

Chapter 21

System Upgrade

This chapter introduces the user to various options that might be used to improve the performance of the Linux Kernel.

Concepts Learned in this Chapter

- How to upgrade or modify a Kernel

Table of Contents

System Upgrade.....	1
21.1 Compiling a New Kernel (not complete).....	3
21.1.1 Kernel Configuration.....	3
21.1.1.1 make config	3
21.1.1.2 make menuconfig	3
21.1.1.3 make xconfig	4
21.1.2 Compiling the Kernel	4
21.1.2.1 make dep	4
21.1.2.2 make clean	4
21.1.2.3 make zImage or make bzImage	4
21.1.3 Installing the Kernel.....	4
21.1.3.1 /boot Directory.....	4
21.1.4 Boot Loader.....	5
21.1.4.1 LILO.....	5
21.1.4.2 GRUB.....	5
21.X Commands Used in this Chapter.....	5
21.X Chapter Review Questions.....	5

21.1 Compiling a New Kernel (not complete)

When you installed the Linux system, not only was the binary installed, but the source code was also installed. This is one source of the code; another option is to download it from the Internet.

The two source of code are located at:

www.kernel.org and
/usr/src/linux

For this discussion, we will assume that we wish to refine our existing configuration and will use the existing source code. We will find the kernel source in the /usr/src/linux directory, and the file looks something like **linux-2.4.16.tar.gz**.

The first task will be to unpack the file. This is accomplished by issuing the command:

```
tar -xvzf linux-2.4.16.tar.gz
```

A kernel source tree has now been created.

21.1.1 Kernel Configuration

Now that the kernel source tree exists, we need to create the kernel build process. This level establishes a process by which we can configure kernel modules that we want – and leave out those that are not needed, thus minimizing memory requirements.

We will use one of three commands to configure the Kernel:

```
make config  
make menuconfig  
make xconfig
```

21.1.1.1 make config

The original configuration uses the make config process from the CLI level to establish the Kernel configuration. This is a detailed question and answer process that asks you about every detail – and you should know what you are doing !

Each option is shown by a name and a short description. You are then prompted to answer whether the option is installed, or a module (external option). You may also respond with a question mark for a little longer description. Once you have answered the question, you cannot go back, you must start over in order to modify you answer.

21.1.1.2 make menuconfig

An alternative method to configure the Kernel from the CLI is to utilize the make menuconfig method. This provides the same option as before, but it provides a graphical menu format. More important, you are able to go back and

change previously made options. If you are working from the CLI, then this is the preferred method.

21.1.1.3 make xconfig

An even more user-friendly make version is from the GUI interface. This is the make xconfig. Greater categorization and options for loading or storing a configuration file, and whether to exit or save and exit. Multiple configurations may be created and loaded into make xconfig, allowing the user to fine tune a system for better performance.

21.1.2 Compiling the Kernel

Now that the Kernel configuration file has been created, we need to compile the Kernel. This is a three-step process.

21.1.2.1 make dep

The make dep process builds a dependency list in order to determine which modules are required, and establishes an order of the pieces.

21.1.2.2 make clean

The make clean process looks at the previously compiled kernel and removes unwanted old files or modules.

21.1.2.3 make zImage or make bzImage

The actual compiling of the kernel takes place with the make zImage process. The differences between zImage and bzImage are two fold:

1. zImage is limited to 1 M Byte in size, whereas bzImage is not.
2. zImage must be used on a non-i386 architecture system.

When the compiled Kernel is complete, it is stored in the `/usr/src/linux/arch/{your-architecture}/boot` directory. In addition to the new Kernel, a `system.map` file is also created.

21.1.3 Installing the Kernel

Now that the Kernel has been created, we need to set it up in the `/boot` directory and set up the boot loader.

21.1.3.1 /boot Directory

It would be very dangerous to just write over our existing kernel. A better approach is to create a new directory below the `/boot` directory in order to maintain the new Kernel. We will set up the boot loader (LILO or GRUB), to first point to our existing operational Kernel, then to the new Kernel. If the new Kernel does not properly install, we can still have an operational system in order to go back and fix our mistake.

Since we have a modified version of our existing Kernel, lets create a new directory called `/boot/linux-2.4.16-M1` (M1 here means first modification).

Now copy the zImage file from the `/usr/src/linux/arch/{your-architecture}/boot` directory to the `/boot/linux-2.4.16-M1` directory. Also copy the `system.map` file.

21.1.4 Boot Loader

Finally we need to set up the boot loader. Assuming that we will be booting from the hard drive, this will be either LILO or GRUB.

21.1.4.1 LILO

Change to the /etc directory. Here we have the lilo.conf file, which is configured to support our existing system. We need add a section that represents the new Kernel. For our example, we would make the following addition at the bottom of the file:

```
image = /boot/linux-2.4.16-M1/vmlinuz-2.4.16-17
label = M1
root = /dev/hda2
read-only
```

Lets explain what each line does.

- image – Specifies the directory location and file name. The vmlinuz file is the one that was copied along with the system.map.
- label – Specifies the label displayed or the name that must be typed in order to invoke the new Kernel.
- root – Specifies the partition that the kernel is located in. In this example, it is same as the one for our existing Kernel.
- read-only – Specifies that during the initial loading, the Kernel image is to be read-only, preventing other loadings from modifying it.

21.1.4.2 GRUB

The grub.conf file resides in the /boot/grub directory. Initially it too is configured to boot our initial system. To the end of this file, we add:

21.X Commands Used in this Chapter

21.X Chapter Review Questions

Chapter Index

	B		Installing	U	4
Boot Loader		5			
GRUB		5	Utility		
LILO		5	make		
	C		clean		4
Compiling New Kernel		3	config		3
	D		dep		4
Directory			menuconfig		3
/boot		4	xconfig		4
/usr/src/linux		3	zImage		4
	K		tar		3
Kernel				W	
Compiling		4	www.kernel.org		3
Configuration		3			