Exam Manual

LearnSmart

(Exam 102) LPIC-1

This LearnSmart exam manual covers the most important topics you will encounter on the LPIC-1 Exam 102. By studying this manual, you will gain familiarity with many exam-related objectives, including:

- Kernel
- Boot, Installation, Shutdown and Runlevels
- Documentation
- Networking Fundamentals
- And more!

Give yourself the competitive edge necessary to further your career as an IT professional and purchase this exam manual today!

LPIC-1 – Junior Level Administration Part 2 (102) LearnSmart Exam Manual

Product ID: 010717
Production Date: July 25, 2011

Warning and Disclaimer

Every effort has been made to make this document as complete and as accurate as possible, but no warranty or fitness is implied. The publisher and authors assume no responsibility for errors or omissions. The information provided is on an "as is" basis. The authors and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this document.

Volume, Corporate, and Educational Sales

Favorable discounts are offered on all products when ordered in quantity. For more information, please contact us directly:

1-800-418-6789
solutions@learnsmartsystems.com

International Contact Information

International: +1 (813) 769-0920

United Kingdom: (0) 20 8816 8036

Table of Contents

Abstract

This Exam Manual will help you prepare for the Linux Professional Institute LPI 102 exam. This exam is one of two exams (the other is LPI 101) that candidates must pass in order to be awarded the LPIC-1 certification (level 1 Linux administration). This exam contains between 60 and 90 multiple-choice and fill-in-the-blank questions. Candidates have up to 2 hours to complete the exam.

What to Know

The objectives covered in the LPI 102 exam are:

- Topic 105: Kernel
- Topic 106: Boot, Initialization, Shutdown and Runlevels
- Topic 107: Printing
- Topic 108: Documentation
- Topic 109: Shells, Scripting, Programming and Compiling
- Topic 111: Administrative Tasks
- Topic 112: Networking Fundamentals
- Topic 113: Networking Services
- Topic 114: Security

Tips

- The LPI exams cover a broad range of Linux administration topics. It is useful to study all of the areas while keeping in mind that the weighting of each objective indicates how many of the exam questions will be devoted to the objective.
  There is no substitute for hands-on experience.

Topic 105: Kernel

1.105.1 Manage/Query kernel and kernel modules at runtime (Weight: 4)

Candidates should be able to manage and/or query a kernel and kernel loadable modules.

The Linux Kernel

The Linux Kernel is responsible for handling many tasks, including memory management, program loading and process scheduling, and providing access to hardware for applications.

Kernel versions are numbered major.minor.patch. If the minor version number is odd, the kernel is considered a development or testing version. If the minor version number is even, the kernel is considered a production version.

You can determine the version of kernel with the command:

```
[matt@pitt:~] uname -r
```

Output:
2.6.15-26-386

Also, you can find out more information about your system using the –a option to uname, for example:

```
[matt@pitt:~] uname -a
```

Output:
Linux otoole 2.6.15-26-386 #1 PREEMPT Mon Jul 17 19:52:53 UTC 2006 i686 GNU/Linux

Numbers after the '-' character indicate package information from your Linux distribution.

Kernel Modules

Modules are stored in the directory /lib/modules/`uname -r`, where 'uname -r' is the version of the kernel.

Linux uses a modular kernel; thus, unneeded functionality need not be included in the running kernel and using extra memory. Additionally, as you need more functionality, you can simply load a module to provide that functionality while you need it.

An example directory tree under /lib/modules can be found; for example:

```
[matt@pitt:~] tree -d /lib/modules
```

Output:

```
/lib/modules
|-- 2.6.15-23-386
| |-- initrd
| |-- kernel
| | |-- arch
| | |-- crypto
| | |-- drivers
| | | |-- cdrom
| | | |-- char
| | | | |-- agp
| | | |-- parport
| | | |-- scsi
| | | | |-- aacraid
| | | | |-- aic94xx
| | |-- net
`-- 2.6.15-26-386
| |-- initrd
| |-- kernel
```

The preceding directory tree has been shortened. It demonstrates how modules for different kernel versions are separated.

Viewing Modules in Use

To determine which modules are currently loaded, use the lsmod(8) command:

```
[matt@pitt:~] lsmod
```

Output:

```
Module       Size Used by
ntfs       103536 1
e1000      118840 0
pcmcia      40508 0
rtc         13492 0
floppy      62148 0
snd_intel8x0    33692 1
snd_ac97_codec    93088 1 snd_intel8x0
snd_ac97_bus    2304 1 snd_ac97_codec
snd_pcm_oss    53664 0
snd_mixer_oss    18688 1 snd_pcm_oss
```

The list on the previous page has been shortened. The columns in the output are:

- Column 1 - The module name
- Column 2 - The amount of memory used by the module
- Column 3 - The count of how many other modules are using functionality from this module
- Column 4 - A list of the modules that are using the functionality

In the above listing, you can see that snd_ac97_codec relies on snd_ac97_bus. The snd_ac97_bus module, in turn, depends on the snd_intel8x0 module.

Loading and Unloading Kernel Modules

To install a simple module, use the insmod(8) command. To load support for fat file systems, use something such as:

```
[root@pitt:~] insmod /lib/modules/`uname -r`/kernel/fs/fat/fat.ko
```

On some distributions, you need not specify the full path to the module.

To unload a module, use the rmmod(8) command. You can now refer to the module by name instead of having to provide the full path.

```
[root@pitt:~] rmmod fat
```

Handling Module Dependencies

Due to the modular nature of the Linux Kernel, sometimes you cannot load a module because it depends on functionality from another module. For example, if you wanted to use a vfat or fat32 file system and try to simply load that module, your results would be:

```
[root@pitt:~] insmod /lib/modules/`uname -r`/kernel/fs/fat/vfat.ko
```

Ouput:
insmod: error inserting '/lib/modules/2.6.15-26-386/kernel/fs/vfat/vfat.ko': -1 Unknown symbol in module

To alleviate the problem presented here, the modprobe(8) command. modprobe will load module dependencies before the actual module that is required. With the vfat module example, you can perform the following:

```
[root@pitt:~] modprobe vfat
```

To confirm that the module was loaded, verify it with the lsmod(8) command. The following command shows that both the fat and vfat modules were loaded:

```
[root@pitt:~] lsmod |grep fat
```

Output:
```
vfat        13440 0
fat         53020 1 vfat
```

To remove these modules when you are finished, use the -r option to modprobe:

```
[root@pitt:~] modprobe -r vfat
```

This command will remove the vfat module. It will then determine if any other modules are no longer required (such as the fat module) and unload them as well.

The modprobe program can determine module dependencies by reading the file /lib/modules/`uname -r`/modules.dep file. This file is created using the depmod(8) program. The depmod(8) program is run on most distributions at boot time but you can also run it whenever it is needed.

