

Comp 7006 - Lab #8

Compiling and Installing a custom Linux Kernel

Objective: To learn how to build a custom kernel.

- A custom kernel can be better for several reasons:
 1. **Speed.** When optimized for your particular hardware, the system will run faster.
 2. **Convenience.** A custom kernel that supports your hardware exactly will be easier to administer because you will not have to manually load modules if **kernel** occasionally fails.
 3. **Memory.** A customized kernel may use slightly less memory.
 4. **Security** – Backdoors, malware, etc

Concepts and Background

- The Linux kernel is released using a dotted triplet notation for the version number, e.g., **4.0.5**.
- The first number indicates the **major** version, the second number indicates the **minor** version, and the third number indicates the **patch** level.
- Even numbered minor versions indicate a stable or production release.
- Odd numbered minor versions indicate development releases.
- It is also a very good idea to append a build number when building kernel, for example **4.0.5-r1**.
- A **monolithic** kernel contains all the device drivers; it is configured for your current architecture. It runs marginally faster at boot time and uses a constant amount of RAM.
- A **modular** kernel loads and unloads drivers as needed and is less secure.
- **CAVEAT:** Do **NOT** delete the existing kernel. Recompile the kernel, giving the new kernel a name that is different from the old kernel's name.
- As part of the compile process, new entries will be added to the **/boot/grub2/grub.cfg** file.
- Then reboot your system.

Basic kernel commands

- The kernel version can be examined with:

uname -a

- To list all loaded modules:

lsmod

- To remove modules (for example the sound module):

rmmod sb

- To insert a module:

insmod sb

Better way to insert a module is:

modprobe sb

Kernel Configuration

- Using one of the build utilities we will first configure the kernel options to be compiled. These utilities must be run from the ***/usr/src/kernels*** directory.
- Whichever utility you choose, you will be offered a large number of kernel options to either select or unselect.
- Kernel features that can be modularized can be specified to be built as modules ("m"), built into the kernel ("y"), or not compiled at all ("n").
- Before configuring and compiling the kernel, ensure that the following packages have been installed:
 - glibc-devel
 - ncurses and ncurses-devel
 - gcc
 - kernel-source

