Linux para Ingeniería:

Personalización/Compilación del Nucleo

Luis Garreta
luis.garreta@javerianacali.edu.co

Ingeniería de Sistemas y Computación
Pontificia Universidad Javeriana – Cali

16 de marzo de 2018

How to Build Compile and Install Latest Linux Kernel Source

On a Debian / Ubuntu Linux (16.x)

Questions:

- How do I download, compile and install (build) the latest version of the Linux kernel ?
- How do I build and install a custom Linux kernel ?

Answers:

- In order to create a custom kernel configuration file and build a custom kernel, the full Linux kernel source tree must first be downloaded and installed.
- The latest Linux kernel stable version is 4.15.8.
- You will learn how to compile the Linux kernel version 4.15.8 on a Ubuntu Linux (Debian) operating system and build **.deb** file.

Some benefits of build a custom Linux kernel

- Modules contain most of the functionality of the Linux kernel:
  - They can be dynamically loaded/unloaded from the kernel as necessary.
- Remove unwanted drivers from the kernel.
- Faster boot time due to small kernel size.
- Increased security due to additional or removed modules/drivers/features.
- You will learn about the kernel and advanced usage.
- Always run the cutting edge latest kernel.
- Lower memory usage.

Prerequisites

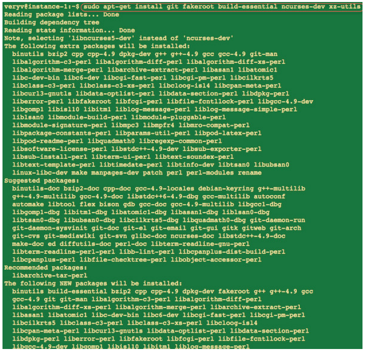
You need to install the following packages on a Debian or Ubuntu Linux to compiler the Linux kernel:

git:	Fast, scalable, distributed revision control system. You can grab the latest source code using the git command.
fakeroot:	Tool for simulating superuser privileges. Useful to build .deb files.
build-essential:	Tools for building the Linux kernel such as GCC compiler and related tools on a Debian or Ubuntu Linux based system.
ncurses-dev:	Developer's libraries for ncurses. This is used by menuconfig while configuring the kernel options.
kernel-package:	Utility for building Linux kernel related Debian packages.
xz-utils:	XZ-format compression utilities to decompress the Linux kernel tar ball.
Disk space:	10 GB or more free disk space.
Time:	Kernel compilation may take quite a while, depending on the power of your machine.

Install required tools:

- Open the terminal application.
- Type the following apt-get command to install the required packages for building the Linux kernel:

```
$ sudo apt-get install git fakeroot build-essential ncurses-dev xz-utils
     libssl-dev bc
```