A sample of the modules.dep file looks like this:

```
[matt@pitt:~] grep fat /lib/modules/2.6.15-23-386/modules.dep
```

Output:
```
/lib/modules/2.6.15-23-386/kernel/fs/msdos/msdos.ko: /lib/modules/2.6.15-23-386/kernel/
fsfat/fat.ko
/lib/modules/2.6.15-23-386/kernel/fs/vfat/vfat.ko: /lib/modules/2.6.15-23-386/kernel/fs/fat/
fat.ko
/lib/modules/2.6.15-23-386/kernel/fs/fat/fat.ko:
```

The module on the left side of the ':' is the target module. Any modules listed on the right side are its dependencies. In this example, the msdos and vfat modules require the fat module. However, the fat module has no dependencies.

Passing Parameters to Modules

Many modules will accept additional configuration parameters. To determine the available parameters for a module, use a command such as:

```
[matt@pitt:~] modinfo parport_pc
```

Output:
```
filename:   /lib/modules/2.6.15-26-386/kernel/drivers/parport/parport_pc.ko
author:     Phil Blundell, Tim Waugh, others
description:  PC-style parallel port driver
license:    GPL
vermagic:   2.6.15-26-386 preempt 486 gcc-4.0
depends:    parport
alias:      pci:v00001106d00000686sv*sd*bc*sc*i*
alias:      pci:v00001106d00008231sv*sd*bc*sc*i*
alias:      pci:v00001283d00008872sv*sd*bc*sc*i*
...clipped...
```

This command's output can be very long. To limit output to only the module parameters, use the -p option. For example:

```
[matt@pitt:~] modinfo -p parport_pc
```

Output:
```
io:Base I/O address (SPP regs)
io_hi:Base I/O address (ECR)
irq:IRQ line
dma:DMA channel
verbose_probing:Log chit-chat during initialization
init_mode:Initialise mode for VIA VT8231 port (spp, ps2, epp, ecp or ecpepp)
```

This output shows that the system (and user) may specify alternate I/O addresses, interrupts, DMA channels and more.

These values can also be stored in the /etc/modules.conf file (some distributions have other mechanisms as well). For example, the eth0 network interface can be told explicitly which module and parameters to use with entries in /etc/modules.conf such as:

```
alias eth0 tulip
options eth0 io=0x456
```

1.105.2 Reconfigure, build and install a custom kernel and kernel modules (Weight: 3)

Candidates should be able to customize, build and install a kernel and kernel-loadable modules from source.

Customize the current kernel configuration. Build a new kernel and appropriate kernel modules. Install a new kernel and any modules. Ensure that the boot manager can locate the new kernel and associated files.

Getting Kernel Source

Sometimes you need a different version of the Linux Kernel than the one provided by your Linux distribution. One way to get a different version is to download it from http://www.kernel.org.

The Linux Kernel website will also provide an MD5 checksum for use in confirming that you downloaded the correct kernel and that it is not corrupt. To confirm your kernel download, use the md5sum(1) command:

```
[matt@pitt:~] md5sum linux-2.6.12.tar.gz
```

Output:
```
3a0beb8c3787915ae0d51d590302d3c5 linux-2.6.12.tar.gz
```

You can also confirm that the kernel source's gpg signature matches the supplied one, as follows:

```
[matt@pitt:~] gpg --verify linux-2.6.12.tar.sign linux-2.6.12.tar.gz
```

Output:
```
gpg: Signature made Sat 03 Apr 2004 11:27:25 PM EST using DSA key ID 517D0F0E
gpg: Good signature from "Linux Kernel Archives Verification Key <ftpadmin@kernel.org>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:    There is no indication that the signature belongs to the owner.
Primary key fingerprint: C75D C40A 11D7 AF88 9981 ED5B C86B A06A 517D 0F0E
```

The significant part of the message is "Good signature from "Linux Kernel Archives...". You may disregard the warnings.

If you need the kernel public key for GPG, retrieve it with the following:

```
[matt@pitt:~] gpg --search-key ftpadmin@kernel.org
```

Choose the key that has no "revoked" comment on it.

Configuring the Kernel

The typical location for unpacking your kernel source code is in /usr/src. A new directory will be created as /usr/src/linux-<version>.

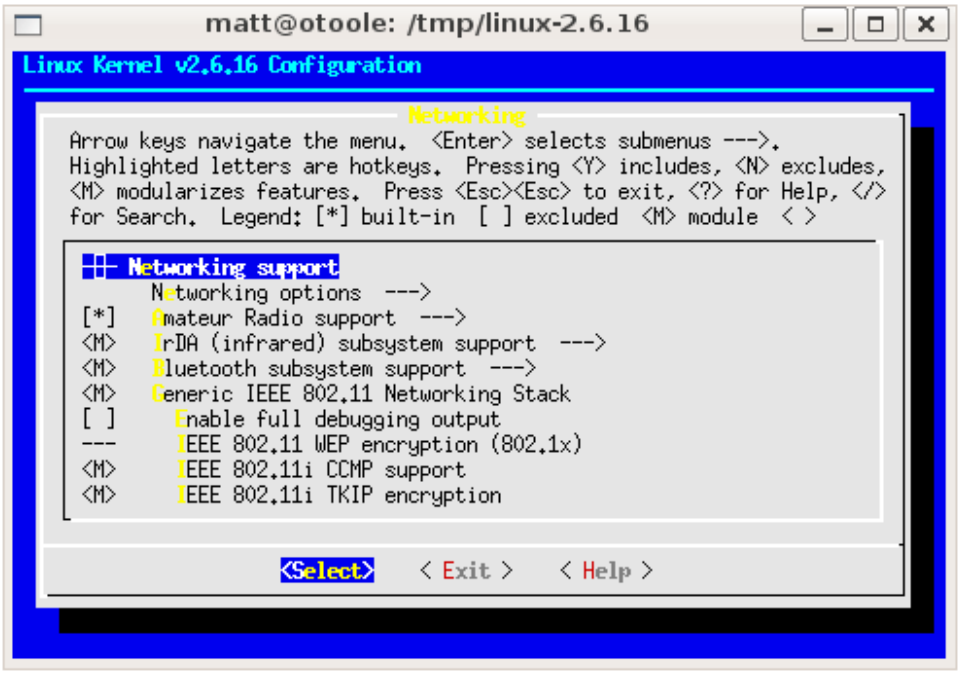
The /usr/src/linux directory should be a symbolic link to the kernel source (also in /usr/src) against which you are trying to compile additional software. Under normal circumstances, you will not need a /usr/src/linux directory.

Once you have unpacked your source code, change directory to the top directory of the source code. You will have a number of configuration commands available through the use of the make(1) command and the Makefile file at the top of the kernel source directory tree. Current configuration options include:

Configuration Target	Description
make config	This is the most foolproof (but tedious) option. A long series of Yes/No questions will be asked; if there are any mistakes, you have to start over.
make menuconfig	A text-based configuration tool with some simple and useful navigation for choosing your options.
make xconfig	A QT-based configuration front end.
make gconfig	A GTK-based configuration front end.
make oldconfig	Update current config using a provided .config as a base.

Once you have successfully created a configuration, the options are saved in the file .config. Protect this file. There is no backup and the file can easily be deleted.

