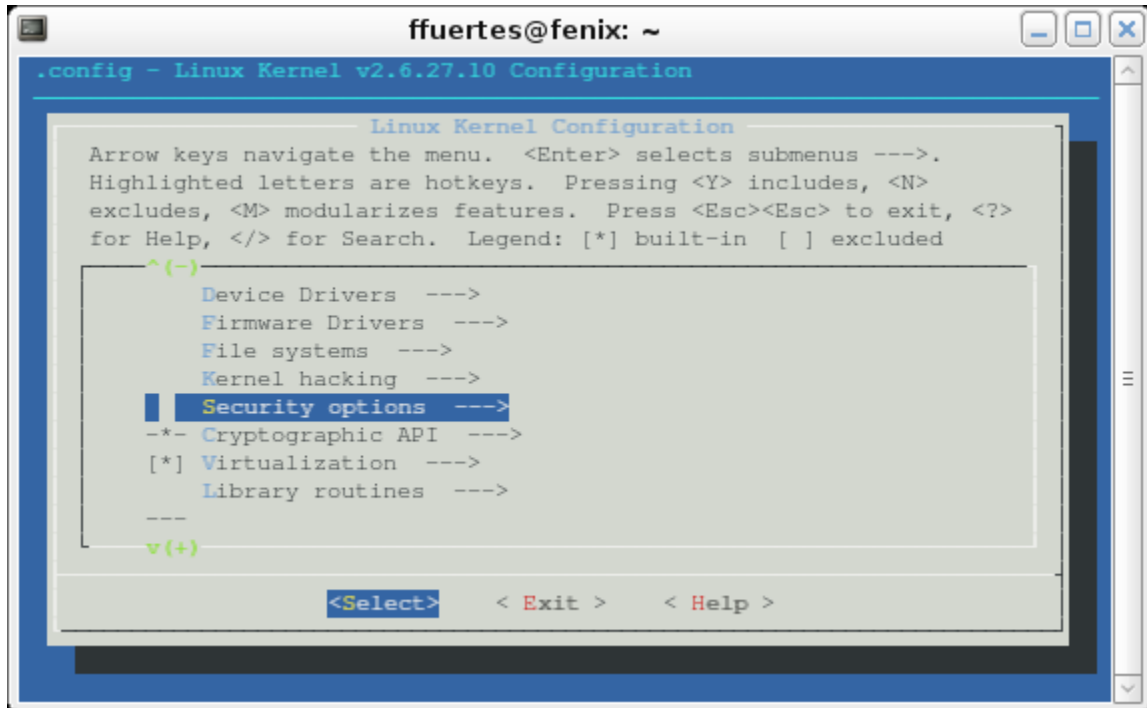


Figura 9. Escoger Security Options