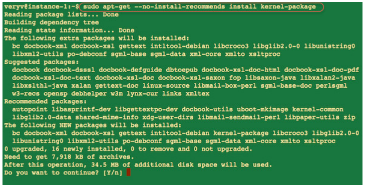
```
veryv@instance-1:~$ sudo apt-get install git fakeroot build-essential ncurses-dev xz-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'libncurses5-dev' instead of 'ncurses-dev'
The following extra packages will be installed:
  binutils bzip2 cpp cpp-4.9 dpkg-dev g++ g++-4.9 gcc gcc-4.9 git-man
  libalgorithm-c3-perl libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libarchive-extract-perl libasan1 libatomic1
  libc-dev-bin libc6-dev libcgi-fast-perl libcgi-pm-perl libcilkrts5
  libclass-c3-perl libclass-c3-xs-perl libcloog-isl4 libcpan-meta-perl
  libcurl3-gnutls libdata-optlist-perl libdata-section-perl libdpkg-perl
  liberror-perl libfakeroot libfcgi-perl libfile-fcntllock-perl libgcc-4.9-dev
  libgomp1 libisl10 libitm1 liblog-message-perl liblog-message-simple-perl
  liblsan0 libmodule-build-perl libmodule-pluggable-perl
  libmodule-signature-perl libmpc3 libmpfr4 libmro-compat-perl
  libpackage-constants-perl libparams-util-perl libpod-latex-perl
  libpod-readme-perl libquadmath0 libregexp-common-perl
  libsoftware-license-perl libstdc++-4.9-dev libsub-exporter-perl
  libsub-install-perl libterm-ui-perl libtext-soundex-perl
  libtext-template-perl libtimedate-perl libtinfo-dev libtsan0 libubsan0
  linux-libc-dev make manpages-dev patch perl perl-modules rename
Suggested packages:
  binutils-doc bzip2-doc cpp-doc gcc-4.9-locales debian-keyring g++-multilib
  g++-4.9-multilib gcc-4.9-doc libstdc++6-4.9-dbg gcc-multilib autoconf
  automake libtool flex bison gdb gcc-doc gcc-4.9-multilib libgcc1-dbg
  libgomp1-dbg libitm1-dbg libatomic1-dbg libasan1-dbg liblsan0-dbg
  libtsan0-dbg libubsan0-dbg libcilkrts5-dbg libquadmath0-dbg git-daemon-run
  git-daemon-sysvinit git-doc git-el git-email git-gui gitk gitweb git-arch
  git-cvs git-mediawiki git-svn glibc-doc ncurses-doc libstdc++-4.9-doc
  make-doc ed diffutils-doc perl-doc libterm-readline-gnu-perl
  libterm-readline-perl-perl libb-lint-perl libcpanplus-dist-build-perl
  libcpanplus-perl libfile-checktree-perl libobject-accessor-perl
Recommended packages:
  libarchive-tar-perl
The following NEW packages will be installed:
  binutils build-essential bzip2 cpp cpp-4.9 dpkg-dev fakeroot g++ g++-4.9 gcc
  gcc-4.9 git git-man libalgorithm-c3-perl libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libarchive-extract-perl
  libasan1 libatomic1 libc-dev-bin libc6-dev libcgi-fast-perl libcgi-pm-perl
  libcilkrts5 libclass-c3-perl libclass-c3-xs-perl libcloog-isl4
  libcpan-meta-perl libcurl3-gnutls libdata-optlist-perl libdata-section-perl
  libdpkg-perl liberror-perl libfakeroot libfcgi-perl libfile-fcntllock-perl
  libgcc-4.9-dev libgomp1 libisl10 libitm1 liblog-message-perl
```

Install required building tools: kernel-package

Utility for building Linux kernel related Debian packages

Finally, install the kernel-package package too:

```
$ sudo apt-get install kernel-package
```

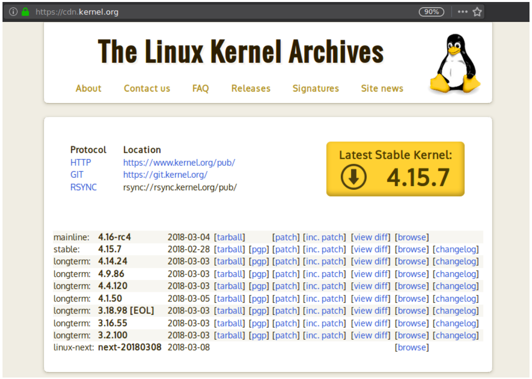


```
veryv@instance-1:~$ sudo apt-get --no-install-recommends install kernel-package
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  bc docbook-xml docbook-xsl gettext intltool-debian libcroco3 libglib2.0-0 libunistring0
  libxml2-utils po-debconf sgml-base sgml-data xml-core xmlto xsltproc
Suggested packages:
  docbook docbook-dsssl docbook-defguide dbtoepub docbook-xsl-doc-html docbook-xsl-doc-pdf
  docbook-xsl-doc-text docbook-xsl-doc docbook-xsl-saxon fop libsaxon-java libxalan2-java
  libxslthl-java xalan gettext-doc linux-source libmail-box-perl sgml-base-doc perlsgml
  w3-recs opensp debhelper w3m lynx-cur links xmltex
Recommended packages:
  autopoint libasprintf-dev libgettextpo-dev docbook-utils uboot-mkimage kernel-common
  libglib2.0-data shared-mime-info xdg-user-dirs libmail-sendmail-perl libpaper-utils zip
The following NEW packages will be installed:
  bc docbook-xml docbook-xsl gettext intltool-debian kernel-package libcroco3 libglib2.0-0
  libunistring0 libxml2-utils po-debconf sgml-base sgml-data xml-core xmlto xsltproc
0 upgraded, 16 newly installed, 0 to remove and 0 not upgraded.
Need to get 7,918 kB of archives.
After this operation, 34.5 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

The Linux kernel sources archives:

https://cdn.kernel.org/

The Linux Kernel Organization is a California Public Benefit Corporation established in 2002 to distribute the Linux kernel and other Open Source software to the public without charge.



Protocol	Location
HTTP	https://www.kernel.org/pub/
GIT	https://git.kernel.org/
RSYNC	rsync://rsync.kernel.org/pub/

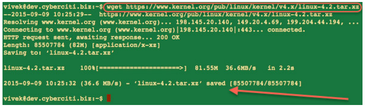
mainline:	4.16-rc4	2018-03-04	[tarball]		[patch]	[inc. patch]	[view diff]	[browse]	
stable:	4.15.7	2018-02-28	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.14.24	2018-03-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.9.86	2018-03-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.4.120	2018-03-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.1.50	2018-03-05	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	3.18.98 [EOL]	2018-03-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	3.16.55	2018-03-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	3.2.100	2018-03-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
linux-next:	next-20180308	2018-03-08						[browse]	

Download the Linux kernel source code

Type the following wget command to grab both source code and pgp keys:

```
$ wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.15.8.tar.sign

$ wget wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.15.8.tar.xz
```



```
vivek@dev.cyberciti.biz:~$ wget https://www.kernel.org/pub/linux/kernel/v4.x/linux-4.2.tar.xz
--2015-09-09 10:25:29--  https://www.kernel.org/pub/linux/kernel/v4.x/linux-4.2.tar.xz
Resolving www.kernel.org (www.kernel.org)... 198.145.20.140, 149.20.4.69, 199.204.44.194, ...
Connecting to www.kernel.org (www.kernel.org)|198.145.20.140|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85507784 (82M) [application/x-xz]
Saving to: 'linux-4.2.tar.xz'

linux-4.2.tar.xz    100%[=====================>]  81.55M  36.6MB/s   in 2.2s

2015-09-09 10:25:32 (36.6 MB/s) - 'linux-4.2.tar.xz' saved [85507784/85507784]

vivek@dev.cyberciti.biz:~$
```

Verify kernel signatures:

Use GnuPG to verify kernel signatures:

```
$ unxz linux-4.15.8.tar.xz
$ gpg --verify linux-4.15.8.tar.sign
```

Output

```
gpg: assuming signed data in ‘linux-4.15.8.tar’
gpg: Signature made Monday 20 February 2017 04:28:37 AM IST using RSA key ID
     6092693E
gpg: Can’t check signature: public key not found
```

Get the public key from the PGP keyserver

Get the public key from the PGP keyserver in order to verify the signature i.e. RSA key ID 6092693E (from the above outputs):

```
gpg --keyserver hkp://keys.gnupg.net --recv-keys 6092693E
```

Output:

```
gpg: requesting key 6092693E from hkp server keys.gnupg.net
gpg: key 6092693E: public key "Greg Kroah-Hartman (Linux kernel stable release
     signing key) <greg@kroah.com>" imported
gpg: key 6092693E: public key "Greg Kroah-Hartman <gregkh@linuxfoundation.org>"
     imported
gpg: no ultimately trusted keys found
gpg: Total number processed: 2
gpg:               imported: 2  (RSA: 2)
```

Now verify again:

Now verify again:

```
gpg --verify linux-4.15.8.tar.sign
```

Output:

```
gpg: assuming signed data in ‘linux-4.15.8.tar’
gpg: Signature made Sat 18 Feb 2017 09:13:06 AM -05 using RSA key ID 6092693E
gpg: Good signature from "Greg Kroah-Hartman <gregkh@linuxfoundation.org>"
gpg:                 aka "Greg Kroah-Hartman <gregkh@kernel.org>"
gpg:                 aka "Greg Kroah-Hartman (Linux kernel stable release
     signing key) <greg@kroah.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 647F 2865 4894 E3BD 4571  99BE 38DB BDC8 6092 693E
```

Untar the Linux kernel

If you do not "BAD signature" output from "gpg –verify" command, untar the Linux kernel tar ball using the tar command enter:

```
$ tar xvf linux-4.15.8.tar
$ ls
$ cd linux-4.15.8/
$ ls
```

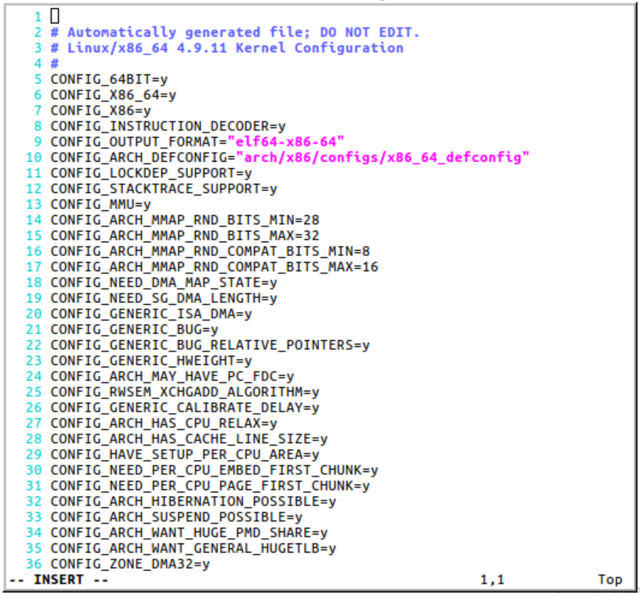
Output:

```
linux-4.15.8/
linux-4.15.8/.cocciconfig
linux-4.15.8/.get_maintainer.ignore
linux-4.15.8/.gitattributes
linux-4.15.8/.gitignore
linux-4.15.8/.mailmap
linux-4.15.8/COPYING
linux-4.15.8/CREDITS
linux-4.15.8/Documentation/
linux-4.15.8/Documentation/.gitignore
linux-4.15.8/Documentation/00-INDEX
linux-4.15.8/Documentation/ABI/
linux-4.15.8/Documentation/ABI/README
linux-4.15.8/Documentation/ABI/obsolete/
linux-4.15.8/Documentation/ABI/obsolete/proc-sys-vm-nr_pdflush_threads
linux-4.15.8/Documentation/ABI/obsolete/sysfs-block-zram
linux-4.15.8/Documentation/ABI/obsolete/sysfs-bus-usb
```

Kernel Configuration

- The Linux kernel configuration file is usually found (debian/ubuntu) in the /boot directory:
  - /boot/config*.
- It is not recommended to edit this file directly but to use one of these configuration options:
  - **make config** - starts a character based questions and answer session
  - **make menuconfig** - starts a terminal-oriented configuration tool (using ncurses)
  - **make xconfig** - starts a X based configuration tool
- Ultimately these configuration tools edit the config* file to allow modifications on it.

A sample of a Kernel Configuration file