An example of the "make menuconfig" option looks like:



The top of the screen explains your choices with each option:

Choice	Description
[ ]	Empty square brackets indicate that this option is turned off. Hit the space bar to turn on the option.
[*]	An asterisk in the square brackets indicates that this option is turned on and will be compiled directly into the kernel. Hit the space bar to turn off the option.
< >	Angled brackets indicate that this functionality can be turned on (<*>), off (< >) or compiled as a module (<M>). Press the "m" key to select compilation as a module.

Full Compilation Steps

There are many options and paths through a kernel compilation. The following is a sample path.

1. Extract the source code from your kernel tarball with a command such as:

[root@pitt:~] cd /usr/src

[root@pitt:~] tar zxvf /tmp/linux-<version>.tar.gz

2. Change directory into the source tree and choose a configuration command, such as:

[root@pitt:~] cd linux-<version>

[root@pitt:~] make menuconfig

If you have any trouble compiling the configuration tool (part of the "make menuconfig" command), you may need to install the gcc compiler or the ncurses-dev package. The -dev and -devel packages include header files and other items needed for software compilation.

3. Make your selections. As an example, navigate into:

Device Drivers -> Network Device Support

And change "Dummy Net Driver support" to be compiled as built-in.

4. When you have finished making selections, press the "Esc" key until you are asked if you would like to save the configuration. Select "Yes" and press Enter. This step will write out a new .config file.

5. Now you can begin the compilation process with:

[root@pitt:~] make dep

This step will determine all of the compile-time dependencies based on your previous selections.

6. Ensure that you are not reusing any compiled source code from a previous build with:

[root@pitt:~] make clean

7. Now it is time to compile the kernel with:

[root@pitt:~] make bzImage

8. Next, compile the modules with:

[root@pitt:~] make modules

9. You can install the modules once the prior step is complete with:

[root@pitt:~] make modules_install

This step will copy the modules into /lib/modules/<version just compiled>

10. You will also need to copy the kernel itself to the /boot directory with something such as:

[root@pitt:~] cp arch/i386/bzImage /boot/vmlinuz-<version>

11. As a precaution, create an initial RAM disk with the mkinitrd or mkinitramfs utilities. An example of using mkinitrd is:

[root@pitt:~] mkinitrd -o /boot/initrd-<version> <version>

Use of these utilities varies from distribution to distribution. Please consult the man pages for proper instructions.

12. Update your /etc/lilo.conf or /boot/grub/menu.lst file to reflect the new kernel (and initial RAM disk). Remember to reinstall the lilo boot loader, if that is what you are using.

It is also not advisable to delete the previous kernel from the system and the boot loader configuration until you are certain that the new kernel is working.

13. Reboot to try out your new kernel.

Topic 106: Boot, Initialization, Shutdown and Runlevels

1.106.1 Boot the system (Weight: 3)

Candidates should be able to guide the system through the booting process.

The boot process includes the following steps:

1. System power comes on and the computer goes through the POST step (Power-On Self Test).
2. The system loads a BIOS (Basic Input/Output System). The BIOS initializes hardware. Its its final responsibility is to load something from a boot sector on another device and pass on execution to the code that is loaded from the boot sector.
3. The boot sector contains a Boot Loader (such as GRUB or LILO). This boot sector can be the first sector on a floppy disk, USB memory stick, CD-ROM or a hard drive. On a hard drive, the first sector is called the Master Boot Record (MBR); it also includes partition information.
4. The Boot Loader's responsibility is to find either another Boot Loader or, more commonly, a Linux Kernel to load. At this point, the user may be able to provide information that will be passed to the Linux Kernel to change hardware settings, the runlevel and many other settings.
5. Once a Linux Kernel is chosen (or the default is selected or the time period for user choices has expired), the Boot Loader will load the Linux Kernel into memory and pass on execution to the Linux Kernel.
6. The Linux Kernel will initialize devices, load an initial RAM disk (initrd) and any required modules. Its final steps are to mount the root file system (which may be overridden with the root=/dev/hdXXX parameter) and load and execute the /sbin/init program (which may also be overridden with the init=/some/program parameter).
7. The /sbin/init program becomes the first process on the running Linux operating system and has a PID (Process ID) of 1.
8. /sbin/init will read its configuration file /etc/inittab to determine the default runlevel unless a runlevel was passed to the Linux Kernel during the Boot Loader stage. The line in /etc/inittab that determines the default run level looks like the following:

   id:5:initdefault:

   The numeric value is the default run level.
9. The init(8) process will then run all programs that are configured for the chosen run level. These are also defined in the /etc/inittab file. Some of these programs are getty programs (such as mingetty) which provide console logins and graphical login programs such as xdm, gdm and kdm.

The /etc/inittab file

The general formatting for the /etc/inittab file is:

id:runlevels:action:process

An example is the getty programs that run to give you console logins. These entries look like the following:

```
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
3:23:respawn:/sbin/getty 38400 tty3
4:23:respawn:/sbin/getty 38400 tty4
5:23:respawn:/sbin/getty 38400 tty5
6:23:respawn:/sbin/getty 38400 tty6
```

The first field is just an ID. In this case, they are numbered to coincide with the tty (terminal device) to which the getty program will read and write.

The second field is the runlevels in which the getty command will run. For ID '1', the /sbin/getty program (with two arguments) will run if the system boots or switches to runlevels 2, 3, 4 or 5. All other entries will be executed only if the system is in runlevels 2 or 3.

The third field is the action. In this case, init(8) will restart the getty if it exits. This step ensures that the login is always available again after use.

The fourth field is the actual program and arguments to run.

There are other possible values for the "action" as well. They include:

Action	Description
respawn	The process will be restarted whenever it terminates. It is commonly used for getty programs, and some distributions use it to provide a graphical login manager.
wait	The process will be started once when the specified runlevel is entered and init will wait for its termination.
once	The process will be executed once when the specified runlevel is entered.
ondemand	A process marked with an ondemand runlevel will be executed whenever the specified ondemand runlevel is called. However, no runlevel change will occur (ondemand runlevels are 'a', 'b', and 'c').
initdefault	An initdefault entry specifies the runlevel which should be entered after system boot. If none exists, init will ask for a runlevel on the console. The process field is ignored.
sysinit	The process will be executed during system boot. It will be executed before any boot or bootwait entries. The runlevels field is ignored.

Using the three finger salute (Ctrl-Alt-Delete)

There is a special action called 'ctrlaltdel' available in /etc/inittab. A typical entry looks like the following:

```
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
```

This setting may not be safe for servers, so you can change it to:

```
ca:12345:ctrlaltdel:/bin/echo "your seat is on fire"
```

This change, however, will not take immediate effect. You can either wait until the next reboot or send a message to init telling it to reread its /etc/inittab file. This message is sent as follows:

```
[matt@pitt:~] init q
        or
[matt@pitt:~] kill -HUP 1
```

The latter example works because Unix has a tradition that sending a HUP signal to a daemon will cause it to reread configuration files and close and reopen any open files (basically, a smart reset of the service).

