By EvilAngel
Published: 2008-11-17 16:58

Hardening The Linux Kernel With Grsecurity (Debian)

Security is based on three characteristics: prevention, protection and detection. Grsecurity is a patch for Linux kernel that allows you to increase each of these points.

This howto was performed on a Debian Lenny system. Thus some tools are Debian specific. However, tasks can be performed with other distro specific tools or even with universal tools (make).

Everything will done with root privileges. However, you can perform them with a limited account thanks to sudo and fake-kpkg tools.

1. Preliminary Note

To compile the kernel, you need to install some specific packages:

```
rom1:/root# aptitude install patch bin86 kernel-package build-essential
```

If you like to configure your kernel in graphical console mode (make menuconfig), you must install one more package:

```
rom1:/root# aptitude install libncurses5-dev
```

Check that iniramfs-tools (used to generated the init ramdisk) is installed (it should be):

```
rom1:/usr/src# dpkg -l initramfs*
```

```
Desired=Unknown/Install/Remove/Purge/Hold
```

```
| Status=Not/Installed/Config-files/Unpacked/Failed-config/Half-installed
|/ Err?=(none)/Hold/Reinst-required/X=both-problems (Status,Err: uppercase=bad)
||/ Name           Version        Description
+++-==============-==============-============================================
ii  initramfs-tool 0.85i          tools for generating an initramfs
rom1:/usr/src#
```

Go to the source folder:

```
rom1:/root# cd /usr/src
```

Download the grsecurity patch and the

2.6.24.5

Linux vanilla kernel:

```
rom1:/usr/src# wget grsecurity.net/grsecurity-2.1.11-2.6.24.5-200804211829.patch.gz
```

```
rom1:/usr/src# wget eu.kernel.org/pub/linux/kernel/v2.6/linux-2.6.24.5.tar.gz
```

NB: you may need to configure wget in case you are using an HTTP proxy (which may use authentication). You need to edit `/root/.wgetrc` so it looks like this:

```
http_proxy=192.168.0.1
proxy-user=foo # Put this line if you need to authenticate against your proxy
proxy-passwd=bar # Put this line if you need to authenticate against your proxy
```

Decompress the archive of the kernel:

```
rom1:/usr/src# tar xzvf linux-2.6.24.5.tar.gz
```

Create a symbolic link on the new kernel folder to ease the following tasks:

```
rom1:/usr/src# ln -s linux-2.6.24.5 linux
```

Now, the environment is ready. Let's go hardening!

2. Patch the vanilla kernel

Move the grsecurity patch to the new directory:

```
rom1:/usr/src# mv grsecurity-2.1.11-2.6.24.5-200804211829.patch.gz linux/grsecurity-2.1.11-2.6.24.5-200804211829.patch.gz
```

Decompress and patch the source of the kernel:

```
rom1:/usr/src# cd linux
```

```
rom1:/usr/src/linux# gunzip < grsecurity-2.1.11-2.6.24.5-200804211829.patch.gz | patch -p1
```

Now the patch is applied and the source of the kernel was modified. Let's configure the kernel to enable Grsecurity.

3. Configure the hardened kernel

In this example, we will configure the kernel using a console menu (make menuconfig). This is why we installed the `libncurses5-dev` package. However, you can configure in pure console mode (make config), or in GUI mode (make xconfig).