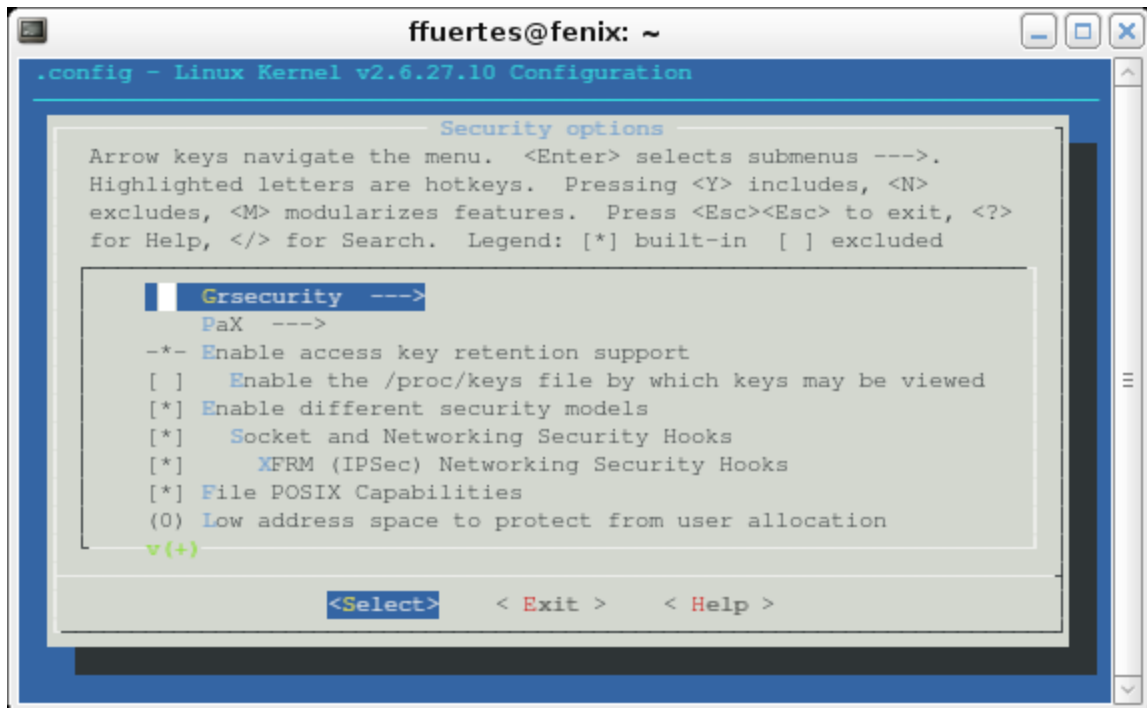


Figura 10. Escoger Grsecurity



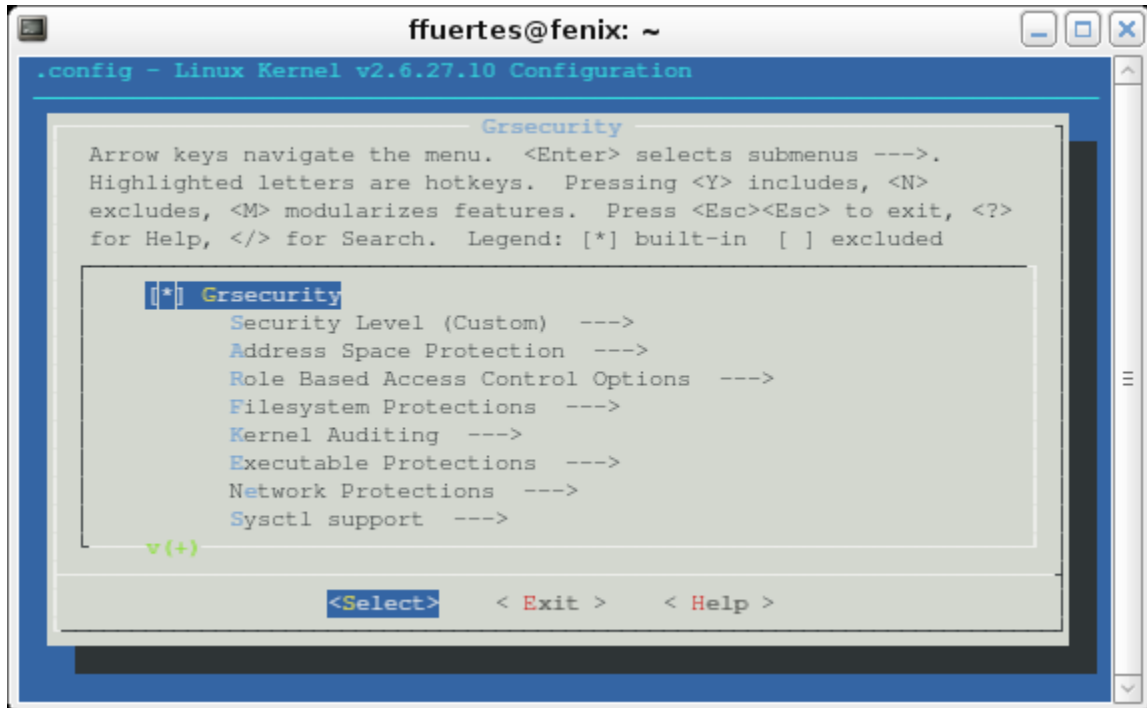