Runlevel Directories

Most of the services made available by a Linux system are through "init" scripts that are run due to an entry in the /etc/inittab file. The exceptional entries in which we are interested look like the following:

```
[matt@pitt:~] grep "rc [0-6]" /etc/inittab
```

Output:

```
l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6
```

The entry that specifies a 2 in the runlevel field means that, when init(8) enters runlevel 2, the script /etc/init.d/rc is executed with a 2 passed as the only argument.

The /etc/init.d/rc script performs the crucial task of running all of the scripts in the directory /etc/rc2.d that begin with a capital "S." If we were entering runlevel 5, it would execute the scripts in /etc/rc5.d. A sample listing of these scripts is listed on the next page.

```
[matt@pitt:~] ls /etc/rc2.d/S*
```

Output:
```
/etc/rc2.d/S10sysklogd /etc/rc2.d/S20uml-utilities
/etc/rc2.d/S11klogd   /etc/rc2.d/S20xfs
/etc/rc2.d/S15usbmgr  /etc/rc2.d/S21aumix
/etc/rc2.d/S20alsa   /etc/rc2.d/S89atd
/etc/rc2.d/S20cupsys  /etc/rc2.d/S89cron
/etc/rc2.d/S20exim   /etc/rc2.d/S99kdm
/etc/rc2.d/S20makedev  /etc/rc2.d/S99rmnologin
/etc/rc2.d/S20pcmcia  /etc/rc2.d/S99stop-bootlogd
/etc/rc2.d/S20ssh    /etc/rc2.d/S99xdm
```

You can also start these services manually by running them with the start option, as follows:

```
[root@pitt:~] /etc/init.d/exim start
```

Other useful options are "stop," "restart" and "status." Most scripts will support all these options.

The numbers after the "S" tell the /etc/init.d/rc script the order in which to execute these startup scripts. The lower-numbered scripts are executed first. These scripts can perform numerous tasks, such as start a web or e-mail server, run a graphical login (some distributions run the graphical login right in /etc/inittab, though) or configure hardware.

RedHat places all of the /etc/rcN.d directories in /etc/rc.d. If you inspect /etc/rc2.d with ls, you will see that it is a link into /etc/rc.d:

```
[matt@pitt:~] ls -l /etc/rc2.d
```

Output:
```
lrwxrwxrwx 1 root root  10 Sep 18 10:20 /etc/rc2.d -> rc.d/rc2.d
```

Although the placement is confusing, the rc script is also in the location /etc/rc.d/rc. Inspect /etc/inittab on a RedHat system to see the differences. You must know both the RedHat and Debian layouts for the exam.

Managing Runlevel Services

The scripts in /etc/rcN.d are symbolic links back to an original script in the /etc/init.d directory. This arrangement ensures that users do not copy the same script into each runlevel. Numerous utility programs are available to help you manage these links; the programs include chkconfig, update-rc.d and ntsysv.

The startup scripts sitting in the /etc/init.d directory are referred to as init scripts.

Managing Runlevel Services with ntsysv

The program ntsysv is a graphical tool. If you run it without any arguments, you can configure your current runlevel. If you want to specify alternative runlevels to configure, run ntsysv with the –level option such as:

```
[root@pitt:~] ntsysv --level 1
```

to configure runlevel 1. Or:

```
[root@pitt:~] ntsysv --level 23
```

to configure multiple runlevels at the same time.

When ntsysv is run, you will be presented with a list of available services. Press the space bar to toggle the service on or off. An asterisk in the square brackets ([*]) indicates that the service is on. No asterisk means that the service will not execute for this runlevel.

Managing Runlevel Services with chkconfig

To list the services and their applicable runlevels with chkconfig, use the --list option. The following listing shows a sample of the output:

```
[root@pitt:~] chkconfig --list
```

Output:

```
pcmcia    0:off  1:off  2:on  3:on  4:on  5:on  6:off
nfs-common  0:off  1:off  2:off  3:on  4:on  5:on  6:off
xprint    0:off  1:off  2:off  3:on  4:on  5:on  6:off
setserial   0:off  1:off  2:off  3:off  4:off  5:off  6:off
```

or if you are interested in a specific service:

```
[root@pitt:~] chkconfig --list nfs-common
```

Output:

```
nfs-common  0:off  1:off  2:off  3:on  4:on  5:on  6:off
```

The output of this command shows the various services that are installed and lists the runlevels. Any runlevel with the word on after it indicates that there is a startup script in the appropriate /etc/rcN.d directory. The letter N represents a runlevel number.

To modify the runlevels that the /etc/init.d/rc script starts, use a command such as:

```
[root@pitt:~] chkconfig --level 345 nfs-common on
```

The options "on,""off" and "reset" are available.

The chkconfig program will also inspect startup scripts in /etc/init.d (or /etc/rc.d/init.d for older RedHat-based systems) for special comments to indicate default runlevels. If these comments are in the file and the suggested levels meet your requirements, you can add it to these runlevels with:

```
[root@pitt:~] chkconfig --add nfs-common
```

What Happened During the Boot Process

Certain Linux kernel and module log information is stored in what is called the kernel ring buffer. You may inspect this information with the command:

```
[matt@pitt:~] dmesg
```

However, there may be more information in the buffer than you can fit into one screen of text. More useful ways to use this command are to pipe it to pager commands or redirect the output to a file for later reading or e-mailing. Examples of these commands include:

```
[matt@pitt:~] dmesg | more
[matt@pitt:~] dmesg | less
[matt@pitt:~] dmesg > boot.messages
```

Linux kernel ring buffer information is also logged to /var/log/dmesg on many Linux distributions.

Another source of logging information is from the system logger (syslogd). The most useful syslogd file to look at in times like this is /var/log/messages.

LILO

The LILO (Linux Loader) boot loader is configured through the use of the /etc/lilo.conf file. This file is generally broken into two main sections; global and per-image options. Some per-image options are further separated depending on whether they are for a Linux kernel or an arbitrary system. Additionally, many of the per-image options may be used at the global level to indicate a default value.

A sample /etc/lilo.conf configuration file looks like the following:

```
# lilo.conf
#
# Global Options:
#
boot=/dev/hda
prompt
timeout=150
default=linux
lba32
vga=normal
root=/dev/hda1
read-only
#
#
```

Kernel Options (may have multiple):

```
#
image=/boot/vmlinuz-2.4.22
label=linux
append="mem=256M"
image=/boot/vmlinuz-2.4.23-experimental
label=test_upgrade
root=/dev/hda2
initrd=/boot/initrd-2.4.23-experimental
#
# Other Operating Systems Options (may have multiple):
#
other=/dev/hda3
label=dos
```

Essential Global Options for /etc/lilo.conf

Many options may be placed in /etc/lilo.conf. Only the most common ones are listed here.

