## NAME

kernelpkg − a tool for compiling and storing the linux kernel into a Slackware compliant package

## SYNOPSIS

**kernelpkg** [ **−B|−−bootloader lilo|grub|yaboot|elilo|detect** ] [ **−c|−−without-clean** ] [ **−D|−−disable-bootloader** ] [ **−h|−−also-headers** ] [ **−s|−−also-source** ] [ **−I|−−initrd MOST|AUTO|ALL|mod1:mod2:mod3[...]** ] [ **−R|−−rc.modules your_custom_file** ] [ **−−help** ] [ **−−version** ]

## DESCRIPTION

kernelpkg compiles a Linux kernel source tree (2.4.x or 2.6.x) linked on /usr/src/linux and make the slackware's package (tgz) naming it "kernel−image". This file includes: kernel image (bzImage), kernel modules and the rc.modules script. Moreover kernelpkg can add new entry on lilo. Optionally kernelpkg create also a slackware package containing kernel source naming it "kernel−source" and other package that includes kernel headers naming it "kernel−headers". This last, optional, file includes "*.h" files required to compile programs. Packages has architecture fields that do not represent sub-architecture like is usually done in the official Slackware's tgzs.

## ADVANTAGES

The advantages in comparison to a classic kernel compilation are:

1) creation of .tgz slackware packages in order to obtain an orderly system, without "wandering" files
2) more comfortable automatic configuration of bootloader
3) portable packages: portability allows to compile an optimized kernel in order to be used on obsolete pc's, using a more recent pc with greater computational power, saving a lot of time.
4) less commands to be prompted ;) for lazy and listless...

## OPTIONS

kernelpkg have few options because "Keep It Simple and Small"

**−B**, **−−bootloader** your_type
: This is a switch that lets you choose which bootloader to configure on the pc where the kernel-image package will be installed; there are five choice: lilo, grub, yaboot, elilo or detect; lilo is default selection. The tag "detect" is a special feature that automatically finds your bootloader between the supported types. This must be installed on MBR (Master Boot Record) of disk drive.

**−c**, **−−without-clean**
: compiling the kernel without cleaning the source tree before starting the process.

**−D**, **−−disable-bootloader**
: disable bootloader upgrade during installation of the kernel-image package.

**−h**, **−−also-headers**
: create also the kernel-headers package containing all the headers of the compiled kernel.

**−s**, **−−also-source**
: create also the kernel-source package of the compiled kernel. This differs from the tarball made by kernel.org. You need to install this package or you will not be able to compile third party kernel modules (outside of the kernel source tree).

**−I**, **−−initrd** MOST|AUTO|ALL|mod1:mod2:mod3[...]
: create initrd image. This option is useful in situations where you require an initrd image to boot your computer. For instance, when booting a kernel that does not have support for your storage or root filesystem built in. There are special tags: MOST includes most modules for boot, ALL includes all modules and AUTO generate an initrd image containing enough driver support to boot the computer. All special tags take into account the use of LVM/LUKS/RAID. Instead if you need specific modules in the initrd image you can pass to the script a colon (:) delimited list of kernel modules to load. Any dependencies of requested modules will be added to the initrd. For normal cases, initrd is not recommended! Read follow notes very well.

**−R**, **−−rc.modules** your_custom_file
specific your custom rc.modules to insert on kernel-image package.

**−−help** display help and exit.

**−−version**
print out version of kernelpkg software.

## OUTPUT

packages are saved in /usr/src directory

## RESOURCES

*/etc/kernelpkg/rc.modules−default*
this file is contained in kernel-image package. You can link, edit or substitute it as you prefer.

*/etc/kernelpkg/kernelpkg.conf*
the file define default values of options. If this is used in conjunction with any other options passed on the command line, the command-line options will override the config file options.

*/var/log/kernelpkg/**
in this directory are contained the compilation log files.

## EXAMPLES

The advice is leave from Pat's original config, run "make oldconfig" as first and use "make menuconfig" for detail settings. This is locate on "ftp//ftp.slackware.com/pub/slackware/" in source sub-directory "slackware-current/source/k/" or on directory "source/k" of cd/dvd slackware.

```
$ finger @finger.kernel.org
# cd /tmp
# wget -c \
> http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.24.7.tar.bz2
# tar -xvjf linux-2.6.24.7.tar.bz2
# ln -s /tmp/linux-2.6.24.7 linux
# cd /usr/src/linux
# echo \
# "remember built-in chip controller ide/sata/scsi \
# and root filesystem is better."
# make menuconfig
# kernelpkg --also-headers --also-source --bootloader detect
# ls /usr/src/*.tgz
# installpkg /usr/src/kernel-image-2.6.24.7-x86-kpkg091.tgz
# installpkg /usr/src/kernel-headers-2.6.24.7-x86-kpkg091.tgz
# installpkg /usr/src/kernel-source-2.6.24.7-noarch-kpkg091.tgz
```

## NOTES

− Before launching kernelpkg, you must link the source tree of the Linux kernel to /usr/src/linux and have them already configured. The config file that kernelpkg use is /usr/src/linux/.config.