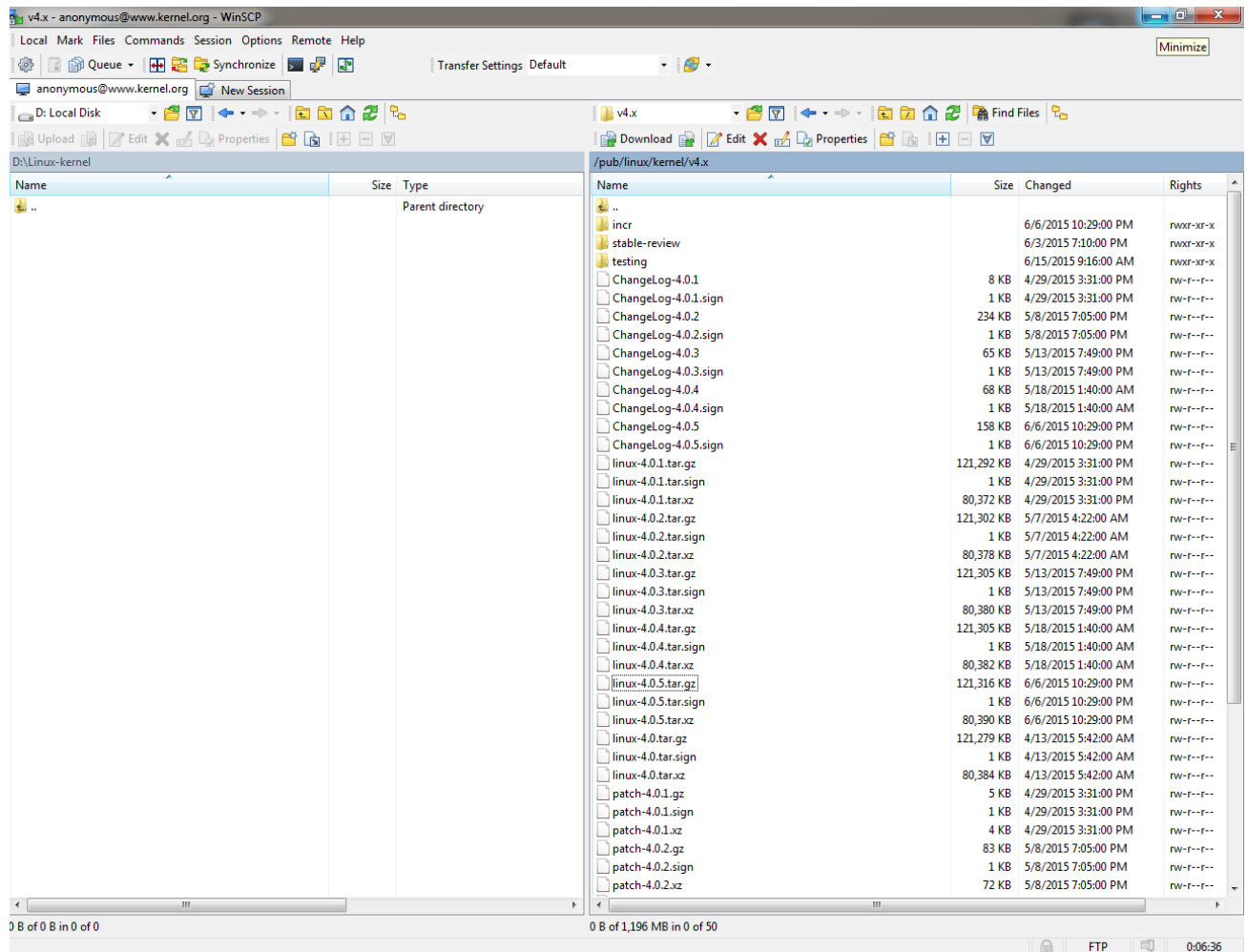
Make Options

- ***make clean***: removes the temporary files from the source tree.
- ***Make***: builds the complete source tree
- ***make modules_install*** compiles the modules and installs them to ***/lib/modules***
- ***make install***: installs all the newly built files and modules into the system directories.

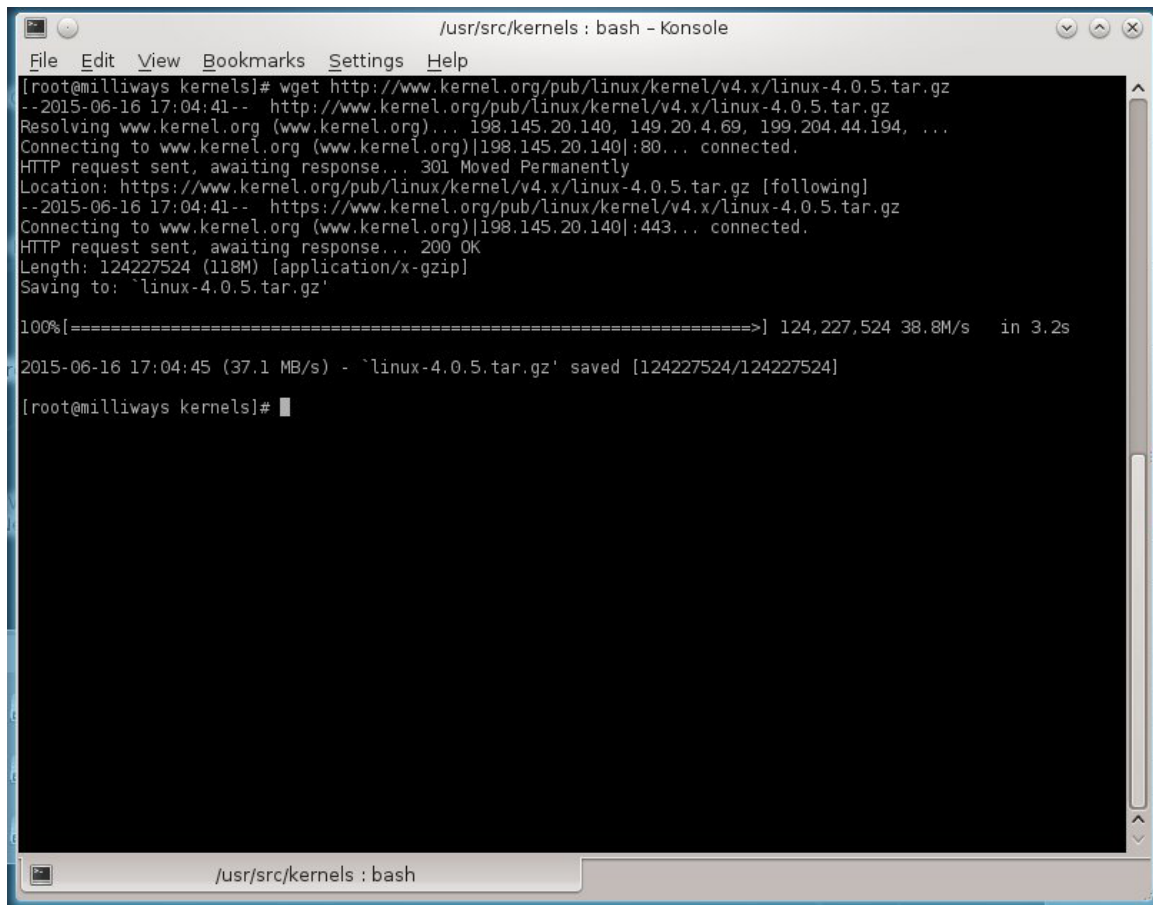
Note: All source files are kept in ***/usr/src/kernels***

