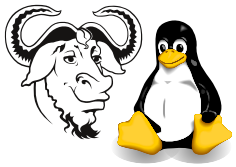

Using grub to Boot various Operating Systems

Contents

1 Aim

After completing these exercises, you will:

- be able to set up grub to boot multiple operating systems
- be able to recover a hard disk with a corrupt MBR
- understand how to start a Linux system in single user mode
- understand how to move between runlevels while the computer is running

2 What You Will Do

Today, you will:

- boot Linux, and create a grub installation disk. I expect you will finish this in twenty minutes.
- boot your computer from the grub installation disk, and re-install grub I expect this will take you about ten minutes. Then we will discuss what you have done.
- You will then destroy your MBR. This is an exercise only; you should not do this at work! The aim is to demonstrate that you can recover from a boot failure. This should take no more than ten minutes. We will then discuss what you have done.
- You will re-install grub into the MBR, and be able to boot your computer again. This should take less than ten minutes (you have done this already!)
- You will modify your grub configuration to boot the operating systems on the internal hard disk. This will take you less than twenty-five minutes. We will discuss this after you have done it.
- You will use grub to boot your computer into *single user mode*. This will take no more than seven minutes.
- You will investigate single user mode. We will discus this as we investigate.
- You will investigate all the other run levels. This will occupy the remaining time in the laboratory.

3 Background

It is often useful to be able to boot more than one operating system from one hard disk. There are some commercial packages to achieve this; examples include *BootMagic* with *Partition Magic* and *System Commander*. There are a number of free software packages also, such as LILO and OSBS. However, we will use grub, because it is very reliable and flexible, and we use it to start Linux. Today we see how to use grub to boot Linux and one or more additional operating systems.

I assume that you have set up the `sudo` command, and understand how to use it. If not, please read the document I have written about it first.

3.1 Installing grub in MBR from a floppy, and from the OS

Grub can be installed into the MBR either directly from a floppy disk, or from within the operating system. We will look at both, since each has its advantages and disadvantages.

3.1.1 Real and Protected Mode: OS cannot run BIOS routines

When the operating system is running, it runs in *protected* mode. When the BIOS is running, it runs in `real` mode. The operating system cannot call BIOS routines while it is running; it can only guess what the BIOS would do, and that is not always correct. For example, you can change the order in which hard disks boot in the BIOS setup, and the OS cannot predict that you will do that!

3.1.2 Making sure grub knows what the BIOS will do

By creating a grub installation floppy, you can be sure that grub and the BIOS agree with each other, since grub runs in real mode, using the BIOS, before the operating system boots.

3.1.3 Using `grub-install` from the OS: the quick but less reliable way

Sometimes, you may want to install grub into the MBR without shutting down the OS. There is a slight possibility that the BIOS and the OS disagree about which disk comes first, so there is a slight possibility that you will need to use the grub installation disk (or a custom boot disk) to fix it.

To install grub into the MBR, we simply type:

```
$ sudo /sbin/grub-install /dev/hda
```

This will install grub into the MBR of `/dev/hda` from the current grub installation.

3.2 The way grub refers to devices

Grub starts numbering devices from zero.

The device syntax used in grub is a bit different from what you may have seen before in your operating system(s), and you need to know it so that you can specify a drive/partition.

Look at the following examples and explanations:

```
(fd0)
```

First of all, grub requires that the device name is enclosed with '`(`' and '`)`'. The '`fd`' part means that it is a floppy disk. The number '`0`' is the drive number, which is counted from *zero*. This expression means that grub will use the whole floppy disk.

```
(hd0,1)
```

Here, '`hd`' means it is a hard disk drive. The first integer '`0`' indicates the drive number, that is, the first hard disk, while the second integer, '`1`', indicates the partition number Once again, please note that the partition numbers are counted from *zero*, not from one. This expression means the second partition of the first hard disk drive. In this case, grub is referring to one partition of the disk, instead of the whole disk.

```
(hd0,4)
```

This specifies the first logical partition of the first hard disk drive. Note that the partition numbers for logical partitions are counted from '4', regardless of the actual number of primary partitions on your hard disk.