− Better NOT to use ever upgradepkg because will update the kernel installed to the new version, while instead is always better to maintain also the old kernel.

− Kernelpkg support building of the initrd image. For initrd question there are follow choices:
1) In general, INITRD IS NOT RECOMMENDED! For this case you must configure as built−in in your kernel's config:
1a) root's filesystem type
# mount | grep −w / | cut −d " " −f 5
1b) chip controller ide/sata/scsi of your root's partition hard drive
# lspci | grep IDE; lspci | grep SATA; lspci | grep SCSI
2) you can use "--initrd" option. For example, root filesystem type is ext3 and the hard drive is on a ide controller of motherboard "asus p4s8x" with sis5513 chip:
# kernelpkg --also-headers --also-source --bootloader detect \

> --initrd sis5513:pata_sis:ext3
Special tags you avoid specify the exact list of modules. The modules included in the initrd image however remain loaded even after the complete boot, therefore is not recommended to use the tag MOST and especially ALL. The special tag AUTO is more precise and incorporating only the modules necessary for the initrd image; but the way-detection work properly only if the package kernel-image will be installed on a kernel that already uses an initrd with all the necessary modules to boot. In conclusion the best way is absolutely not to use initrd. If you want to use initrd (for example with LVM) you prefer specify modules to be inserted in initrd image or used MOST if you want to be sure that the kernel boots. For all, the LUKS/LVM/RAID will to be detected automatically.
3) you can create manually with mkinitrd command, after kernel−image*.tgz installation and add section on your bootloader's configuration file. For more info read "man mkinitrd", "mkinird --help" or this file "/usr/doc/mkinitrd-x.y.z/README.initrd".

− kernelpkg finds CONFIG_LOCALVERSION automatically; support modified EXTRAVERSION, for example:
# sed -i 's/^EXTRAVERSION = .5/EXTRAVERSION = .5−string/g' \
> /usr/src/linux/Makefile

## SEE ALSO

The additional documentation is in /usr/doc/kernelpkg-x.y.z.
A good guide for the classic compilation <http://www.digitalhermit.com/linux/Kernel-Build-HOWTO.html>
An advanced guide for classic compilation of linux kernel <http://www.thomashertweck.de/kernel.html>

## FAQS

Q: Kernelpkg works like the command "make targz-pkg" (meaning that you create a file .tar.gz and then you rename it "kernel-image-$version-$arch-1.tgz"), therefore with that is possible to obtain tgz package and doing so kernelpkg is useless?

A: "make targz-pkg" creates a tar.gz archive in the directory root of kernel source. This archive is NOT a Slackware compliant package even if it is the same type of archive. Moreover it is not an automated procedure and it does not include the features of kernelpkg. My purpose is to have something similar to "make-kpkg" of debian.

Q: Why did you do a port of "make-kpkg" on Slackware?

A: The "make-kpkg" tool is very complicated! I wanted something more basic and "Slackware-style", so I wrote all the code in shell scripting language like every other Slackware tool.

Q: Kernelpkg support new libata include from Linux kernel 2.6.19?

A: Yes, bootloder update support this feature.

Q: This tool runs on other unofficial ports of Slackware (slamd64, slackintosh) too?

A: Yes, it is architecture indipendent.

Q: Why did this tool create kernel-doc package like other distro?

A: My tool do not create kernel-doc because is a part of vanilla source and for this reason, i do not want to risk broken official package.

Q: It can happen to compile the same version of the kernel several times, to refine the configuration, there is a way to distinguish the various revision?

A: If it want to install more versions of the same version of the kernel linux (for example with different config) uses the LOCAL_VERSION or EXTRAVERSION, which the tool detect automatically.

## EXIT STATUS

0 - success
10 - make error
20 - wrong options

25 - config file lost
26 - config format is wrong
28 - initrd modules list format is wrong
29 - your custom rc.modules doesn't exists
30 - default rc.modules lost
35 - no headers for compile
40 - link to sources doesn't exists
50 - kernel sources doesn't exists
60 - file .config doesn't exist
70 - wrong version of linux (only 2.4/2.6 is supported)
80 - localversion too long
90 - bootloader not supported

## BUGS

There were a few bugs, but my cat ate all of them.
Report bugs to <submax@tiscalinet.it>.

## AUTHOR

Written by Massimo Cavalleri

## COPYRIGHT