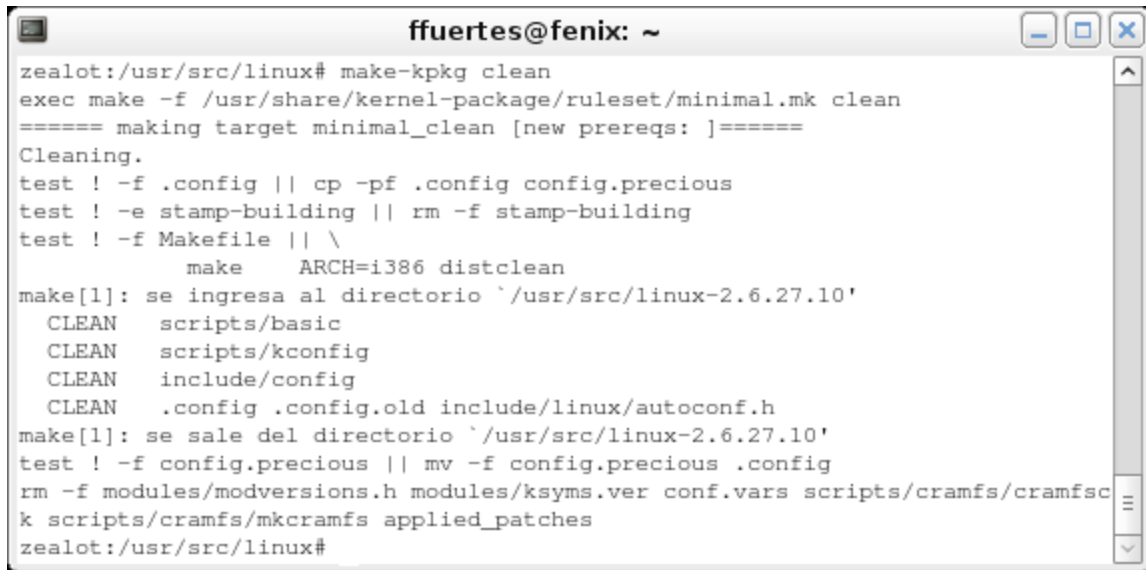
Figura 11. Marcar Grsecurity e ir a Security Level