Now the question is, how to specify a file? Again, see this example:

```
(hd0,8)/boot/vmlinuz-2.4.18-14
```

This specifies the file named '/boot/vmlinuz-2.4.18-14', found on the ninth partition of the first hard disk drive.

3.3 The Structure of the Master Boot Record

The master boot record (MBR) has three sections; see figure 1. The first 446 bytes contain the code that boots the operating system; the next 62 bytes contain the partition table. The last two bytes contain a "Magic number" that helps identify this sector as a master boot record.

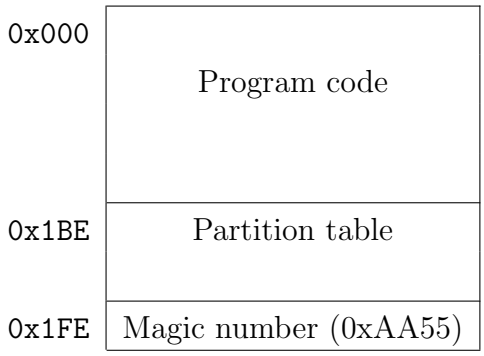

Figure 1: The master boot record structure.

3.4 What does grub do when you install it using setup?

In section 4.2 on page 11, you will use a grub installation floppy disk to install grub into the MBR of your hard disk. What is going on?

There are three steps:

```
grub> find /boot/grub/stage2
grub> root (hd0,8)
grub> setup (hd0)
```

So what is going on in each step?

3.4.1 using find in grub to locate your Linux partition containing your /boot directory

The aim of the first step is to identify paritions on your computer that contain the directory /boot, and the grub directory below that. Grub needs this to be able to install a good MBR. The find command searches all the disks on the computer for the file /boot/grub/stage2, and it lists all partitions where it found it.

The second step, using the grub root command, mounts the partition you found in the previous step. This identifies the type of file system (for example, ext2)

3.4.2 The grub setup command

The third step is the important one that actually sets up the MBR correctly. The grub `setup` command:

1. Determines the `..._stage1_5` file that matches the type of file system found in the previous step
2. puts a list of the sector numbers that contain the file into the code for the MBR
3. installs the code (446 bytes of it) into the MBR.

3.5 How does grub work when it boots your computer from the hard disk?

Here is a summary:

1. BIOS loads the code in the MBR, Stage 1, that you installed in the procedure discussed in section 3.4.2.
2. This short assembly language code, Stage 1 from the MBR, reads the file `/boot/grub/e2fs_stage1_5` (if your `/boot` partition has a file system of type `ext3`). The physical location of this file is hard coded into the MBR during the installation of Stage 1 into the MBR, as described in section 3.4.2.
3. The code in `/boot/grub/e2fs_stage1_5` can now read the file system, and loads the file `/boot/grub/stage2` into RAM, and then executes that code.
4. Stage 2 now contains a sophisticated boot loader, which reads your hard disk and loads the menu file `/boot/grub/grub.conf`. The menu allows you to select an OS using arrow keys, and to select using the (Enter) key. It also allows you to enter the same commands interactively, so that you can boot any operating system on the computer, even if you do not have a menu item for that OS.

3.6 Reading the Documentation for grub

The grub manual consists of about 50 pages of online material. To read it in Emacs, type (Ctrl-h Ctrl-i), then `m grub`. You can view the same documentation using another standalone browser with `pinfo grub`. As a last resort, you can view the same documentation by typing `info grub`, but my favourite way is in Emacs.

