A Guided Tour

It's time to take our tour. The table below lists some interesting places to explore. This is by no means a complete list, but it should prove to be an interesting adventure. For each of the directories listed below, do the following:

- **`cd`** into each directory.
- Use **`ls`** to list the contents of the directory.
- If you see an interesting file, use the **`file`** command to determine its contents.
- For text files, use **`less`** to view them.

Interesting directories and their contents

Directory	***Description***
`/`	The root directory where the file system begins. In most cases the root directory only contains subdirectories.
`/boot`	This is where the Linux kernel is kept. It is a file called **`vmlinuz`**.
`/etc`	The **`/etc`** directory contains the configuration files for the system. Most of the files in **`/etc`** are text files. Points of interest: **`/etc/passwd`** The **`passwd`** file contains the essential information for each user. It is here that users are defined. **`/etc/rc.d`** This directory contains the scripts that get the system started. **`/etc/sysconfig`** Red Hat systems have this directory. It contains a lot of start-up scripts and configuration files for various services.
`/bin, /usr/bin`	These two directories contain most of the programs for the system. The **`/bin`** directory has the essential programs that the system requires to operate, while **`/usr/bin`** contains applications for the system's users.
`/sbin, /usr/sbin`	The **`sbin`** directories contain programs for system administration, mostly for use by the superuser.

/usr	The **/usr** directory contains a variety of things that support user applications. Some highlights: **/usr/X11** The X Windows system **/usr/dict** Dictionaries for the spelling checker. Bet you didn't know that Linux had a spelling checker. See **look** and **ispell**. **/usr/doc** Various documentation files in a variety of formats. **/usr/man** The man pages are kept here. **/usr/src** Source code files. If you installed the kernel source code package, you will find the entire Linux kernel source code here.
/usr/local	**/usr/local** and its subdirectories are used for the installation of software and other files for use on the local machine. What this really means is that software that is not part of the official distribution (which usually goes in **/usr/bin**) goes here. When you find interesting programs to install on your system, they should be installed in one of the **/usr/local** directories. Most often, the directory of choice is **/usr/local/bin**. On Red Hat systems, the **/usr/local** directories are created but they are empty, ready for your use.
/var	The **/var** directory contains files that change as the system is running. This includes: **/var/log** Directory that contains log files. These are updated as the system runs. You should view the files in this directory from time to time, to monitor the health of your system. **/var/spool** This directory is used to hold files that are queued for some process, such as mail messages and print jobs. When a user's mail first arrives on the local system (assuming you have local mail), the messages are first stored in **/var/spool/mail**
/lib	The shared libraries (similar to DLLs in that other operating system) are kept here.
/home	**/home** is where users keep their personal work. In general, this is the only place users are allowed to write files. This keeps things nice and clean :-)

/root	This is the superuser's home directory.
/tmp	**/tmp** is a directory in which programs can write their temporary files.
/dev	The **/dev** directory is a special directory, since it does not really contain files in the usual sense. Rather, it contains devices that are available to the system. In Linux (like Unix), devices are treated like files. You can read and write devices as though they were files. For example **/dev/fd0** is the first floppy disk drive, **/dev/hda** is the first IDE hard drive. All the devices that the kernel understands are represented here.
/proc	The **/proc** directory is also special. This directory does not contain files. In fact, this directory does not really exist at all. It is entirely virtual. The **/proc** directory contains little peep holes into the kernel itself. There are a group of numbered entries in this directory that correspond to all the processes running on the system. In addition, there are a number of named entries that permit access to the current configuration of the system. Many of these entries can be viewed. Try viewing **/proc/cpuinfo**. This entry will tell you what the kernel thinks of your CPU.
/mnt	Finally, we come to **/mnt**, a normal directory which is used in a special way. The **/mnt** directory is used for *mount points*. As we learned in the second lesson, the different physical storage devices (like hard disk drives) are attached to the file system tree in various places. This process of attaching a device to the tree is called *mounting*. For a device to be available, it must first be mounted. When your system boots, it reads a list of mounting instructions in the file **/etc/fstab**, which describes which device is mounted at which mount point in the directory tree. This takes care of the hard drives, but you may also have devices that are considered temporary, such as CD-ROMs and floppy disks. Since these are removable, they do not stay mounted all the time. The **/mnt** directory provides a convenient place for mounting these temporary devices. In a normal installation, you will see the directories **/mnt/floppy** and **/mnt/cdrom**. To see what devices and mount points are used, type **mount**.

A weird kind of file...

During your tour, you probably noticed a strange kind of directory entry, particularly in the **/boot** and **/lib** directories. When listed with **ls -l**, you would have seen something like this:

```
lrwxrwxrwx     25 Jul  3 16:42 System.map -> /boot/System.map-2.0.36-3
-rw-r--r-- 105911 Oct 13  1998 System.map-2.0.36-0.7
-rw-r--r-- 105935 Dec 29  1998 System.map-2.0.36-3
-rw-r--r-- 181986 Dec 11  1999 initrd-2.0.36-0.7.img
-rw-r--r-- 182001 Dec 11  1999 initrd-2.0.36.img
lrwxrwxrwx     26 Jul  3 16:42 module-info -> /boot/module-info-2.0.36-3
-rw-r--r--  11773 Oct 13  1998 module-info-2.0.36-0.7
-rw-r--r--  11773 Dec 29  1998 module-info-2.0.36-3
lrwxrwxrwx     16 Dec 11  1999 vmlinuz -> vmlinuz-2.0.36-3
-rw-r--r-- 454325 Oct 13  1998 vmlinuz-2.0.36-0.7
-rw-r--r-- 454434 Dec 29  1998 vmlinuz-2.0.36-3
```

Notice the files, `System.map, module-info` and `vmlinuz`. See the strange notation after the file names?

These three files are called *symbolic links*. Symbolic links are a special type of file that point to another file. With symbolic links, it is possible for a single file to have multiple names. Here's how it works: Whenever the system is given a file name that is a symbolic link, it transparently maps it to the file it is pointing to.

Just what is this good for? This is a very handy feature. Let's consider the directory listing above (which is the `/boot` directory of an old Red Hat 5.2 system). This system has had multiple versions of the Linux kernel installed. We can see this from the files `vmlinuz-2.0.36-0.7` and `vmlinuz-2.0.36-3`. These file names suggest that both version 2.0.36-0.7 and 2.0.36-3 are installed. Because the file names contain the version it is easy to see the differences in the directory listing. However, this would be confusing to programs that rely on a fixed name for the kernel file. These programs might expect the kernel to simply be called `"vmlinuz"`. Here is where the beauty of the symbolic link comes in. By creating a symbolic link called `vmlinuz` that points to `vmlinuz-2.0.36-3`, we have solved the problem.

To create symbolic links, use the **`ln`** command.