Figura 12. Marcar el nivel Medium

## 5. Compilación del Kernel endurecido

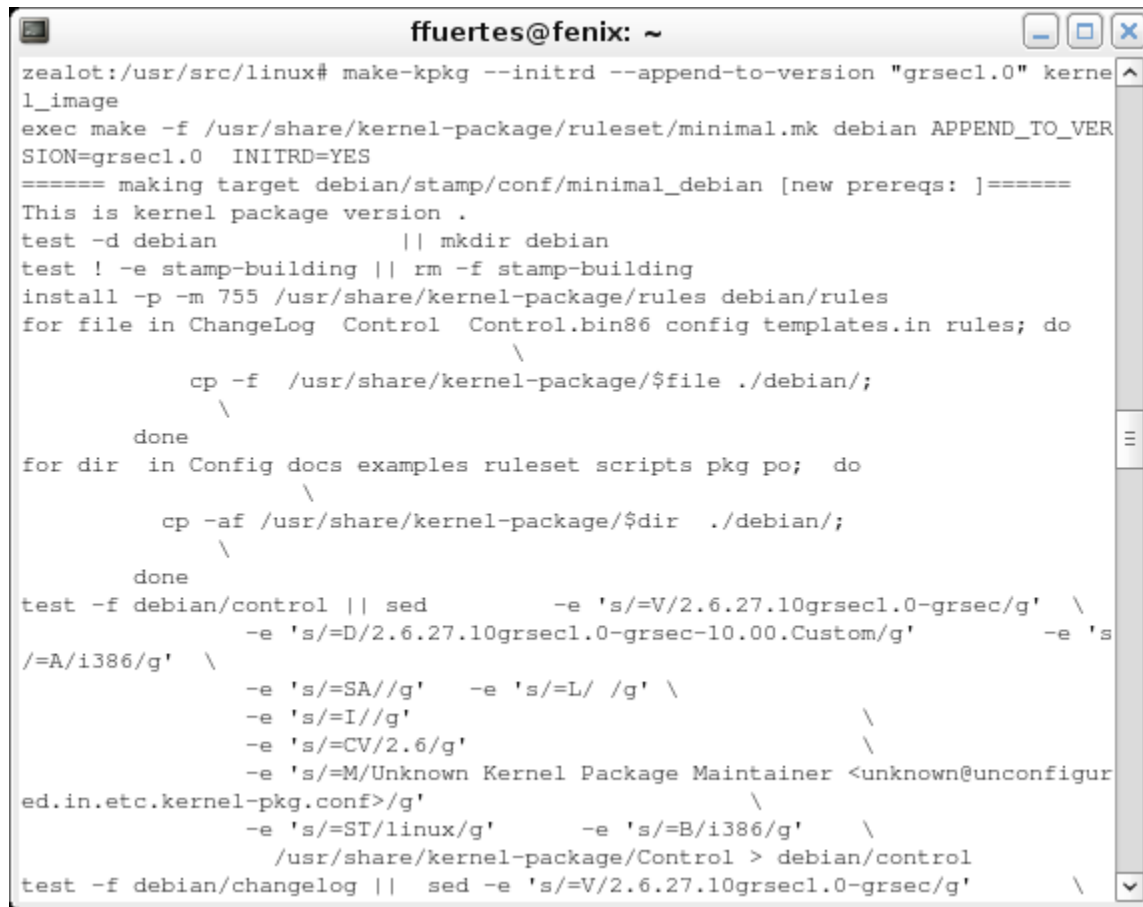
Ahora se procede a compilar las fuentes ya configuradas con las opciones de seguridad de Grsecurity siguiendo los siguientes pasos:  
Realizar limpieza de la configuración (Figura 13).



```
ffuertes@fenix: ~
zealot:/usr/src/linux# make-kpkg clean
exec make -f /usr/share/kernel-package/ruleset/minimal.mk clean
===== making target minimal_clean [new prereqs: ]=====
Cleaning.
test ! -f .config || cp -pf .config config.precious
test ! -e stamp-building || rm -f stamp-building
test ! -f Makefile || \
    make ARCH=i386 distclean
make[1]: se ingresa al directorio `/usr/src/linux-2.6.27.10'
  CLEAN  scripts/basic
  CLEAN  scripts/kconfig
  CLEAN  include/config
  CLEAN  .config .config.old include/linux/autoconf.h
make[1]: se sale del directorio `/usr/src/linux-2.6.27.10'
test ! -f config.precious || mv -f config.precious .config
rm -f modules/modversions.h modules/ksyms.ver conf.vars scripts/cramfs/cramfsc
k scripts/cramfs/mkcramfs applied_patches
zealot:/usr/src/linux#
```

Figura 13. Realizar limpieza de la configuración previa a la compilación del Kernel