```

# Automatically generated file; DO NOT EDIT.
# Linux/x86_64 4.9.11 Kernel Configuration
#
CONFIG_64BIT=y
CONFIG_X86_64=y
CONFIG_X86=y
CONFIG_INSTRUCTION_DECODER=y
CONFIG_OUTPUT_FORMAT="elf64-x86-64"
CONFIG_ARCH_DEFCONFIG="arch/x86/configs/x86_64_defconfig"
CONFIG_LOCKDEP_SUPPORT=y
CONFIG_STACKTRACE_SUPPORT=y
CONFIG_MMU=y
CONFIG_ARCH_MMAP_RND_BITS_MIN=28
CONFIG_ARCH_MMAP_RND_BITS_MAX=32
CONFIG_ARCH_MMAP_RND_COMPAT_BITS_MIN=8
CONFIG_ARCH_MMAP_RND_COMPAT_BITS_MAX=16
CONFIG_NEED_DMA_MAP_STATE=y
CONFIG_NEED_SG_DMA_LENGTH=y
CONFIG_GENERIC_ISA_DMA=y
CONFIG_GENERIC_BUG=y
CONFIG_GENERIC_BUG_RELATIVE_POINTERS=y
CONFIG_GENERIC_HWEIGHT=y
CONFIG_ARCH_MAY_HAVE_PC_FDC=y
CONFIG_RWSEM_XCHGADD_ALGORITHM=y
CONFIG_GENERIC_CALIBRATE_DELAY=y
CONFIG_ARCH_HAS_CPU_RELAX=y
CONFIG_ARCH_HAS_CACHE_LINE_SIZE=y
CONFIG_HAVE_SETUP_PER_CPU_AREA=y
CONFIG_NEED_PER_CPU_EMBED_FIRST_CHUNK=y
CONFIG_NEED_PER_CPU_PAGE_FIRST_CHUNK=y
CONFIG_ARCH_HIBERNATION_POSSIBLE=y
CONFIG_ARCH_SUSPEND_POSSIBLE=y
CONFIG_ARCH_WANT_HUGE_PMD_SHARE=y
CONFIG_ARCH_WANT_GENERAL_HUGETLB=y
CONFIG_ZONE_DMA32=y
-- INSERT --                                    1,1           Top
```

Options in the Kernel Configuration File

- An option will either indicate:
  - some driver is built into the kernel ("=y") or
  - will be built as a module ("=m") or
  - is not selected.
- The unselected state can either be indicated by:
  - a line starting with "#" (e.g. "# CONFIG_SCSI is not set") or
  - by the absence of the relevant line from the .config file.
- Example: The 3 states of the main selection option for the SCSI subsystem:

```
CONFIG_SCSI=y
CONFIG_SCSI=m
# CONFIG_SCSI is not set
```

Configure the Linux kernel

First, copy your existing Linux kernel config file

```
$ cd linux-4.15.8
$ cp -v /boot/config-$(uname -r) .config
```

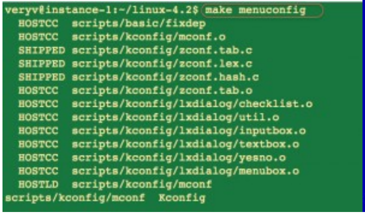
Sample outputs:

```
/boot/config-4.4.0-62-generic' -> '.config'
```

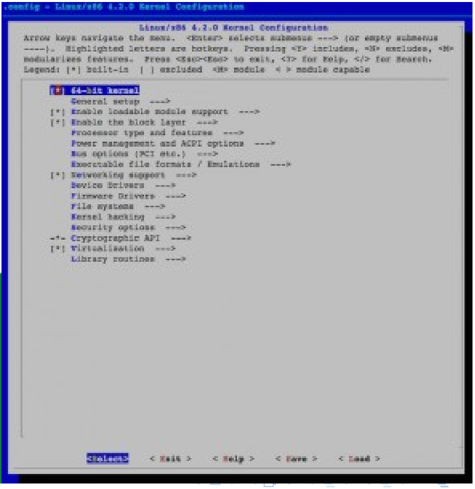
Start Configuring the Kernel

To configure the kernel, run:

```
$ make menuconfig
```



```
veryv@instance-1:~/linux-4.2$ make menuconfig
  HOSTCC  scripts/basic/fixdep
  HOSTCC  scripts/kconfig/mconf.o
  SHIPPED scripts/kconfig/zconf.tab.c
  SHIPPED scripts/kconfig/zconf.lex.c
  SHIPPED scripts/kconfig/zconf.hash.c
  HOSTCC  scripts/kconfig/zconf.tab.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTLD  scripts/kconfig/mconf
scripts/kconfig/mconf  Kconfig
```



```
.config - Linux/x86 4.2.0 Kernel Configuration
                     Linux/x86 4.2.0 Kernel Configuration
 Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus
 ----).  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M>
 modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.
 Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

        [*] 64-bit kernel
            General setup  --->
        [*] Enable loadable module support  --->
        [*] Enable the block layer  --->
            Processor type and features  --->
            Power management and ACPI options  --->
            Bus options (PCI etc.)  --->
            Executable file formats / Emulations  --->
        [*] Networking support  --->
            Device Drivers  --->
            Firmware Drivers  --->
            File systems  --->
            Kernel hacking  --->
            Security options  --->
        -*- Cryptographic API  --->
        [*] Virtualization  --->
            Library routines  --->