3.7 The grub Configuration File `/boot/grub/grub.conf`

Here is an example grub configuration file, `/boot/grub/grub.conf`:

```
1  default=0
2  timeout=10
3  splashimage=(hd0,8)/boot/grub/splash.xpm.gz
4  title Red Hat Linux (2.4.18-14)
5          root (hd0,8)
6          kernel /boot/vmlinuz-2.4.18-14 ro root=LABEL=/1
7          initrd /boot/initrd-2.4.18-14.img
```

```
 8   title Windows 2000
 9           rootnoverify (hd0,0)
10           chainloader +1
```

Let's examine this line by line. Of course, the grub manual (see section 3.6 on the preceding page) gives details of each grub command.

`default=0`

The first line selects which operating system will boot if no grub menu selection is made. The first `title` line in the menu has index 0, so this will boot Linux if you just turn the computer on and do not press any arrow key and then press (Enter).

The second `title` has index 1, so if you want to automatically boot Windows, you would change the line to `default=1`.

`timeout=10`

The second line is the time in seconds before the default entry will be automatically booted.

`splashimage=(hd0,8)/boot/grub/splash.xpm.gz`

The third line selects a graphics image that will be used as a background to the menu. See section 3.2 on page 4 for the meaning of `(hd0,8)/boot/grub/splash.xpm.gz`.

`title Red Hat Linux (2.4.18-14)`

Line 4 provides the text that will appear in the first menu entry.

```
root (hd0,8)
```

Line 5 means: select the ninth partition on the first hard disk as containing the file `stage2`, and mount it. Refer to section 3.2 on page 3 for more about the way grub names devices and partitions.

```
kernel /boot/vmlinuz-2.4.18-14 ro root=LABEL=/1
```

Line 6 is the actual operating system kernel that will be loaded. The kernel can take command line arguments:

`ro` means "read only"; the root partition will be mounted read only at first. Later, during the boot process, the partition will be re-mounted read write after it has been checked for errors.

`root=LABEL=/1` Each partition of type `ext2` or `ext3` can have a label. You can view the labels of mounted partitions with `mount -l`; you can change the labels with the `e2label` command.

Here we are telling the kernel that it should mount the partition with label "`/1`" as the root ('`/`') partition.

Note that labels should be `unique` on a computer system: if two disks have the same label, then the kernel might mount the wrong partition as the '`/`' partition.

You can also add other command line arguments. There are many that the kernel can accept; a number will select the runlevel that the operating system will default to—see section 4.7.1 on page 14, particularly item number 5.

```
initrd /boot/initrd-2.4.18-14.img
```

Line 7 specifies a file containing device drivers that will be put into a temporary RAM disk before the kernel accesses the hard disk. This solves a problem: if the kernel needs drivers to read the hard disk, how can the kernel get the drivers? The answer is from the "`initrd`" RAM disk. You can generate an `initrd` image file using the program `mkinitrd`.

```
title Windows 2000
```

Line 8 gives the text for the second item in the grub boot menu.

```
rootnoverify (hd0,0)
```

Line 9 tells grub that it should read from this partition (`/dev/hda1`), but not try to mount it.

```
chainloader +1
```

Line 10 means: read from the root device (in this case, the device is `(hd0,0)`, or `/dev/hda1`), starting at sector zero, and finishing just before sector 1. In other words, it loads the first sector from this partition, and transfers control to it. This will boot Windows, since Windows has installed a boot loader in that sector.

3.8 Chainloading: Can I Install grub Anywhere Besides the MBR?

Yes! You can also install grub in the boot partition of your operating system. This is an option when installing Red Hat Linux. When would you want to do that? You would do this if you are using another boot loader to boot your operating system. This is called *chainloading*. We do this when we boot Windows: grub chainloads the Microsoft bootloader from the first sector of the partition where Windows is installed.

3.8.1 Example: many Linux installations on one hard disk: part time classes

group	*partition where* `/` *mounted*	grub *device name*	*where* grub `stage1` *installed*
A1	`/dev/hda1`	`(hd0,0)`	MBR `(hd0)`
A2	`/dev/hda5`	`(hd0,4)`	`(hd0,4)`
B	`/dev/hda7`	`(hd0,6)`	`(hd0,6)`

Table 1: The partitioning arrangement for part time classes.

