

## Kernel Modules and Configuration

This lab will help you learn about the Linux kernel loadable modules available in a Red Hat Linux distribution. Perform the indicated steps, and write the answers to questions in the appropriate spaces on this hand-out. Turn-in one filled in handout per team by the beginning of class next week. Be sure to include the names of all members of your team.

### Boot the System

Boot the system as usual

Step 1. Boot your system into single or multi-user mode.

### Kernel Modules

Learn where Linux stores kernel modules.

Step 2. List the kernel modules directory. Refer to your lecture notes if necessary to recall the location of the kernel modules. Modules are specific to a kernel version. And there may be more than one version of a kernel on your system, which is selectable at boot time. Use the command:

```
$ uname -r
```

to output the version of the kernel you are running:

**Q1.** Which subdirectories are under the modules directory? Hint: there are more than 3. \_\_\_\_\_

**Q2.** What are each of the various sub-directories under the **modules** directory used for? Spend a few minutes looking through each of the subdirectories. \_\_\_\_\_

\_\_\_\_\_

Step 3. Find the **vfat** module, which is the driver for the Windows FAT filesystem with NT long name support. Where is it located? Hint: modules files end with the **.o** suffix

Step 4. What type of device do you suspect the **cm206** module supports? Use the **find** command to help you discover its location:

```
$ find . -name "*cm206*" -print
```

### Installing Modules

Learn about commands for installing, removing, and listing loaded modules.

Step 5. The **lsmod** command is used to list the currently loaded modules. Try running:

```
$ lsmod
```

**Q3.** Which modules are loaded on your system? \_\_\_\_\_

Step 6. The **insmod** command is used to load a kernel module. Try loading the **dummy** loopback device:

```
# insmod dummy
```

**Q4.** What is the output from **insmod**? \_\_\_\_\_

**Q5.** What command would you run to validate that the module loaded? \_\_\_\_\_

**Q6.** What is the size of the **dummy** module? \_\_\_\_\_

Q7. Is it being *used*? \_\_\_\_\_

Q8. How many *references* are there to it? \_\_\_\_\_

Step 7. Unloading modules is done with the **rmmmod** command. Unload the **dummy** module.

```
# rmmmod dummy
```

Q9. How do you check if the module is unloaded? \_\_\_\_\_

## Modules with Dependencies

*Learn about module inter-dependencies and how to resolve loading errors.*

Step 8. Now load the **msdos** module. You should get several errors about unresolved symbols. This occurs because the **msdos** module, like several other modules, has dependencies on another module. Such modules can only be loaded after their dependencies are satisfied.

Q10. What is the common prefix for each of the unresolved symbols? \_\_\_\_\_

Step 9. It should be fairly obvious by now that the **msdos** module depends on the **fat** module. Go ahead and load the **fat** module first, and then try to load the **msdos** module next. This should work correctly because you have loaded the dependent module.

Step 10. Check to be sure that the **fat** and **msdos** modules are loaded.

Q11. What does the **[msdos]** mean in the output of **lsmod**?

Step 11. Now unload the **fat** module. You should get an error.

Q12. What does the error messages state and what does it mean? \_\_\_\_\_

Q13. How do you correct the problem so that you can unload the **fat** module? \_\_\_\_\_

## Automatically Loaded Modules

*Learn how modules are automatically loaded upon reference*

Step 12. In a previous lab, you created a DOS formatted floppy using the **mkfs** command. This used the **msdos** module; yet you did not have to load this module before you ran **mkfs**. When the kernel is asked to use a driver that is not yet loaded, it asks the kernel module loader to find it. Look at the file **modules.dep** in the modules directory. The **modules.dep** file lists the modules available in the system, and the modules they depend on. Lines in the **modules.dep** file look like:

```
module:  dependee-modules-list
```

where *dependee-modules-list* is a list of modules that *module* depends upon.

Step 13. Look for a line in the **modules.dep** file that lists the module **msdos**.

Q14. What does that module depend on? \_\_\_\_\_

Step 14. The **modprobe** utility is like a smart **insmod** command. It also uses the **modules.dep** file to resolve module dependencies. Use **modprobe** to load the **msdos** module without having to first load the **fat** module:

```
# modprobe msdos
```

Step 15. List the loaded modules to see what was loaded.

Step 16. Unload both the **msdos** and **fat** modules:

```
# rmmmod msdos fat
```

Step 17. This time, use **modprobe** to load the **dos** module:

```
# modprobe dos
```

Step 18. Run **lsmod** to see which modules are loaded.

**Q15.** Is there an **dos** module? If not, what was loaded instead? \_\_\_\_\_

Step 19. To understand what is going on, run the command:

```
$ modprobe -c | grep dos
```

**Q16.** What was the output? \_\_\_\_\_

**Q17.** What is the **-c** option for **modprobe**? \_\_\_\_\_

Step 20. The module loader has an **alias** mechanism that allows you to create another name for a module. Let's create another name for the **dummy** module to see how it works. Edit the file **/etc/modules.conf**, go to the end of the file, and add the line:

```
alias big dummy
```

Then save the file and quit the editor.

Step 21. Unload both the **msdos** and **fat** modules again.

Step 22. Now load the **big** module using **modprobe**. Ignore any messages about one file being more recent than the other. Look at which modules were loaded.

**Q18.** Explain how you know that the alias mechanism worked: \_\_\_\_\_

## Other modules

*Prepare for the networking lab.*

Step 23. Bring the system into single user mode, and then unload all unused modules.

Step 24. In the next lab, we are going to learn about setting up basic networking. To enable networking, you need a driver for the network interface card (NIC). Your machine has a 3Com NIC, model **3c509** or **3c59x**, and if you have a second card, model **3c90x** also. Explore the modules directory to look for the available device driver modules. Find the drivers which seem appropriate for these cards.

**Q19.** Which **.o** driver files seem like candidates? \_\_\_\_\_

Step 25. Try loading each of the modules you believe are correct for your card, one at a time, to see which one loads. The module that loads successfully is probably the correct driver for a given NIC. Which module works for your NIC? Write this module down and save it for next week.

Step 26. Shut down the system and turn off the power after the system has halted.