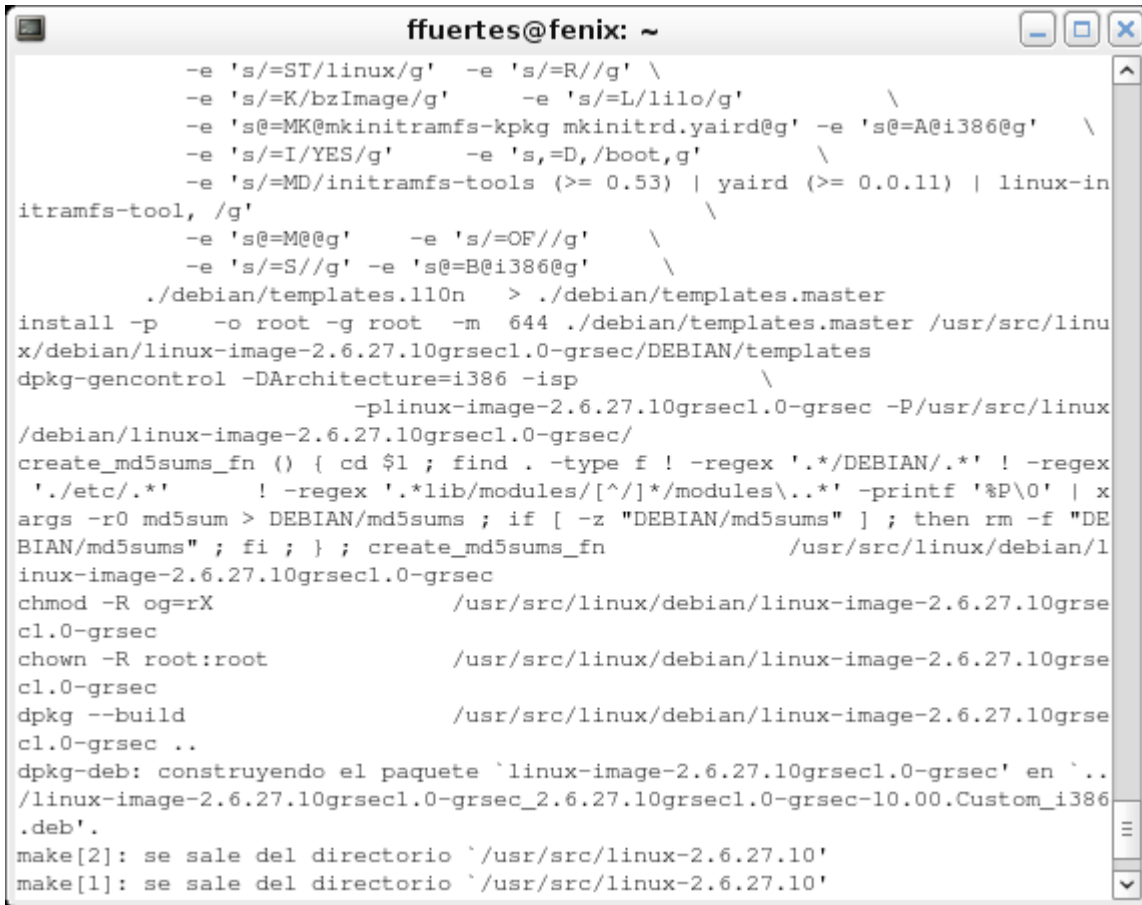
Lanzar la compilación como tal (Figura 14)



```
ffuertes@fenix: ~
zealot:/usr/src/linux# make-kpkg --initrd --append-to-version "grsecl.0" kernel_image
exec make -f /usr/share/kernel-package/ruleset/minimal.mk debian APPEND_TO_VERSION=grsecl.0 INITRD=YES
===== making target debian/stamp/conf/minimal_debian [new prereqs: ]=====
This is kernel package version .
test -d debian || mkdir debian
test ! -e stamp-building || rm -f stamp-building
install -p -m 755 /usr/share/kernel-package/rules debian/rules
for file in ChangeLog Control Control.bin86 config templates.in rules; do
    cp -f /usr/share/kernel-package/$file ./debian/;
done
for dir in Config docs examples ruleset scripts pkg po; do
    cp -af /usr/share/kernel-package/$dir ./debian/;
done
test -f debian/control || sed -e 's/=V/2.6.27.10grsecl.0-grsec/g' \
    -e 's/=D/2.6.27.10grsecl.0-grsec-10.00.Custom/g' -e 's/=A/i386/g' \
    -e 's/=SA//g' -e 's/=L/ /g' \
    -e 's/=I//g' \
    -e 's/=CV/2.6/g' \
    -e 's/=M/Unknown Kernel Package Maintainer <unknown@unconfigured.in.etc.kernel-pkg.conf>/g' \
    -e 's/=ST/linux/g' -e 's/=B/i386/g' \
    /usr/share/kernel-package/Control > debian/control
test -f debian/changelog || sed -e 's/=V/2.6.27.10grsecl.0-grsec/g' \
```

