Operating Systems - Fall 2001
Instructor: Craig A. Struble
Compiling the Linux Kernel in 368 Cudahy

Assigned: Tuesday, Oct. 9, 2001 **Due:** None

1 Introduction

This document covers the steps needed to build a kernel on the RedHat 7.1 machines in 368 Cudahy. I've tested this process as best as I can, but some glitches may still exist. Please let me know if you run into problems.

2 Copying the Original Source

The first step you should take is to make a copy of the original linux kernel source. The machines have plenty of disk space, so don't worry about that. The original source is located in the directory named `/usr/src/linux-2.4.2`. You should make a recursive copy of this directory to one for your own kernel, for example `${HOME}/linux` with the commands

```
cp -r /usr/src/linux-2.4.2 ${HOME}/linux
```

I will use the directory `${HOME}/linux` throughout the remainder of this document. Make sure the directory you create does not already exist. This command only need to be executed once.

3 Cleaning Up

The copy of the source tree that you get is not quite ready for use. The next step you should take is to clean up your copy of the source with:

```
cd ${HOME}/linux
make mrproper
```

This will clean up the source directory and prepare it for use. You should only need to execute this once, but if you run into problems compiling the kernel, you might need to start at this step and carry out all subsequent steps.

4 Modifying the Kernel Version

After cleaning the environment, you should modify the `Makefile` and edit the `EXTRAVERSION` variable so that you do not overwrite the original RedHat Kernel. I changed `EXTRAVERSION` to be

```
EXTRAVERSION = -cstruble
```

to denote that this is my kernel. Replace `-cstruble` with an appropriate name for your kernel, such as your group login ID (e.g., `-11grp1` for group 1 from the 11 a.m. section.

4.1 Configuring Your Kernel

The first step is to configure your kernel. There are several options for configuration, which will allow you to build a lean kernel: `make config`, `make menuconfig`, and `make xconfig`. I went the easy route and copied the kernel configuration used by RedHat, which is stored in `${HOME}/linux/configs/kernel-2.4.2-i686.config`. To use this, copy the file to `${HOME}/linux/.config` and then configure the kernel with

```
make oldconfig
```

This command uses the configuration file in `${HOME}/linux/.config` to configure the kernel.

5 Making Source Dependencies

After configuring the kernel, you should make the source dependencies with

```
make dep
```

Both commands do the same thing.

5.1 Compile the Kernel

The next step is to compile the kernel. This is accomplished with the command

```
make bzImage
```

5.2 Building Kernel Modules

The default RedHat kernel configuration configures several drivers as kernel modules: shared object files loadable by the kernel as needed. To build all of these modules, execute the command

```
make modules
```

You only need to do this the first time you build and install your kernel.

5.3 Installing the New Kernel

Once the kernel and kernel modules are built, you should install the kernel and modules in such a way that doesn't overwrite the original Linux kernel. Fortunately, RedHat 7.1 provides nicely set up Makefiles for installation. You need root permission to do this, which you can get by using the `sudo` command. The `sudo` command will ask you for a password. Use **your** password when asked. If you followed the directions above, the commands

```
sudo make install
sudo make modules_install
```

will install a new kernel and modules without overwriting the old ones.

5.4 LILO

Before booting your new system, you need to edit your LILO configuration file, which is stored in `/etc/lilo.conf`. You need to add an entry for your newly created kernel. I assume that you have followed my directions above, in which case the kernel you should boot for the OS project is `/boot/vmlinuz-2.4.2-cstruble`.

My `/etc/lilo.conf` is

```
boot=/dev/hda3
map=/boot/map
install=/boot/boot.b
prompt
linear
default=linux
message=/boot/message

image=/boot/vmlinuz-2.4.2-2
        label=linux
        read-only
        root=/dev/hda3

image=/boot/vmlinuz-2.4.2-cstruble
        label=cstruble
        read-only
        root=/dev/hda3
```

which is set up to boot the original RedHat kernel and the kernel I modified (the second `image` entry). Remember to replace `cstruble` with a name appropriate for your kernel. Also remember to use the `sudo` command to gain root access for editing the file.

The LILO configuration file on the 368 Cudahy machines may look different. My suggestion is to copy the entry for the original Linux kernel and change it to load your modified kernel (don't forget to change the label!). You should only need to edit the file once, unless you make a mistake.

Each time you compile and install a modified kernel, you must run LILO to set up booting:

```
sudo /sbin/lilo
```

5.5 Running the Modified Kernel

To run your modified kernel, simply type in the label you gave it (`cstruble` in my example) at the LILO `boot:` prompt. To run the original kernel, simply type the label you gave the original kernel (`linux` in my example). By setting things up this way, you can recover by switching to the original kernel.