For example, in part-time OSSI classes, three students share one hard disk. The first laboratory group, group A1, boots Linux from the first primary partition, `/dev/hda1`. The second group A2 installs Linux into the logical partition `/dev/hda5`, while students from the third laboratory class installs Linux into the logical partition `/dev/hda7`. See table 1. It is best if each class takes responsibility for booting their own operating system. The way it is set up is like this.

1. Group A1 install `stage1` of grub into the MBR. Their grub cconfiguration file looks like this:

```
default=0
timeout=10
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
title Red Hat Linux for group A1 (2.4.18-14)
        password --md5 $1$tupXW/$yCwElzrIIVoE0cfnxTlUS0
        root (hd0,0)
        kernel /boot/vmlinuz-2.4.18-14 ro root=/dev/hda1
        initrd /boot/initrd-2.4.18-14.img
title Red Hat Linux for group A2
        rootnoverify (hd0,4)
        chainloader +1
title Red Hat Linux for group B
        rootnoverify (hd0,6)
        chainloader +1
```

2. Group A2 has a `/boot/grub/grub.conf` file that looks like this:

```
default=0
timeout=1
splashimage=(hd0,4)/boot/grub/splash.xpm.gz
title Red Hat Linux A2 (2.4.18-14)
        password --md5 $1$r0qXW/$Y7.OuCvhJcKrOgERkZtkt.
        root (hd0,4)
        kernel /boot/vmlinuz-2.4.18-14 ro root=/dev/hda5
        initrd /boot/initrd-2.4.18-14.img
```

3. Group B have a similar configuration to group A2, but with the root set to `/dev/hda7`.

4. The way it works is that if group A2 want to boot their OS, they select their menu item and press (Enter). Their operating system then boots. What is happening is:

   - The menu installed by group A1 appears, offering to boot any of the three installed version of Linux
   - If you select the second choice (for group A2), then the grub installed in the MBR and in `/dev/hda1` *chainloads* grub from the first sector of `/dev/hda5`.
   - The second grub starts up, and after a 1 second delay, boots the Linux installed in `/dev/hda5`.

   This is a case of a chain of boot loaders, where one bootloader loads another boot loader, which finally loads the operating system.

5. The advantage is that the grub set up in `/dev/hda1` by group A1 does not need to know what version of kernel is set up in partitions `/dev/hda5` or `/dev/hda7`. The other two groups can upgrade their kernels without disturbing the boot process, which begins with the grub installed by group A1.

6. What about the "`password`" lines? These require a password to be entered before that boot menu item will boot. It provides protection for students of one group from accidentally booting the partition of another group.

   How to generate the password?

```
$ grub-md5-crypt
Password: ⟨type your password here⟩
$1$r0qXW/$Y7.OuCvhJcKrOgERkZtkt.
```

Then use your mouse to copy and paste the hashed password into the test editor editing `/boot/grub/grub.conf`.

3.9 Runlevels and `init`

- Linux has seven modes of operation
- Referred to as *runlevels*
- The Linux Standards Base (`http://www.linuxbase.org/spec/refspecs/LSB_1.1.0/gLSB/runlevels.html`) defines the following standard runlevels that all distributions must follow to be compliant:

  0 halt
  1 single user mode
  2 multiuser with no network services exported
  3 normal/full multiuser
  4 reserved for local use, default is normal/full multiuser
  5 multiuser with `xdm` or equivalent
  6 reboot

3.9.1 Single User Mode

Often, booting into *single user mode* is the solution to a problem. Single user mode is a bit like "Safe Mode" in Windows. No networking is started, and only one user (`root`) can log in. It is a quick way to fix a forgotten `root` password, and it is a good way to fix problems that prevent services from starting. Single user mode is also called runlevel 1.

