CPE453 Laboratory Assignment #3
Compiling Linux

Michael Haungs, Spring 2011

1 Objective

In this assignment, you will make a small change to the Linux kernel, compile the kernel, and boot to your newly compiled kernel. All work will be done in a virtual machine (VMware). You will need to know how to compile the kernel to complete future assignments.

2 Resources

The resource links found at the bottom of our course web page will be **very helpful for this lab assignment**. Also, you should read these to familiarize yourself with the kernel compiling process:

- http://www.cyberciti.biz/tips/compiling-linux-kernel-26.html
- http://www.tuxradar.com/content/how-compile-linux-kernel

3 Assignment

For this lab assignment, the most important thing for you to do is follow my instructions ***to the letter***. Nothing can be more frustrating then compiling and installing your kernel incorrectly.

3.1 Preparing your VMWare image

Ok, the first think we need to do is increase the size of your (virtual) hard drive. Right now, it's only 6G and that's mostly used up by applications. Unfortunately, the linux kernel code can grow to several gigabytes after all the ".o" files and final binary are generated. So, for your virtual machine you need to do the following:

1. Completely shutdown your virtual machine. You can either:
   (a) Click on the "Switch" icon in the upper, right hand corner and select "Shut Down..." or
   (b) at the command line type (as root), "shutdown -h now".
2. After your VM shuts down, go the the VMWare "Virtual Machine Library" window and select the Ubuntu image and then click on the "Settings" button. In the settings window,

(a) click the "Hard Disks" icon and then resize the disk to 20GB. (Probably could go as low as 5G and be OK.)

(b) Hit "Apply". (this will take a few minutes to complete).

3. Start your VM and login.

4. In the Ubuntu menu bar, select System->Administration->Disk Utility.

5. In Disk Utility, select the VMware hard disk under "Peripheral Devices" and

   (a) Select the free space under "Volumes" and then click the create partition option to make a new ext4 partition.

      i. Be sure to label the new partition "Home".

6. Now that we have a new partition, we need to mount it:

   (a) First, we need a location in which to mount the new partition To maximize the use of the partition, we are going to mount it on /home, so that our home directories will be on the new partition. However, since we presently using the /home directory for our accounts we are going to have to move them temporarily. Do the following:

      i. Logout of the "student" account and then login as root.
      ii. mv /home/student /tmp
      iii. mv /home/mhaungs /tmp
      iv. Append the line "/dev/sda3 /home ext4 errors=remount-ro 0 1" to the end of the file "/etc/fstab" (mimic the existing format)
      v. type "mount -a"
         A. if you type "df -h" you should see a line like this "/dev/sda3 14.0G 0 13.0G 2% /home".
      vi. mv /tmp/student /home
      vii. mv /tmp/mhaungs /home
      viii. Logout of the "root" account and back into the "student" account.

7. If everything looks correct, you can move on to the next section where we download, modify, compile, and run version 2.6.35 of the linux kernel.

3.2 Preparing the Linux source

The first thing we need to do is make a fresh copy of the kernel source that you can modify and do some preparation for later compilation. The steps below assume you are logged onto your virtual machine using the "student" account. Do the following steps[1] in order:

1. We need to download some additional software to help with compiling the kernel. Type in the following line:

   (a) sudo apt-get install kernel-package fakeroot build-essential ncurses-dev qt3-dev-tools

2. cd; mkdir kernel; cd kernel

3. wget http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.35.9.tar.bz2

4. bunzip2 linux-2.6.35.9.tar.bz2

5. tar xvf linux-2.6.35.9.tar

[1] Adapted from https://wiki.kubuntu.org/KernelTeam/GitKernelBuild

6. mv linux-2.6.35.9 linux-2.6.35.9-lastname (substitute you own last name in place of "lastname")
7. cd linux-2.6.35.9-lastname
8. wget http://www.csc.calpolye.edu/~mhaungs/pub/Haungs453Ubuntu10.10.config
9. cp Haungs453Ubuntu10.10.config .config
10. make oldconfig
11. /*** OPTIONAL STEP **/
    (a) make xconfig
        i. You can change what is compiled into your kernel to reduce its size. However, be careful as you can easily leave out important functionality that results in an unbootable kernel. I spent a looooooong time trying to get ours as small as possible.
12. sed -rie 's/echo "\+"/#echo "\+"/' scripts/setlocalversion (Small change to a kernel config file.)
13. make-kpkg clean
14. CONCURRENCY_LEVEL=`getconf _NPROCESSORS_ONLN` fakeroot make-kpkg –initrd –append-to-version=-lastnam kernel_image kernel_headers (Note: This will take a long, long time. About 45 mins on a slow machine.)
15. cd ..
16. Type the following two commands:
    (a) sudo dpkg -i linux-image-2.6.35.9-lastname_2.6.35.9-lastname-10.00.Custom_i386.deb
    (b) sudo dpkg -i linux-headers-2.6.35.9-lastname_2.6.35.9-lastname-10.00.Custom_i386.deb
17. sudo reboot

At the boot screen, select the "-lastname" kernel. Logon and type "uname -a" after the virtual machine reboots. You should see something like "Linux <machinename> 2.6.35.9-lastname <today's date> <stuff> GNU/Linux". Notice the name of the kernel and the date information.

3.3 Making a small change

Let's make a small change to the kernel. We'll print a customized message after the kernel initializes the processor, console, and memory. Do the following:

1. cd /kernel/linux-2.6.35.9-lastname (Again, substitute your lastname)
2. cd init
3. vi main.c
4. After the calibrate_delay() function in the function start_kernel() function, add the following line: "printk("Chalk dust is lethal!\n");"
5. save and exit vi

3.4 Compiling

The steps you did in Section 3.2 was a one time deal. ***From now on, when you modify the kernel you just need to do the following to compile***:

1. cd ~/kernel/linux-2.6.35.9-lastname
2. CONCURRENCY_LEVEL=‘getconf _NPROCESSORS_ONLN‘ fakeroot make-kpkg –initrd –append-to-version=-lastnam kernel_image kernel_headers (Note: This will not take as long as the first compile)
3. sudo rm -rf /lib/modules/2.6.35.9-lastname
4. cd ..
5. Type the following two commands:
   (a) sudo dpkg -i linux-image-2.6.35.9-lastname_2.6.35.9-lastname-10.00.Custom_i386.deb
   (b) sudo dpkg -i linux-headers-2.6.35.9-lastname_2.6.35.9-lastname-10.00.Custom_i386.deb
6. sudo reboot

Boot to your new custom kernel. Logon and type “uname -a” after the machine reboots to double check that you are running the latest version of your kernel. Type “dmesg | grep Chalk”. You should see your custom message. ***Congratulations!!***

Deliverables

In Lab, on Thursday, April 28, I will have you boot to your modified kernel and check that the custom message was displayed. Lab on that day is MANDATORY in order for you to get a grade for this assignment.