Option	Description
boot=<device>	This setting specifies the name of the device that contains the boot sector. In this example, the first sector (or MBR) of the first IDE hard drive is used as the boot sector. If you wanted the LILO boot loader placed into a partition on /dev/hda, you would specify a specific partition such as /dev/hda3.
default=<label>	This setting specifies the default kernel image that will boot. If this option is omitted, the first image listed in /etc/lilo.conf will be used as the default.
timeout=<tsecs>	This setting specifies the amount of time (in tenths of a second) that LILO will wait for keyboard input before booting the default kernel image. You must use the prompt option to enable the timeout. A setting of 150 indicates a 15 second timeout.
prompt	This option instructs LILO to issue the boot: prompt and wait for user input.
lba32	This option enables LILO to boot from disks where the kernel image resides on a partition that is past the 1,024th cylinder.
vga=<value>	This selects the VGA text mode that is used when booting. Values for this setting are "normal," "extended," "ask" or a number.
read-only	This option indicates that the root file system should be mounted read-only. Usually, the operating system will remount the file system to read-write.

Essential Per-image Options for /etc/lilo.conf

The type of kernel that is about to be loaded is indicated as either a Linux kernel or another operating system. The image= option is used to indicate a Linux kernel and the other= option is used to indicate some other system.

Most lilo.conf files and examples have the options indented after a image= or other= setting. The indenting is done only to make the files more readable. All settings up to the next image= or other= setting are applicable to the current image.

Option	Description
image=<kernel>	Indicates the kernel image to use when booting.
other=<device>	Indicates a device path to an arbitrary operating system.
label=<title>	Provides a name for LILO to use. You can get a list of available labels when you are at the boot: prompt by hitting the tab key.

Do not remove old image= or other= configurations until you are certain that the new images are working. This precaution will allow you to boot to the old kernel image in case of failure. Another alternative is to install LILO with your new configuration onto a temporary device such as a floppy disk or a USB memory stick.

The remaining list of the per-image options are for Linux kernel images.

Option	**Description**
root=<device>	This is the root partition for the specified kernel image.
initrd=<ramdisk>	This is the initial RAM disk to use with the specified kernel image.
append="options"	This option is used to specify additional parameters that will be passed to the kernel. To tell the kernel that you have 256MB of RAM use append="mem=256M". More parameters may be added to this append line by separating them with spaces.

Installing LILO

Install the lilo boot loader with the lilo(8) command such as:

```
[matt@pitt:~] lilo
```

It is possible to install a new LILO boot loader and still have it fail when trying to use it. The lilo(8) man page describes the errors that may occur. LILO prints the letters 'L', 'I', 'L', and 'O' to indicate how it is progressing.

Interacting with the LILO Boot Loader

Sometimes you will need to boot with options that you did not place into the /etc/lilo.conf file. If you have configured LILO to provide a boot: (or sometime LILO:) prompt, you may pass extra options.

One thing that you may want to do is boot into single-user mode. Assuming that the image that you want has been labelled "linux," you would type something such as:

```
boot: linux 1
```

Your init program (usually /sbin/init) may be corrupt, missing or incorrectly configured. At such times, you may specify an alternate init program. A common technique is to use a regular shell program such as bash as your init. You would do so at the boot: promp, as follows:

```
boot: linux init=/bin/sh
```

Multiple parameters may be typed on the LILO boot prompt.

GRUB

The GRUB (Grand Unified Boot Loader) boot loader was the first loader to boot Linux from above the 1024th cylinder of a hard drive. GRUB has taken over as the default boot loader for many Linux distributions because it offers many features that LILO lacks. For example, you need not reinstall the GRUB boot loader after editing its configuration, and you have many more interactive options while booting.

Configuring the GRUB Boot Loader

The usual location for the GRUB boot loader's configuration file is /boot/grub/menu.lst. On some systems the configuration file is located at /boot/grub/grub.conf. GRUB can read its configuration file at boot time because it supports many different file systems.

A sample GRUB configuration file looks like the following:

```
# grub.conf
#
# Global Options:
#
default=0
timeout=10
#
#
```

Kernel Image Options:

```
#
title rh (2.6.5-1.358)
   root (hd0,3)
   kernel /boot/vmlinuz-2.6.5-1.358 ro
   initrd /boot/initrd-2.6.5-1.358.img
title debian
   kernel (hd0,4)/vmlinuz-2.4.22-halley.1 ro mem=96M
title xp
   rootnoverify (hd0,1)
   chainloader +1
```

Note that the GRUB boot loader does not refer to disk drives by letter the way that Linux does. GRUB numbers drives so that instead of /dev/hda, GRUB uses (hd0). Similarly, /dev/hdb would be (hd1). IDE drives are discovered and numbered before SCSI drives.

Additionally, GRUB numbers partitions on a drive starting at zero(0) instead of the one(1) that Linux uses. Putting the drive and partition numbering together provides a way to uniquely refer to partitions. To specify the first partition of the first hard drive, you would use:

```
(hd0,0) # Same as /dev/hda1
```

To refer to the first floppy device on your computer, you would use:

```
(fd0)   # Same as /dev/fd0 . Floppy disks have no partitions.
```

Essential Global Options for /boot/grub/grub.conf

The following table lists common global options used in a GRUB configuration file.

Option	Description
default=<value>	This option tells GRUB which operating system in the configuration file to boot as the default (GRUB indexes from 0). To boot the second listed operating system, use default=1.
timeout=<secs>	This option defines how long, in seconds, to wait for user input before booting the default operating system. A setting of 15 indicates a 15-second timeout.

The GRUB timeout option is in seconds while the LILO timeout option is in tenths (1/10) of a second.

Essential Per-image Options for /boot/grub/grub.conf

These are the most common options used in defining how to boot Linux and other operating systems.

Option	Description
title <label>	This setting specifies the label to list when the boot loader runs. It also starts a per-image configuration section.
root <device>	This option specifies the location of GRUB's root partition. This would be the /boot partition if a separate one exists. If one does not exist, it is the Linux system's root partition. Specify the device in GRUB syntax such as (hd0,3).
kernel <kernel and options>	This setting describes the location of the Linux kernel as well as any kernel options that are to be passed. Paths are relative to GRUB's "root" partition. As an alternative, you can specify devices using GRUB's syntax such as "kernel (hd0,5)/vmlinuz ro."

Option	Description
initrd <ramdisk>	Use this option to specify an initial RAM disk.
rootnoverify <device>	This option is similar to the root option except that GRUB will not try to access files on this partition. Specify the device in GRUB syntax such as (hd0,3).
chainloader +1	This option tells GRUB to load the first sector of the root partition (usually specified with rootnoverify) and to transfer execution to this secondary boot loader.

Installing the GRUB Boot Loader

Installation of the GRUB boot loader differs slightly from installation for the LILO boot loader. First, the command for installing GRUB is grub-install. Second, you have to specify the boot sector by device name when you install the boot loader. The basic command looks like the following:

```
[root@pitt:~] grub-install /dev/hda
              or:
[root@pitt:~] grub-install '(hd0)'
```

This command will install the GRUB boot loader into the first sector (or the MBR) of your first hard drive. In the second example, you need quotes around the device name. To indicate that you want the first sector of the hard drive and not the first sector of the first partition, omit the partition number.

Interacting with the GRUB Boot Loader

The first screen the GRUB boot loader shows you is a list of all the operating systems you specified with the title option in your GRUB configuration file.