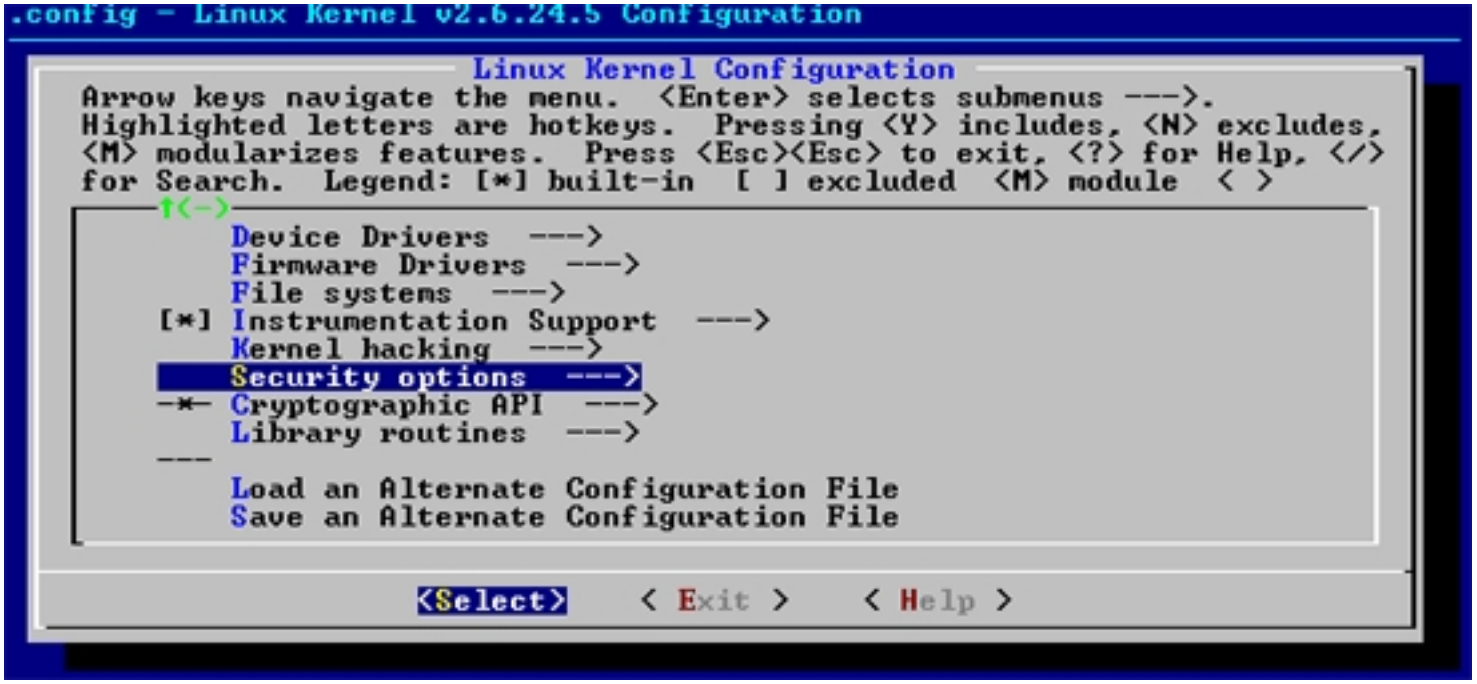
Grsecurity has predefined levels: low, medium, high. It can also be configured in custom level where you choose to enable or not option by option. See