Step 0: Download and install the source tree

- There are several ways to download the complete kernel source tree. The following are three examples.
- The easiest way is to download the source package of your choice from: www.kernel.org using a web browser. Download the latest stable release.
- Another way is to use WinSCP (in anonymous ftp mode) to locate and download the latest stable release:



- Alternatively, if you know the exact version you want you can download it using **wget** as follows:



```
/usr/src/kernels : bash - Konsole
File Edit View Bookmarks Settings Help
[root@milliways kernels]# wget http://www.kernel.org/pub/linux/kernel/v4.x/linux-4.0.5.tar.gz
--2015-06-16 17:04:41-- http://www.kernel.org/pub/linux/kernel/v4.x/linux-4.0.5.tar.gz
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]:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://www.kernel.org/pub/linux/kernel/v4.x/linux-4.0.5.tar.gz [following]
--2015-06-16 17:04:41-- https://www.kernel.org/pub/linux/kernel/v4.x/linux-4.0.5.tar.gz
Connecting to www.kernel.org (www.kernel.org)[198.145.20.140]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 124227524 (118M) [application/x-gzip]
Saving to: `linux-4.0.5.tar.gz'

100%[=====>] 124,227,524 38.8M/s in 3.2s

2015-06-16 17:04:45 (37.1 MB/s) - `linux-4.0.5.tar.gz' saved [124227524/124227524]

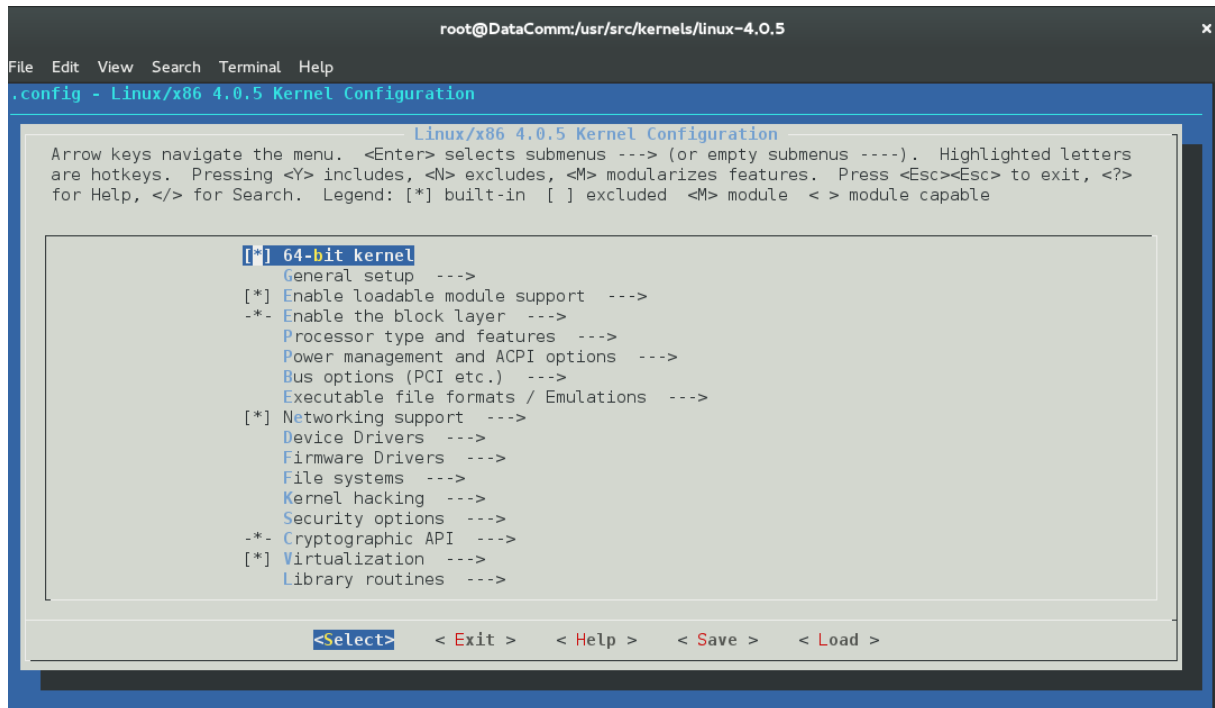
[root@milliways kernels]#
```