3.9.2 How to Change Runlevels with `init`

- The `init` process governs runlevels. `init` is *the first program that the kernel runs.*
- You can change runlevels as the administrator by

  `$ sudo /sbin/init` *level*

  For example, if you are currently in runlevel 5, you can change to runlevel 3 by executing the command:

  ```
  $ sudo /sbin/init 3
  ```

  Runlevel 3 is useful for fixing problems with starting X (the graphical user interface).

  You can change to single user mode from any other level by:

  ```
  $ sudo /sbin/init 1
  ```

If you are in single user mode, you can change to runlevel 5 by the same method:

```
# init 5
```

The program `/sbin/runlevel` will show you the current runlevel, and the previous runlevel.

There is an exercise in section 4.7.1 on page 14 which explains how to boot the computer directly into any runlevel, using grub.

4 Procedure

Before you begin:

- Linux should be installed
- You need a good floppy disk to create a grub installation disk.

Here I am going to assume that you have installed Windows so that its boot files are in partition `/dev/hda1`.

4.1 Creating a Grub Installation Disk

1. Use a good, empty, formatted disk. In Linux, you can do:

   ```
   $ fdformat /dev/fd0
   ```

   to make sure that the disk has no bad sectors.

   > This does a *low-level format* of the floppy disk. It divides the floppy into the appropriate number of cylinders (80), and the appropriate number of sectors per track (18), and writes information onto the disk so that the file system may be put into these sectors. Each sector contains 512 bytes, so there are $80 \times 18 \times 512 = 737280$ bytes on one side, and so a total of $737280 \times 2 = 1474560$ bytes, or 1440 kilobytes.

2. Then make the FAT file system with:

   ```
   $ /sbin/mkdosfs /dev/fd0
   ```

   OR

   you could, instead of a FAT filesystem, make an extended 2 file system like this if you want to:

   ```
   $ /sbin/mke2fs /dev/fd0
   ```

   > Note that when formatting a floppy disk, Microsoft `format` does *both* a low level format and adds the file system (with a "high level format") at the same time. However, when the same program formats a hard disk, it performs a high level format only. Linux tools keep the two operations separate.

3. Then mount the floppy disk. You can do that easily with the shell functions `a` and `z` I appended to your ∼`/.bashrc` login script. To do it manually:

```
$ mount /dev/fd0
```

Or using my shell functions:

```
$ a
```

The end result is that the floppy disk is mounted on the directory `/mnt/floppy`. This means that the contents of the floppy disk are available under that directory.

4. Check the the floppy is *mounted* with the `mount` command. First type:

```
$ mount
```

to see all the mounted filesystems on your computer. Hmm, there is rather a lot. So let us filter out all but the one we are looking for with `grep`:

```
$ mount | grep /dev/fd0
```

You should see which directory the floppy disk device (`/dev/fd0`) is *mounted* on. If you see no output, your floppy disk is not mounted, so go back to item 3.

5. Now install grub into the floppy disk with:

```
$ /sbin/grub-install --root-directory=/mnt/floppy '(fd0)'
```

6. When the installation completes successfully (no error messages), then unmount the floppy. Do not remove the floppy before you unmount it:

```
$ umount /dev/fd0
```

Note that if your current directory is on the floppy disk, you will not be able to unmount it. It is a bit like trying to lift a wooden plank you are standing on. You need to get off any directory on the device before you can unmount it.

Or, using my handy little shell functions:

```
$ z
```

7. Remove your boot floppy only *after* floppy disk activity has stopped, as shown by the light going out on the floppy drive. You now have a grub installation disk.

4.2 Using the grub installation disk to install grub into the MBR

Please refer to section 3.4 on page 4 while doing this activity.