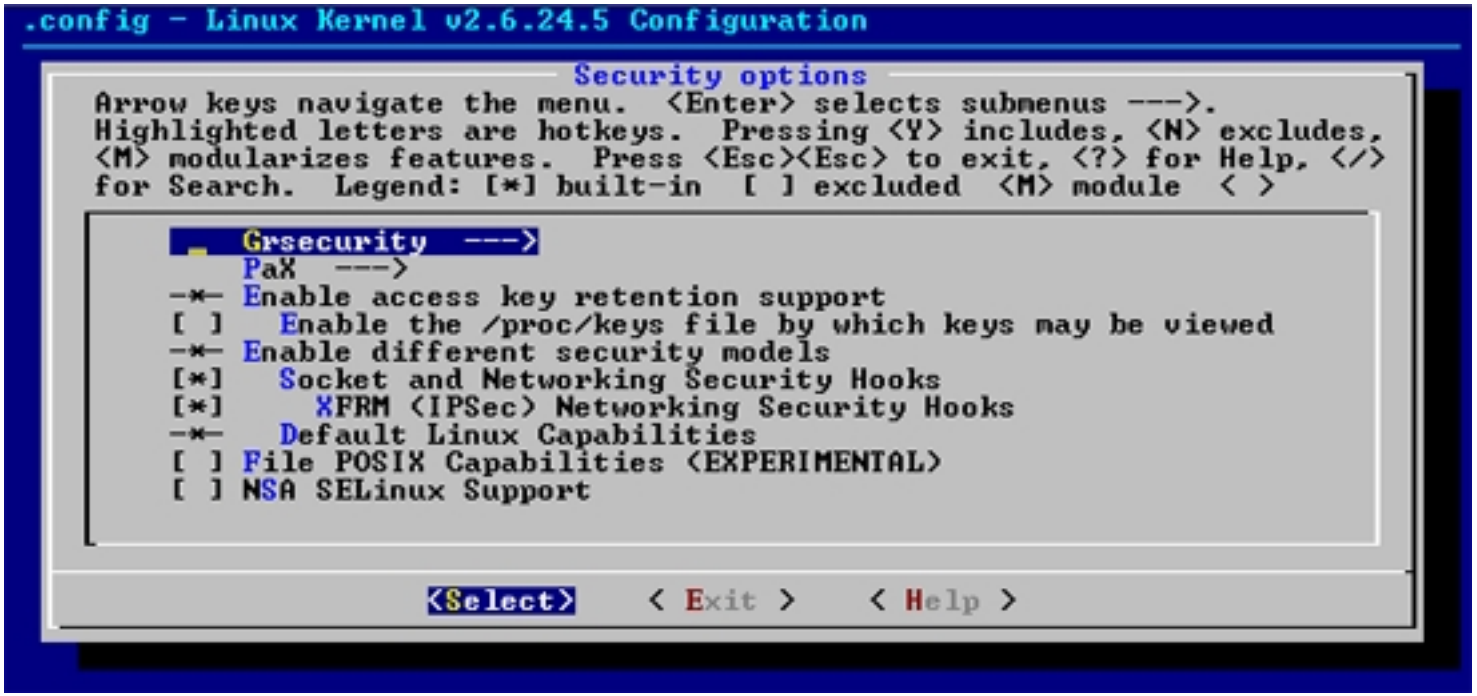
http://www.grsecurity.net/confighelp.php/ for more info on each option. In this HowTo, we will configure Grsecurity in High level.

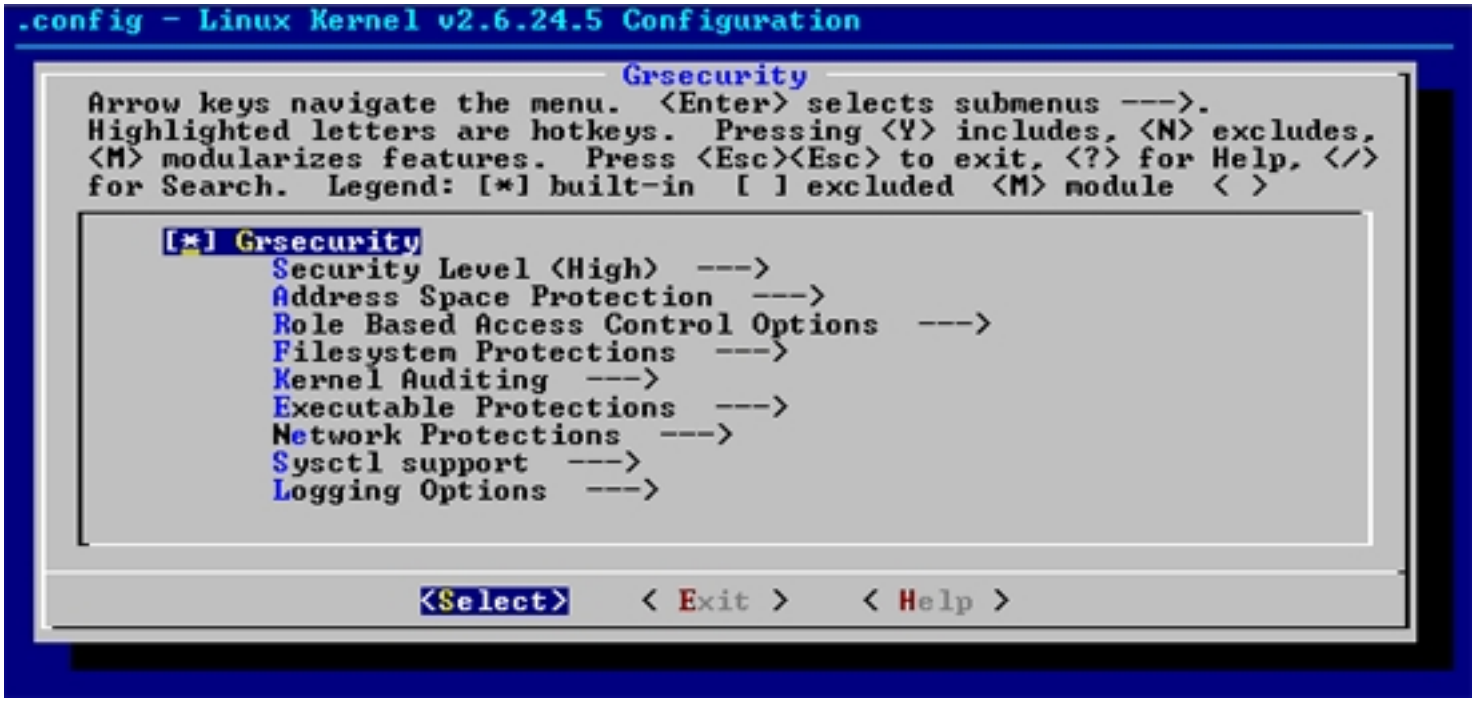
```
rom1:/usr/src/linux# make menuconfig
```