Step 1: Configuration (make sure you are in the source directory, for example: /usr/src/kernels/linux-4.0.5):

- Unzip and install the source tree in: /usr/src/kernels
- Use one of the configuration utilities. I prefer "*menuconfig*". Run the following command:

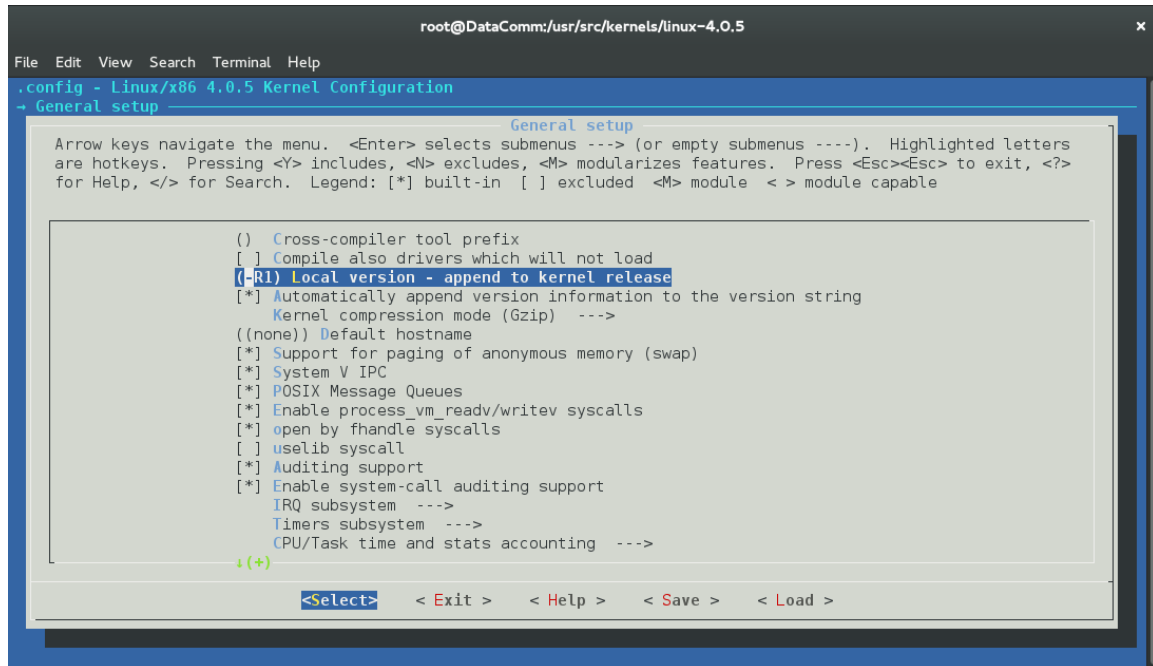
make menuconfig

- You will see a screen similar to the one below:



- Go through all the menu options and configure your kernel.

- For example, under the “**General setup**” you can set the “**local version**” to reflect the new build. For example I might set it to “**-R1**”:



- Then: Save and exit.
- Run: **make clean** (this will get rid of any previous builds under this directory)

Step 2: Compile and Install

- To compile the kernel, run: **make**
- To speed things up (and if you have multiple cores) you can make in multithreaded mode as follows:

make -j 4

- The “-j” switch specifies the number of jobs to run simultaneously. In my case I am fully utilizing the four cores.
- Once make is done you will have to generate all the modules. To create the modules and install them run: **make modules_install**
- Once that is done, install the kernel image:

make install

- The above steps will generate a complete kernel image and install the key files in the boot directory:
 - **System.map-3.0.0**
 - **vmlinuz-3.0.0**
 - **initramfs-3.0.0.img**