1. Put your grub installation disk into the computer, and restart it.
2. After a while, you should see the grub prompt.
3. Select which partition you are going to set up grub for. This is the partition that contains the file `/boot/grub/stage2` It is best to let grub determine which partition holds your Linux grub files, as described in section 3.4.1 on page 4:

```
grub> find /boot/grub/stage2
```

This will search all the disks in your computer for the file name `/boot/grub/stage2` and show the partitions which contain the file. Look for the right partition, and select it as your root device:

```
grub> root (hd0,8)
```

This will determine the file system type of the partition, and mount the partition.

4. Once you've set the root device correctly, run the command `setup`:

```
grub> setup (hd0)
```

> This will put the file `stage1` into the MBR on your hard disk, along with a list of hard disk block numbers telling it where to find the file `e2fs_stage1_5`. Later, when `grub` loads this file, it will then be able to read individual files on the hard disk, and will be able to load the file `/boot/grub/stage2`, which provides most of the functionality of `grub`.

4.3 Destroying your Master Boot Record!

Note: this is an exercise only. Do not do this in your workplace! The aim here is to get your computer into an unbootable state, and to demonstrate that you can recover from this situation. It is *never* necessary to delete the MBR for you to install `grub`.

Here we are going to do some very dangerous things, so *be **very** careful*. Look once, look twice, look *three* times before you press (Enter). There is no turning back if you destroy data on your disk beyond your MBR.

First you will mount your `grub` installation disk, and make sure it is mounted. Then you will make a backup of your MBR into a file on your `grub` installation boot disk by copying the first 512 bytes from your hard disk to a file on your floppy disk. Finally you will overwrite the first 446 bytes of your master boot record with binary zeros, making your computer unbootable.

1. Mount your `grub` installation disk, and make sure it is mounted:

```
$ mount /dev/fd0
$ mount | grep fd0
```

2. Now use the data dump program to backup the master boot record into a file on your floppy disk:

```
$ sudo dd if=/dev/hda of=/mnt/floppy/master-boot-record.img bs=512 count=1
```

3. Now *destroy* your master boot record, *carefully*:

```
$ sudo dd if=/dev/zero of=/dev/hda bs=446 count=1
```

> **Note** that the block size is 446 bytes and *not* 512. The partition table is contained in the last 64 bytes of the MBR, so don't delete that! See figure 1 on page 4.
>
> The device `/dev/zero` returns 0 whenever you read it, so this command will totally clear the code part of the MBR, and replace it with all binary zero bits.

4. Now try to boot your hard disk. It should fail.

4.4 Recovering From Disaster

1. Now use your grub installation disk to re-install grub in the MBR.
2. Verify that you can boot both Linux and Windows.

4.5 Booting an operating system interactively with a grub installation disk

4.5.1 Overcoming a limitation of the Windows Bootloader

The Windows bootloader will only boot from the first hard disk ("first" is defined by the BIOS before the operating system starts). How can we boot Windows from the second hard disk? By having grub tell the BIOS to swap the hard disks, so that it thinks `/dev/hdc` is the first hard disk, and `/dev/hda` is the second hard disk. Read about the `map` grub command in the grub manual; see section 3.6 on page 5.

4.5.2 Experiments with grub

You can use the grub shell to interactively boot any operating system installed on a computer. Try this:

1. Boot your computer with your grub installation disk
2. at the `grub>` prompt, type the following:

```
grub> map (hd1) (hd0)
grub> rootnoverify (hd1,0)
grub> chainloader +1
grub> boot
```

3. What happens? Which partition did this operating system boot from?
4. Try this sequence of grub commands:

```
grub> map (hd1) (hd0)
grub> rootnoverify (hd0)
grub> chainloader +1
grub> boot
```

5. Test it. Explain what is happening.

4.6 Setting up a Menu to Boot other Operating Systems

Refer to section 3.7 on page 5 to do this activity.

1. Make a backup of your grub configuration file to the `tmp` directory (the `root` user cannot write to your network home directory):

   ```
   $ sudo cp /boot/grub/grub.conf /tmp/grub.conf)
   ```

2. Now edit your grub configuration file so that it can boot Linux from the internal hard disk, and can also boot Windows from the internal hard disk. You may need to refer to section 4.5 on the previous page for some ideas on how to boot Windows from the second hard disk.