Figura 14. Compilación del Kernel endurecido

Esta compilación puede tardar varias horas, dependiendo de la velocidad de procesador y memoria RAM del equipo donde se esté realizando la operación. Para el caso del host zealot la duración de este proceso fue de alrededor de 3 horas, después de las cuales se termina el proceso como lo muestra la Figura 15.



```
ffuertes@fenix: ~
-e 's/=ST/linux/g' -e 's/=R//g' \
-e 's/=K/bzImage/g' -e 's/=L/lilo/g' \
-e 's@=MK@mkiniramfs-kpkg mkinitrd.yaird@g' -e 's@=A@i386@g' \
-e 's/=I/YES/g' -e 's,=D,/boot,g' \
-e 's/=MD/initramfs-tools (>= 0.53) | yaird (>= 0.0.11) | linux-in
itramfs-tool, /g'
-e 's@=M@g' -e 's/=OF//g' \
-e 's/=S//g' -e 's@=B@i386@g' \
./debian/templates.110n > ./debian/templates.master
install -p -o root -g root -m 644 ./debian/templates.master /usr/src/linu
x/debian/linux-image-2.6.27.10grsec1.0-grsec/DEBIAN/templates
dpkg-gencontrol -DArchitecture=i386 -isp \
-plinux-image-2.6.27.10grsec1.0-grsec -P/usr/src/linux
/debian/linux-image-2.6.27.10grsec1.0-grsec/
create_md5sums_fn () { cd $1 ; find . -type f ! -regex '.*DEBIAN/.*' ! -regex
'./etc/.*' ! -regex '.*lib/modules/[^/]*modules\..*' -printf '%P\0' | x
args -r0 md5sum > DEBIAN/md5sums ; if [ -z "DEBIAN/md5sums" ] ; then rm -f "DE
BIAN/md5sums" ; fi ; } ; create_md5sums_fn /usr/src/linux/debian/l
inux-image-2.6.27.10grsec1.0-grsec
chmod -R og=rX /usr/src/linux/debian/linux-image-2.6.27.10grse
c1.0-grsec
chown -R root:root /usr/src/linux/debian/linux-image-2.6.27.10grse
c1.0-grsec
dpkg --build /usr/src/linux/debian/linux-image-2.6.27.10grse
c1.0-grsec ..
dpkg-deb: construyendo el paquete `linux-image-2.6.27.10grsec1.0-grsec' en `..
/linux-image-2.6.27.10grsec1.0-grsec_2.6.27.10grsec1.0-grsec-10.00.Custom_i386
.deb'.
make[2]: se sale del directorio `/usr/src/linux-2.6.27.10'
make[1]: se sale del directorio `/usr/src/linux-2.6.27.10'
```

Figura 15. Terminación del proceso de compilación del Kernel endurecido

## 6. Instalación del Kernel Endurecido

El resultado de realizar el proceso de compilación es un paquete *.deb* (paquete instalable específico para distribuciones de Linux basadas en Debian), que se encuentra en el directorio `/usr/src`, el cual puede instalarse del mismo modo que cualquier otro paquete para esta distribución mediante comando: `dpkg -i paquete.deb`, como lo muestra la Figura 16.

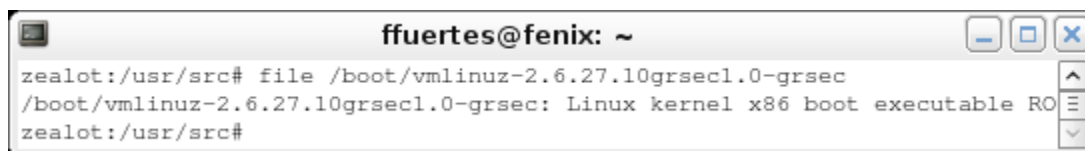


```
ffuertes@fenix: ~
zealot:/usr/src# dpkg -i linux-image-2.6.27.10grsec1.0-grsec_2.6.27.10grsec1.0-grsec-10.00.Custom_i386.deb
Seleccionando el paquete linux-image-2.6.27.10grsec1.0-grsec previamente no seleccionado.
(Leyendo la base de datos ...
28877 ficheros y directorios instalados actualmente.)
Desempaquetando linux-image-2.6.27.10grsec1.0-grsec (de linux-image-2.6.27.10grsec1.0-grsec_2.6.27.10grsec1.0-grsec-10.00.Custom_i386.deb) ...
Done.
Configurando linux-image-2.6.27.10grsec1.0-grsec (2.6.27.10grsec1.0-grsec-10.00.Custom) ...
Running depmod.
Finding valid ramdisk creators.
Using mkinitramfs-kpkg to build the ramdisk.
Running postinst hook script update-grub.
Searching for GRUB installation directory ... found: /boot/grub
Searching for default file ... found: /boot/grub/default
Testing for an existing GRUB menu.lst file ... found: /boot/grub/menu.lst
Searching for splash image ... none found, skipping ...
Found kernel: /boot/vmlinuz-2.6.27.10grsec1.0-grsec
Found kernel: /boot/vmlinuz-2.6.26-2-686
Found kernel: /boot/vmlinuz-2.6.26-1-686
Updating /boot/grub/menu.lst ... done

zealot:/usr/src#
```