Step 5: Updated *grub.cfg*

- As mentioned earlier, the installation process will insert new entries reflecting the fresh builds in the files mentioned above.
- You may make changes to the default kernel boot sequence by editing these files.
- Assume that your current **grub.cfg** (/boot/efi/EFI/fedora/grub.cfg) looks as follows (note that it boots the first kernel, which is specified by the first stanza):

```

root@DataComm:/boot/grub2
File Edit View Search Terminal Help

### BEGIN /etc/grub.d/10_linux ###
menuentry 'Fedora (3.18.7-200.fc21.x86_64) 21 (Twenty One)' --class fedora --class gnu-linux --class gnu --class os $menuentry_
id_option 'gnulinux-3.9.5-301.fc19.x86_64-advanced-a72ad2c8-667f-4fc7-935d-731e951445a9' {
    load_video
    set gfxpayload=keep
    insmod gzio
    insmod part_msdos
    insmod ext2
    set root='hd0,msdos3'
    if [ x$feature_platform_search_hint = xy ]; then
        search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos3 --hint-efi=hd0,msdos3 --hint-baremetal=ahci0,msdos3 --
hint='hd0,msdos3' 1704bcb3-0516-47e0-9c09-0685d275791e
    else
        search --no-floppy --fs-uuid --set=root 1704bcb3-0516-47e0-9c09-0685d275791e
    fi
    linux /vmlinuz-3.18.7-200.fc21.x86_64 root=/dev/mapper/fedora_localhost-root ro rd.md=0 rd.dm=0 vconsole.keymap=us rd
.lvm.lv=fedora_localhost/root rd.luks=0 vconsole.font=latarcyrheb-sun16 rd.lvm.lv=fedora_localhost/swap rhgb quiet LANG=en_US
UTF-8
    initrd /initramfs-3.18.7-200.fc21.x86_64.img
}
menuentry 'Fedora (3.16.3-200.fc20.x86_64) 20 (Heisenbug)' --class fedora --class gnu-linux --class gnu --class os $menuentry_i
d_option 'gnulinux-3.9.5-301.fc19.x86_64-advanced-a72ad2c8-667f-4fc7-935d-731e951445a9' {
    load_video
    set gfxpayload=keep
    insmod gzio
    insmod part_msdos
    :

```

- Here is an example of the new **grub.cfg** containing the freshly compiled kernel with the new stanza inserted:

```

root@DataComm:/boot/grub2
File Edit View Search Terminal Help
### END /etc/grub.d/00_header ###

### BEGIN /etc/grub.d/10_linux ###
menuentry 'Fedora (4.0.5-R1) 21 (Twenty One)' --class fedora --class gnu-linux --class gnu --class os $menuentry_id_option 'gnu
linux-3.9.5-301.fc19.x86_64-advanced-a72ad2c8-667f-4fc7-935d-731e951445a9' {
    load_video
    set gfxpayload=keep
    insmod gzio
    insmod part_msdos
    insmod ext2
    set root='hd0,msdos3'
    if [ x$feature_platform_search_hint = xy ]; then
        search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos3 --hint-efi=hd0,msdos3 --hint-baremetal=ahci0,msdos3 --
hint='hd0,msdos3' 1704bcb3-0516-47e0-9c09-0685d275791e
    else
        search --no-floppy --fs-uuid --set=root 1704bcb3-0516-47e0-9c09-0685d275791e
    fi
    linux /vmlinuz-4.0.5-R1 root=/dev/mapper/fedora_localhost-root ro rd.md=0 rd.dm=0 vconsole.keymap=us rd.lvm.lv=fedora
_localhost/root rd.luks=0 vconsole.font=latarcyrheb-sun16 rd.lvm.lv=fedora_localhost/swap rhgb quiet LANG=en_US.UTF-8
    initrd /initramfs-4.0.5-R1.img
}
menuentry 'Fedora (3.18.7-200.fc21.x86_64) 21 (Twenty One)' --class fedora --class gnu-linux --class gnu --class os $menuentry_
id_option 'gnulinux-3.9.5-301.fc19.x86_64-advanced-a72ad2c8-667f-4fc7-935d-731e951445a9' {
    load_video
    set gfxpayload=keep
    insmod gzio
    insmod part_msdos