You can wait for the timeout to expire for the default operating system to boot. To select an alternative, you may highlight the operating system that you want to have boot by using the arrow keys. Once your choice is highlighted, hit the "enter" key to start booting.

Passing Kernel Options with GRUB

Follow these steps when you want to change or pass additional options to your operating system:

1. Use the arrow keys to highlight the operating system that most closely matches what you want to boot.
2. Press the "e" key to edit this entry. You will now see a new screen listing all the options for this entry.
3. Use the arrow keys to highlight the kernel option line.
4. Press the "e" key to edit the kernel options.
5. Edit the kernel line to add a 1 or single on the end. GRUB will pass the extra option to the kernel and boot into single-user mode.
6. Press the "enter" key to complete the edits.
7. Press the "b" key to start booting.

Once this sequence is working, repeat the sequence, but change Step 5 to use a different init program. Try appending init=/bin/sh to the end of the kernel line.

1.106.2 Change runlevels and shutdown or reboot system (Weight: 3)

Candidates should be able to manage the system's runlevel. This objective includes changing to single-user mode, shutting down or rebooting the system. Candidates should be able to alert users before switching runlevel and properly terminate processes. This objective also includes setting the default runlevel.

Set the default runlevel

To set the default run level, change the numeric value in /etc/inittab on the line that looks like the following:

```
id:2:initdefault:
```

Values between 1 and 5 are the most useful. A standard exists for the meaning of each runlevel defined by the LSB (Linux Standard Base) but not every distribution follows the standard.

The following table provides a list of the most commonly used runlevels:

Level	Purpose	Description
0	System Halt	The system will do a complete and proper shutdown of all services. If possible, the system will also be powered down.
1	Single User/ Maintenance Mode	Minimal system facilities will start. On Debian systems, this level includes networking and some core services. On Red Hat systems, networking and core services are not included, and the system will have only a simple shell available. No users may log in at this run level.
2	Variable	Undefined on Red Hat but default runlevel for Debian systems.
3	Variable	Full multiple-user system on Red Hat with text or console logins. User configurable for Debian systems.
4	Undefined	User-configurable runlevel.
5	Variable	On Red Hat systems, this is the full multiple-user system with graphical login and a GUI for the user. User configurable for Debian systems.
6	Reboot	The system will do a complete and proper shutdown and then reboot.

Change between run levels

Use the runlevel command to determine your current run level. The output will look something like the following:

```
[matt@pitt:~] runlevel
```

Output:
N 5

The first character indicates the previous runlevel. An "N" indicates that the system booted directly into a runlevel and has not been changed since boot time.

The second character is the current runlevel (in this example, that is runlevel 5).

To change to another runlevel, use the init(8) or telinit(8) programs. For example, to switch to runlevel 3, use the command:

```
[root@pitt:~] init 3
```

This command will send a message to the currently running init process (PID 1), which will subsequently reread the /etc/inittab file, remove any processes that are no longer required for the runlevel and call the /etc/init.d/rc script with the new runlevel to add or remove even more services.

Shutdown and reboot from the command line

The shutdown(8) command is the most versatile command for shutting down the system. A sample use of the command would look like the following:

```
[root@pitt:~] shutdown -h 19:00 "system coming down for a new disk"
```

This example would switch the system to runlevel 0 (halt) at 7 p.m. that night (or the next night if the command is executed later than 7 p.m.) and send the message "system coming down for a new disk" to all logged-in users. Further login attempts will also be blocked.

Shutdown performs its functions by calling init(8) for the runlevel change.

The time may be given in the hh:mm format (hh is hours and mm is minutes on a 24-hour clock), in the format +m (m is the number of minutes to wait) or the word "man" (an alias for +0).

Other useful options for shutdown are:

Option	Description
-t sec	Tell init(8) to wait sec seconds between sending processes the warning and the kill signal, before changing to another runlevel.
-r	Reboot after shutdown (default is to switch to runlevel 1).
-h	Halt or power off after shutdown.
-c	Cancel an already-running shutdown. With this option, the time argument is unavailable, but you can enter a explanatory message on the command line that will be sent to all users.

Alert users before switching runlevels or other major system event

There are other ways to notify users of pending system changes. They include the commands:

Command	Description
write	Send a message to another user.
wall	Write a message to all users who are currently logged in.
talk	Set up a two-way chat with another user. Half the screen will be used to display what one user types and the other half of the screen is for the second user.
mesg	While not strictly a messaging program, mesg controls access to your terminal from users trying to send you messages.

Use:

```
[matt@pitt:~] mesg n
```

to disallow messages (from non-root users) and:

```
[matt@pitt:~] mesg y
```

to allow messages.

Other commands for shutting down the system

Possible commands include halt, reboot and poweroff. If these command are run while the system is _not_ in runlevel 0 or 6, shutdown will be invoked with the halt (-h) or reboot (-r) flag.

Topic 107: Printing

1.107.2 Manage printers and print queues (Weight: 1)

Candidates should be able to manage print queues and user print jobs.

A print queue is a spool area for files being printed.

A print job is an instance of a file in the print queue.

Two printing systems are in common use, LPD and CUPS. CUPS is the newer, better technology but there is a compatibility layer where the LPD commands are supported.

The LPD Commands

LPD commands come in two flavors, BSD and SysV. In the table below, the SysV versions are in parentheses. You should completely understand the BSD commands.

Command	Purpose
lpr (lp)	Used to submit files to a print queue
lprm (cancel)	Used to cancel print jobs
lpq (lpstat)	Displays the print jobs in a queue

The most commonly used option to these commands is "-P destination," which indicates the named print queue that will receive the print job (lpr), cancel request (lprm) or queue query (lprm).

The default print queue is "lp" but this default can be changed with the PRINTER environment variable.

The Line Printer Control Program

The lpc program is used to control the behavior and query the state of print jobs. Lpc provides an interactive shell to type commands. A few of these commands are (from the BSD lpc(8) man page):

? [command ...]
help [command ...]

Print a short description of each command specified in the argument list, or, if no argument is given, a list of the recognized commands.

disable { all | printer }

Turn the specified printer queues off. This prevents new printer jobs from being entered into the queue by lpr(1).

down { all | printer } message [...]

Turn the specified printer queue off, disable printing and put message in the printer status file. The message need not be quoted; the remaining arguments are treated like echo(1). This is normally used to take a printer down and let users know why. lpq(1) will indicate the printer is down and print the status message.

enable { all | printer }

Enable spooling on the local queue for the listed printers. This will allow lpr(1) to put new jobs in the spool queue.

exit
quit

Exit from lpc.

start { all | printer }

Enable printing and start a spooling daemon for the listed printers,

status { all | printer }

Display the status of daemons and queues on the local machine.

stop { all | printer }

Stop a spooling daemon after the current job completes and disable printing.

Please note, if CUPS is installed the lpc(8), man page will document fewer commands and direct the administrator to use the lpadmin(8) command, which offers many more options and features.

1.107.3 Print files (Weight: 1)

Candidates should be able to manage print queues and manipulate print jobs.

Sample uses of the lpr command

You can simply print files on the command line with:

[matt@pitt:~] lpr file1.ps file2.pdf

and the filters will convert the files to an appropriate printer-specific format.