3. Test your setup and verify that you can boot these operating systems.

4.7 Investigating Runlevels

Refer to section 3.9 on page 9 when doing this activity.

4.7.1 Booting into Runlevel 1, or Single User Mode

Refer to the description of single-user mode in section 3.9.1 on page 9 before doing this activity.

You need to add a '`1`' at the end of the `grub` line that selects the OS kernel. To do this, do the following:

1. Select the desired kernel.

2. Press the (e) key to edit that entry.

3. Use the arrow keys to navigate to the kernel line (for example: `kernel /boot/vmlinuz-2.4.18-14 ro root=LABEL=/1`)

4. Press the (e) key to edit the line.

5. Add the argument '`1`' (or '`single`') to the end of the line and press (Enter). The line will then look something like this:

   ```
   kernel /boot/vmlinuz-2.4.18-14 ro root=LABEL=/1 1
   ```

   This selects the runlevel. In the same way, you could boot the system into runlevel 3 instead of the normal runlevel 5 by adding a '`3`' at the end instead of '`1`'.

6. Press the (b) key to boot.

4.7.2 Examining single user mode

Examine the conditions of single user mode:

1. Did you need to enter your password?

   You can change the root password by typing:
   ```
   # passwd
   ```

   and, when prompted each of two times, enter a new password.

2. Can you change to another virtual console by pressing the key combination (Alt-Ctrl-F2)?

3. Is networking available?

   Some quick ways to check whether any network devices are active is by typing:
   ```
   $ /sbin/ifconfig
   ```

   You can also examine the `routing table` by typing
   ```
   $ /sbin/route -n
   ```

   Please try these two commands at a number of runlevels.

4. Can you change to runlevel 5 without rebooting?

4.7.3 Investigating the Runlevels

1. Change to all the available runlevels. At each runlevel,
   - **(a)** determine whether networking is up;
   - **(b)** verify your current runlevel with the `/sbin/runlevel` program.
   - **(c)** Can you change to another virtual console (VC) (by pressing the key combinations (Alt-Ctrl-F2), ..., (Alt-Ctrl-F6)?
   - **(d)** Can you change to the graphical login screen by pressing the key combination (Alt-Ctrl-F7)?
   - **(e)** Complete this table:

runlevel	*networking?*	*can change VC?*	*graphical login?*
1			
2			
3			
4			
5			

2. What happens when you switch to runlevel 0?

3. To runlevel 6?

4.7.4 The Runlevel Scripts

The directory `/etc/rc.d` contains the shell scripts associated with starting up *services* on the system. In it are a number of other directories, and the scripts `rc`, `rc.sysinit` and `rc.local`:

```
$ ls
init.d  rc0.d  rc2.d  rc4.d  rc6.d        rc.sysinit
rc      rc1.d  rc3.d  rc5.d  rc.local
```

1. Look in each of the directories `init.d`, `rc0.d`, `rc1.d`, ..., `rc6.d`. Change into each one, and list the contents of the directory:

   ```
   $ cd /etc/rc.d/init.d
   $ ls -l
   $ cd ../rc0.d
   $ ls -l
   $ cd ../rc1.d
   $ ls -l
   ...
   $ cd ../rc6.d
   $ ls -l
   ```

2. Discuss the purpose of what you see.

   > The files in the directories `rc`*n*`.d` are *symbolic links*. They are all linked to the shell scripts in the directory `/etc/rc.d/init.d`.

3. You can start a service using the program `/sbin/service`. The web server service is called `httpd`.

4. Go to see the default web site of your web server by opening the URL in your web browser: `http://localhost.tyict.vtc.edu.hk/`. Can you see any web page?

5. Start the web server by typing:

   ```
   $ sudo /sbin/service httpd start
   ```

6. Now return to view the default web page. Can you see it now?

Index