Figura 16. Instalación del Kernel endurecido

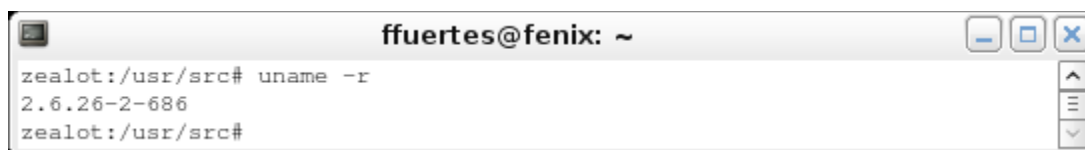
Durante la instalación se genera una imagen *initrd* la cual corresponde al Kernel instalado. Posteriormente se revisa las características de la imagen del Kernel generado (ver Figura 17).



```
ffuertes@fenix: ~
zealot:/usr/src# file /boot/vmlinuz-2.6.27.10grsec1.0-grsec
/boot/vmlinuz-2.6.27.10grsec1.0-grsec: Linux kernel x86 boot executable RO
zealot:/usr/src#
```

Figura 17. Verificación del archivo vmlinuz generado

Finalmente, es necesario reiniciar el equipo para que el nuevo Kernel entre en operación, ya que en el momento todavía se encuentra corriendo el anterior como lo muestra la Figura 18.



```
ffuertes@fenix: ~
zealot:/usr/src# uname -r
2.6.26-2-686
zealot:/usr/src#
```

Figura 18. Verificación de la versión actual del Kernel antes del reinicio

Una vez el equipo reinicia se revisa la versión del Kernel mostrando que está corriendo la versión endurecida (ver Figura 19).

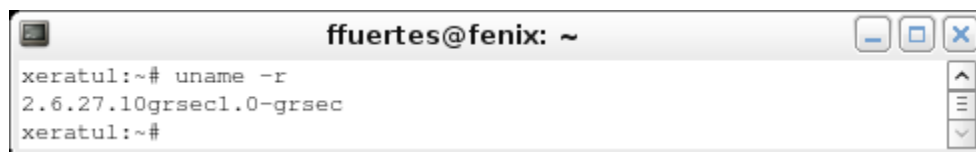


```
ffuertes@fenix: ~  
zealot:~# uname -r  
2.6.27.10grsecl.0-grsec  
zealot:~#
```

Figura 19. Verificación de la versión actual del Kernel después del reinicio

Como se mencionó al iniciar este documento, el hecho de que se obtiene un paquete instalable (.deb) después de la compilación, hace que no sea necesario volver a realizar todo el anterior proceso en cada host del prototipo, ya que simplemente se copia este archivo a los demás equipos y se instala como se mostró en el punto 5.

Las figuras 20 a 22 muestran la verificación de la instalación de este Kernel en los demás equipos del prototipo.




```
ffuertes@fenix: ~  
xeratul:~# uname -r  
2.6.27.10grsecl.0-grsec  
xeratul:~#
```

Figura 20. Verificación de la versión actual del Kernel en el host xeratul



```
ffuertes@fenix: ~  
dragoon:~# uname -r  
2.6.27.10grsecl.0-grsec  
dragoon:~#
```

Figura 21. Verificación de la versión actual del Kernel en el host dragoon



```
ffuertes@fenix: ~  
fenix:~# uname -r  
2.6.27.10grsecl.0-grsec  
fenix:~#
```

Figura 22. Verificación de la versión actual del Kernel en el host fenix