Or you can have other commands create the content and pipe the output to the lpr command, such as:

[matt@pitt:~] cut -d: -f1,6 /etc/passwd | lpr

Additionally, certain utilities will format text documents to look and print more cleanly. They are:

The a2ps and enscript Commands
Both of these utilities can convert text to PostScript. Features include multicolumn page generation, landscape or portrait printing, number of copies to print, and header and footer information.

The pr and fmt Commands
These utilities reformat plain text to easier-to-read style. These tools can paginate by a supplied page width, add header information, choose different page lengths and margins and more.

1.107.4 Install and configure local and remote printers (Weight: 1)

Candidates should be able to install and configure local and remote printers.

For print queues to work, the lpd(8) daemon must be running.

Refer to the printcap(5) man page for more details. For details on the formatting for /etc/printcap, please refer to the termcap(5) man page.

Entries in /etc/printcap are a single logical line with a '\' used at the end of each line to indicate a continuation to the next real line.

The first field in a printcap entry contains a list of the names by which the print queue can be called. This list is separated by '|'. The following fields define the behavior of this queue and can take the following forms:

Type	Example
Boolean values	:sh:\
Numeric values	:mx#0:\
String values	:lp=/dev/lp0:\

Commonly used fields are:

Name	Type	Default	Description
lp	str	/dev/lp	local printer device, or port@host for remote
sd	str	/var/spool/lpd	spool directory
af	str	NULL	name of accounting file
lf	str	/dev/console	error logging file name
mx	num	1000	max file size (in BUFSIZ blocks); 0=unlimited
sh	bool	false	suppress printing of burst page header
pc	num	200	price per foot or page in hundredths of cents
pl	num	66	page length (in lines)

Name	Type	Default	Description
pw	num	132	page width (in characters)
rm	str	NULL	machine name for remote printer
rp	str	lp	remote printer name argument

A sample /etc/printcap entry for local printers

lp|Generic dot-matrix printer entry:\

```
:lp=/dev/lp0:\
:sd=/var/spool/lpd/lp:\
:af=/var/log/lp-acct:\
:lf=/var/log/lp-errs:\
:pl#66:\
:pw#80:\
:pc#150:\
:mx#0:\
:sh:
```

This example defines the default "lp" print queue. The physical device can be accessed through the /dev/lp0 device file (lp), the spool directory is /var/spool/lpd/lp (sd), there is no maximum file size that may be printed (mx), no burst page will be printed before each job (sh), and accounting and errors are logged to /var/log/lp-acct (af) and /var/log/lp-errs (lf), respectively.

Additionally, the printer page is 66 lines (pl) by 80 characters (pw) in size and printing costs are $0.015 per foot (pc).

/var/spool/lpd/ is the default location for the queue spool directories.

A sample /etc/printcap entry for remote printers

rlp|Remote printer entry:\

```
:lp=:\
:rm=remotehost:\
:rp=remoteprinter:\
:sd=/var/spool/lpd/remote:\
:mx#0:\
:sh:
```

This example defines a remote printer called "rlp." The physical device is left unset (lp) and the remote host and remote print queue are "remotehost" (rm) and "remoteprinter" (rp), respectively.

There is no maximum size that will be accepted (mx), although, the remote printer may have its own limits. Headers will be suppressed (sh) and print jobs will spool locally in /var/spool/lpd/remote until they can be sent to the remote host.

Configuring CUPS printers

CUPS is much more complex to configure. Only the basics are covered here. Normally all configuration of the CUPS printing system is done via the URL and web application at:

http://localhost:631/

This service is provided by the main CUPS server process, cupsd(8).

Important files for CUPS are:

/etc/cups/cupsd.conf	The configuration file for the CUPS scheduler, cupsd(8).
/etc/cups/classes.conf	Defines local printer classes that are available and is automatically generated by the cupsd(8) daemon when printer classes are added and removed.
/etc/cups/client.conf	Configuration for cups client programs. May also be personalized by placing in a ~/.cups directory for each user.
/var/spool/cups/	The default spool directory for print jobs.

Topic 108: Documentation

1.108.1 Use and manage local system documentation (Weight: 4)

Candidates should be able to use and administer the man facility and the material in /usr/share/doc/.

The man Command

Man pages are split into multiple categories or sections, as follows:

Section	Description
1	Executable programs or shell commands
2	System calls (functions provided by the kernel)
3	Library calls (functions within program libraries)
4	Special files (usually found in /dev)
5	File formats and conventions eg /etc/passwd
6	Games
7	Miscellaneous (including macro packages and conventions)
8	System administration commands (usually only for root)
9	Kernel routines [Non standard]

To read a man page, use the man command with the name of the command or topic that you want to learn about. For example, to read the man page on the man command, type:

```
[matt@pitt:~] man man
```

To read about the ls command, type:

```
[matt@pitt:~] man ls
```

A Typical Man Page Structure

NAME	The name of the command or topic along with a short (one line) description of the command.
SYNOPSIS	A brief description of how the command is used. Anything in square brackets ([]) is an optional part of the command structure.
DESCRIPTION	An English-language description of how the command behaves or what the topic is about.
OPTIONS	A summary of the options listed in the SYNOPSIS with a description of the option's purpose.
FILES	A list of the files associated with this command or topic. Typically, it refers to configuration files.
SEE ALSO	A list of man pages related to this command or topic. They are generally listed with their section. For example, ls(1) refers to the man page for the ls command in section 1.
BUGS	A list of any known bugs.
AUTHOR	Information about the author of the program or topic.

Dealing with Multiple Pages on a Topic
To specify a section to look in for the man page, specify the section number before the command or topic of interest. For example, the crontab command has two man pages. One in section 1 and one in section 5. By default, section 1 will be presented. To specify the section 5 man page, type:

```
[matt@pitt:~] man 5 crontab
```

To view all man pages on a topic such as crontab, in order, type:

```
[matt@pitt:~] man -a crontab
```

You may need to hit the "q" key to quit the pager program that is displaying the current man page.

Addition Manual Commands
The whatis command displays man page descriptions. For example, whatis man will yield the following output:

```
[matt@pitt:~] whatis man
```

Output:
```
man (1)      - an interface to the on-line reference manuals
man (7)      - macros to format man pages
```

The "-k" option to the man command will give the same information.

The apropos command is a more thorough search than whatis. It will search through the manual page names and descriptions. For example, apropos crontab will yield the following output:

```
[matt@pitt:~] apropos crontab
```

Output:
```
/etc/anacrontab (5) [anacrontab] - configuration file for anacron
anacrontab (5)    - configuration file for anacron
crontab (1)     - maintain crontab files for individual users (V3)
crontab (5)     - tables for driving cron
```

The "-f" option to the man command will give the same information.

Configuring the Manual System
The man configuration file is /etc/man.conf. Some Linux distributions also name the configuration file /etc/man.config or /etc/manpath.config.