          <Select>    < Exit >    < Help >    < Save >    < Load >
```

Linux Commands to Explore your Hardware

1. How to View Linux System Information:
   - **uname**, options (-n, -v, -r, -m, -a)
2. How to View Linux System Hardware Information: **lshw**, options (-short)
3. How to View Linux CPU Information: **lscpu**
4. How to Collect Linux Block Device Information: **lsblk**
5. How to Print USB Controllers Information**: lsusb,** options (-v)
6. How to Print PCI Devices Information: **lspci**, options (-v)
7. How to Print SCSI Devices Information: **lsscsi**
8. How to Print Information about SATA Devices:
   - **hdparm** /dev/sda1
9. How to Print Linux File System Information:
   - sudo **fdisk** -l
10. How to Extract Information about Hardware Components:
    - sudo **dmidecode** -t memory

WARNING

WARNING

- It is easy to remove support for a device driver or option and end up with a broken kernel.
- For example, if the ext4 driver is removed from the kernel configuration file, a system may not boot.
- When in doubt, just leave support in the kernel.

Make sure you save the changes before exit from menuconfig.

Kernel Compilations Commands

fakeroot:

- Run a command in an environment faking root privileges for file manipulation
- This is useful for allowing users to create archives (tar, ar, .deb etc.) with files in them with root permissions/ownership

maker-kpkg:

- build Debian kernel packages from Linux kernel sources, and options are:

–initrd:	Create an initrd image.
–revision=1.0.NAS:	Set custom revision for your kernel such as 1.0.NAS or -1.0-custom-kernel etc.
kernel_image:	This targetz produces a Debian package of the Linux kernel source image, and any modules configured in the kernel configuration file .config.
kernel_headers:	This target produces a Debian package of the Linux kernel header image.

Compile the Linux kernel

Now, you can compile the kernel, run:

- First, make a clean

```
$ make-kpkg clean
```

- Now, you can compile the kernel, run:

```
$ fakeroot make-kpkg --initrd --revision=1.0.NAS kernel_image
     kernel_headers
```

- To speed up the compile process pass the -j option (-j 4 means you are using all 4 cores to compile the Linux kernel):

```
$ fakeroot make-kpkg --initrd --revision=1.0.NAS kernel_image
     kernel_headers -j 4
```

Some Requirements for the Compilation

- Please note that kernel compilation may take quite a while, depending on the power of your machine.
- On my shared 4 CORE CPU and 8GB ram it took 60 mins to build the Linux kernel.
- In the end you should see something as follows on screen:

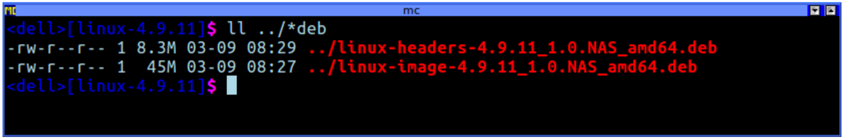
```
test ! -e debian/control~ || rm -f debian/control~
dpkg-gencontrol -isp -DArchitecture=amd64 -plinux-headers-4.15.8 \
                                    -P/tmp/linux-4.15.8/debian/linux
                                        -headers-4.15.8/
dpkg-gencontrol: warning: -isp is deprecated; it is without effect
create_md5sums_fn () { cd $1 ; find . -type f ! -regex './DEBIAN/.*' ! -
    regex './var/.*'     -printf '%P\0' | xargs -r0 md5sum > DEBIAN/
    md5sums ; if [ -z "DEBIAN/md5sums" ] ; then rm -f "DEBIAN/md5sums" ;
    fi ; } ; create_md5sums_fn                /tmp/linux-4.15.8/debian
    /linux-headers-4.15.8
chown -R root:root /tmp/linux-4.15.8/debian/linux-headers-4.15.8
chmod -R og=rX     /tmp/linux-4.15.8/debian/linux-headers-4.15.8
dpkg --build       /tmp/linux-4.15.8/debian/linux-headers-4.15.8 ..
dpkg-deb: building package `linux-headers-4.15.8' in `../linux-headers
    -4.15.8_1.0.NAS_amd64.deb'.
cp -pf debian/control.dist          debian/control
make[2]: Leaving directory '/tmp/linux-4.15.8'
make[1]: Leaving directory '/tmp/linux-4.15.8'
```

Verify kernel deb files:

List the parent directory:

```
$ ls  ../*.deb
```

Output:

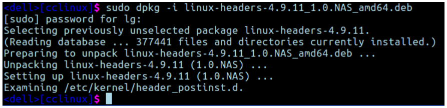


```
<dell>[linux-4.9.11]$ ll ../*deb
-rw-r--r-- 1 8.3M 03-09 08:29 ../linux-headers-4.9.11_1.0.NAS_amd64.deb
-rw-r--r-- 1  45M 03-09 08:27 ../linux-image-4.9.11_1.0.NAS_amd64.deb
<dell>[linux-4.9.11]$
```

Installing a custom kernel

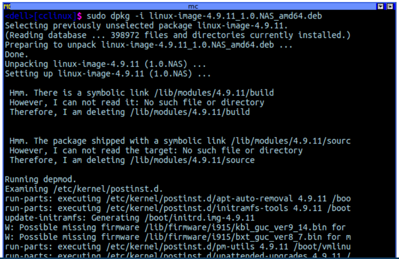
Type the following dpkg command to install a custom kernel on your system:

```
$ cd ..
$ sudo dpkg -i linux-headers-4.15.8_1.0.NAS_amd64.deb
```



```
<dell>[cclinux]$ sudo dpkg -i linux-headers-4.9.11_1.0.NAS_amd64.deb
[sudo] password for lg:
Selecting previously unselected package linux-headers-4.9.11.
(Reading database ... 377441 files and directories currently installed.)
Preparing to unpack linux-headers-4.9.11_1.0.NAS_amd64.deb ...
Unpacking linux-headers-4.9.11 (1.0.NAS) ...
Setting up linux-headers-4.9.11 (1.0.NAS) ...
Examining /etc/kernel/header_postinst.d.
<dell>[cclinux]$
```

```
$ sudo dpkg -i linux-image-4.15.8_1.0.NAS_amd64.deb
```



```
<dell>[cclinux]$ sudo dpkg -i linux-image-4.9.11_1.0.NAS_amd64.deb
Selecting previously unselected package linux-image-4.9.11.
(Reading database ... 398972 files and directories currently installed.)
Preparing to unpack linux-image-4.9.11_1.0.NAS_amd64.deb ...
Done.
Unpacking linux-image-4.9.11 (1.0.NAS) ...
Setting up linux-image-4.9.11 (1.0.NAS) ...

 Hmm. There is a symbolic link /lib/modules/4.9.11/build
 However, I can not read it: No such file or directory
 Therefore, I am deleting /lib/modules/4.9.11/build


 Hmm. The package shipped with a symbolic link /lib/modules/4.9.11/sourc
 However, I can not read the target: No such file or directory
 Therefore, I am deleting /lib/modules/4.9.11/source

Running depmod.
Examining /etc/kernel/postinst.d.
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.9.11 /boo
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.9.11 /boot
update-initramfs: Generating /boot/initrd.img-4.9.11
W: Possible missing firmware /lib/firmware/i915/kbl_guc_ver9_14.bin for
W: Possible missing firmware /lib/firmware/i915/bxt_guc_ver8_7.bin for m
run-parts: executing /etc/kernel/postinst.d/pm-utils 4.9.11 /boot/vmlinu
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 4.9.11 /
```

Reboot the box/server/laptop

Type the following command:

```
$ sudo reboot
```

Verify that everything is working

Type the following command to verify your new kernel and everything is working fine:

```
$ uname -a
$ uname -r
$ uname -mrs
$ dmesg | more
$ dmesg | egrep -i --color 'error|critical|failed'
```

Output:

```
Linux ubuntu-box-1 4.15.8 #1 SMP Mon Feb 20 21:10:55 IST 2017 x86_64 x86_64
     x86_64 GNU/Linux
```