```

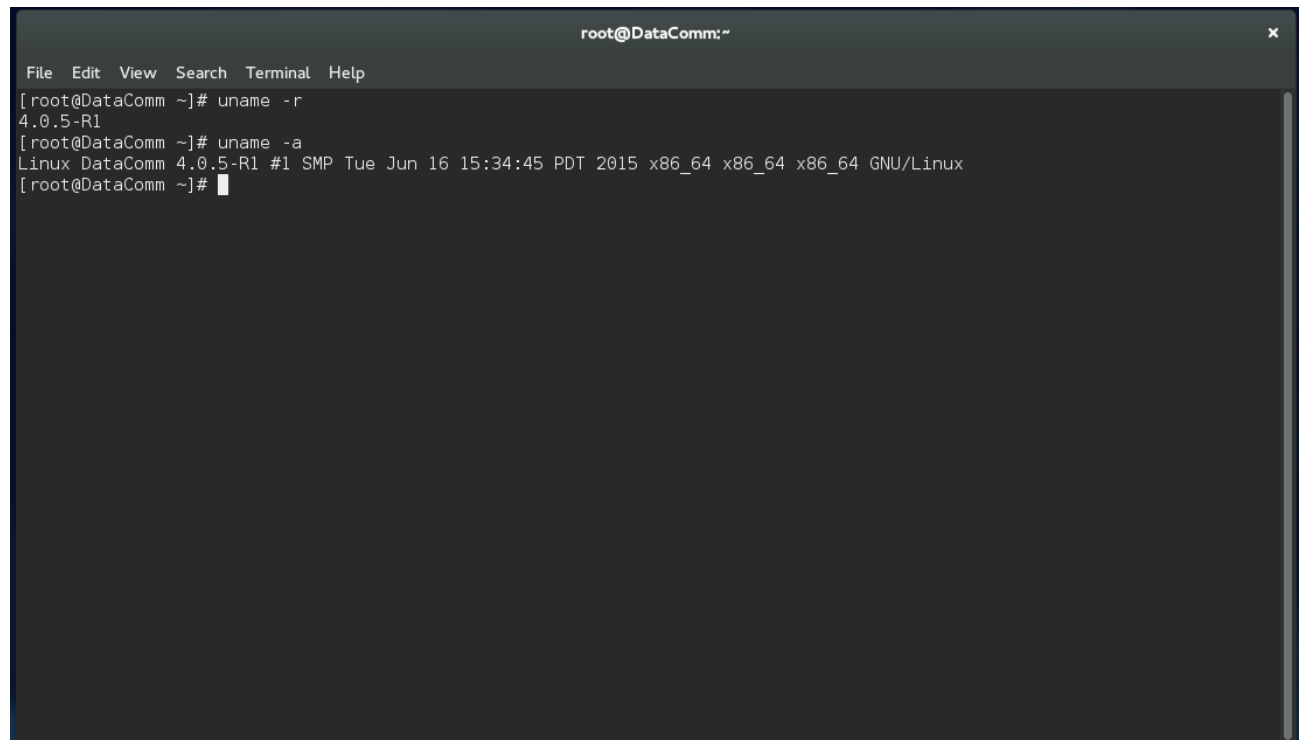
- Open the **grub.cfg** to view the updated configuration. Locate your new menu entry stanza and determine its place in the stanza order.
- To make your new kernel the default, either place its section first or change the default entry number to the appropriate number (remember that it starts counting with 0) in the **/etc/default/grub** file.
- You can change the **GRUB_DEFAULT** option in **/etc/default/grub** to point to the new stanza you added. 0 will refer to the first stanza, 1 to the second, and so on.
- For example:

GRUB_DEFAULT=0

- Alternatively, specify the menu entry title. This is particularly useful if you have a number of menu entries across the various script files.

GRUB_DEFAULT="Fedora Linux"

- From now on, when the system boots you will see all of the available GRUB boot options corresponding to each of the stanzas in **grub.cfg**.
- The default kernel will be highlighted so if you do not press any keys the system will simply boot that one. Alternatively you can scroll to the desired kernel and boot that one.
- Boot up your new kernel and check for the version as follows:

A terminal window titled 'root@DataComm:~' with a menu bar containing 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the output of two 'uname' commands. The first command 'uname -r' returns '4.0.5-R1'. The second command 'uname -a' returns 'Linux DataComm 4.0.5-R1 #1 SMP Tue Jun 16 15:34:45 PDT 2015 x86_64 x86_64 x86_64 GNU/Linux'. The prompt '[root@DataComm ~]#' is shown at the end of each line.

```
root@DataComm:~
File Edit View Search Terminal Help
[root@DataComm ~]# uname -r
4.0.5-R1
[root@DataComm ~]# uname -a
Linux DataComm 4.0.5-R1 #1 SMP Tue Jun 16 15:34:45 PDT 2015 x86_64 x86_64 x86_64 GNU/Linux
[root@DataComm ~]#
```