Now, we will enable Grsecurity in the menu.

Go to *Security options* > *Grsecurity* > tick *Grsecurity*. Then, you can go to *Security Level* and tick *High*.







You can profit from configuring Grsecurity to optimise your kernel. Eg: On your server you probably don't need support for infrared, blutooth, probably neither wifi, ipx, X25, token ring, ATM, firewire, PCcard, joystick, mouse, sound....

4. Compile the hardened kernel

It is now time to compile your hardened kernel. First, just in case, clean up:

```
rom1:/usr/src/linux# make-kpkg clean
```

Launch compilation itself (this may take a while depending on your CPU power and RAM availability!!!):

```
rom1:/usr/src/linux# make-kpkg --initrd --append-to-version "grsec1.0" kernel_image
```

In case you are not using a Debian distro, you can compile the classic way with:

```
make mrproper

make menuconfig

make clean

make

make modules_install

mkinitramfs

make install
```

5. Install the hardened kernel

Your new kernel is now compiled and a .deb package file has been generated in the `/usr/src` folder. You need to install your kernel as any .deb package:

```
rom1:/usr/src# dpkg -i linux-image-2.6.24.5-grsec_grsec1.0_i386.deb
```

During the installation, an initrd image will be generated. This may take a while depending on your CPU power and RAM availability! You may also check that the new kernel image is really a kernel !

```
rom1:/usr/src# file vmlinuz-2.6.24.5-grsec

vmlinuz-2.6.24.5-grsec: Linux kernel x86 boot executable RO-rootFS, root_dev 0x801, swap_dev 0x1, Normal VGA
```

It is now time to restart your system with your new hardened kernel:

```
rom1:/usr/src/linux# shutdown -r now
```

Now that your system has restarted, you can check that your new kernel is running:

```
rom1:~# uname -r
```

```
2.6.24.5-grsec
```

6. Testing the hardened kernel

Except the fact that `uname -r` is saying your kernel is a grsec one, how do you know you are running a hardened kernel ? This is where we will use `paxtest` which will simulate an attack on the kernel and show if you are vulnerable or not. Download paxtest:

```
rom1:/tmp# wget http://www.grsecurity.net/~paxguy1/paxtest-0.9.7-pre5.tar.gz
```

Extract it:

```
rom1:/tmp# tar xzvf paxtest-0.9.7-pre5.tar.gz

rom1:/tmp# cd paxtest-0.9.7-pre5
```

Compile it (type `make` to have the list of targets):

```
rom1:/tmp/paxtest-0.9.7-pre5# make generic
```

Run it (there are 2 differents modes: kiddie and blackhat):

```
rom1:/tmp/paxtest-0.9.7-pre5# ./paxtest kiddie
```

NB: unless you are using high grsecurity level or custom level, you will have a vulnerable kernel. Indeed, you are only getting userland ASLR protection in a medium mode.

7. Links

- Grsecurity home site: **http://www.grsecurity.net**
- Linux kernel site: **http://www.kernel.org**
- Wikipedia article on ASLR: **http://en.wikipedia.org/wiki/Address_space_layout_randomization**