To change the directories in which the man command will search for man pages, adjust the MANPATH entries in the configuration file. A typical configuration file will have entries such as:

```
MANPATH /usr/share/man
MANPATH /usr/X11R6/man
MANPATH /usr/local/man
MANPATH /usr/kerberos/man
MANPATH /usr/man
```

To include more search directories, just add another MANPATH line to the configuration file.

To include additional sections beyond the numbered ones (most distributions do), adjust the MANSECT line in the configuration file. For example, the following MANSECT entry adds a tcl, n, l, p and o section to the search:

```
MANSECT    1:8:2:3:4:5:6:7:9:tcl:n:l:p:o
```

NOTE: On Debian-based systems, the system is configured somewhat differently.

The info Command
An alternative documentation system to man pages is called info pages. Info pages use a custom reader for the documentation (as opposed to man using only the default pager). To learn how to use the info command, type:

```
[matt@pitt:~] info info
```

Other System Documentation

Package documentation that is not in the man or info formats are placed in the directory /use/share/doc. Package documentation is in the subdirectory <package>-<version> or <package>. Some documentation is also placed in /usr/doc.

1.108.2 Find Linux documentation on the Internet (Weight: 3)

Candidates should be able to find and use Linux documentation on the Internet.

This objective includes using Linux documentation at sources such as the Linux Documentation Project (LDP), vendor and third-party Web sites, newsgroups, newsgroup archives and mailing lists.

The home of the LDP is http://www.tldp.org. It is the official home of guides or tutorials on how to set up certain technologies called HOWTOs. Many other guides, FAQs (Frequently Asked Questions) and mini-HOWTOs are also available.

1.108.5 Notify users on system-related issues (Weight: 1)

Candidates should be able to notify the users about current issues related to the system.

This requirement means being familiar with the various files that contain messages as users log in.

File	Description
/etc/issue	Contains a message that is displayed above the login: prompt for local logins.
/etc/issue.net	Contains a message that is displayed above the login: prompt for remote logins (such as telnet). SSH doesn't use this file, though.
/etc/motd	The "message of the day" which is displayed after a successful login but before your shell is executed.

Topic 109: Shells, Scripting, Programming and Compiling

1.109.1 Customize and use the shell environment (Weight: 5)

Candidates should be able to customize shell environments to meet users' needs.

More topics relating to shells and their use are found in the LPI 101 exam.

Setting Variables

A variable is a value stored by your shell (or programming language) and is referred to by a name.

The easiest way to declare a variable is to assign a value to it, such as:

```
[matt@pitt:~] MYVAR=SomeValue
```

To determine the value of the MYVAR variable, use the $ symbol before the variable name (such as $MYVAR). To display the value of the MYVAR variable, use the echo command:

```
[matt@pitt:~] echo $MYVAR {SomeValue}
```

To assign the value of the MYVAR variable to another variable, type:

```
[matt@pitt:~] YOURVAR=$MYVAR
```

Variables are case-sensitive in Linux. MYVAR, MyVar and myvar are three different variables.

Variable Substitution

You will frequently need to do more than simple variable assignment, especially when you want to use spaces in your variable values or the output from some command.

This task can be accomplished with one of three special quoting characters:

double quotation marks ("), the single quotation mark (') and the back tick (`).

Use double quotation marks (") around the value if you want variable substitution to occur in the value. For example:

```
[matt@pitt:~] MYVAR=SomeValue
[matt@pitt:~] YOURVAR="MYVAR is $MYVAR" {MYVAR is SomeValue}
[matt@pitt:~] echo $YOURVAR
```

If you want to prevent the shell from treating a character as special, escape it with a '\'. For example:

```
[matt@pitt:~] MYVAR=SomeValue
[matt@pitt:~] YOURVAR="MYVAR is \$MYVAR" {MYVAR is $MYVAR}
[matt@pitt:~] echo $YOURVAR
```

It is sometimes hard to tell where a variable name ends and regular text starts. To avoid this ambiguity, use curly braces ({ and }) around the variable name. Consider the difference between these two assignments:

```
[matt@pitt:~] MYVAR=SomeValue
[matt@pitt:~] YOURVAR="the long string is '$MYVAROnly'"
[matt@pitt:~] echo $YOURVAR

[matt@pitt:~] YOURVAR="the long string is '${MYVAR}Only'"
[matt@pitt:~] echo $YOURVAR
```

The first echo command does not produce the expected output because the shell is looking for the non-existent variable MYVAROnly. Nonexistent variables are replaced with an empty string.

Use single quotation marks (') around the value if you want to prevent variable substitution. For example:

```
[matt@pitt:~] MYVAR=SomeValue
[matt@pitt:~] YOURVAR='MYVAR is $MYVAR' {MYVAR is $MYVAR}
[matt@pitt:~] echo $YOURVAR
```

This is exactly what you set in the variable.

Also, you can use the output of a command as the value of a variable. For example:

```
[matt@pitt:~] MYVAR=`date`    {Fri Jul 14 19:29:16 EDT 2006}
[matt@pitt:~] echo $MYVAR
```

Output:
Fri Jul 14 19:29:16 EDT 2006

You can mix and match special quoting characters. For example:

```
[matt@pitt:~] MYVAR=really
[matt@pitt:~] YOURVAR="I am '$MYVAR' happy that today is '`date`'"
[matt@pitt:~] echo $YOURVAR
```

Output:
I am 'really' happy that today is 'Fri Jul 14 19:31:00 EDT 2006'

In this example, the double quotation marks indicate that variable and command substitution will still occur (the $MYVAR and `date`, respectively). The single quotation marks are regular single quotation marks on a standard keyboard.

The bash shell also provides an alternative syntax to the back ticks (`). The following two commands are equivalent:

```
[matt@pitt:~] MYVAR=`date +%s`
[matt@pitt:~] MYVAR=$(date +%s)
```

Exporting Variables

Normal variables are accessible only to the shell instance that created them. However, if you need a variable to be available to the scripts and programs that your shell executes, you must use the export command to indicate that the variable should be passed to the child process that you execute.

To export an existing variable, use a command such as:

```
[matt@pitt:~] export MYVAR
```

Also, you can do the variable assignment and export in one command, if that is more convenient. For example:

```
[matt@pitt:~] export MYVAR=SomeValue
```

You may also export (and assign) more than one value at a time:

```
[matt@pitt:~] export MYVAR=SomeValue YOURVAR=SomeOtherValue
```

As the root user, try this experiment:

```
[root@pitt:~] vipw
```

exit the editor (it will probably be the vi editor)

```
[root@pitt:~] export EDITOR=emacs
[root@pitt:~] vipw
```

exit the emacs editor

Finally, you may also assign a temporary variable which will be passed to the child process but not be set permanently in your shell. For example:

```
[matt@pitt:~] MYVAR=SomeValue some_command
```

The "some_command" will have the MYVAR environment variable passed to it.

Inspecting Variables

To see a list of all your shell variables, use the set command:

```
[matt@pitt:~] set
<snip>
OPTIND=1
OSTYPE=linux-gnu
P4CONFIG=.p4settings.
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
<snip>
```

The set command will also list functions that you may have defined. Also, the set command has many other uses. Refer to the bash(1) man page for more details.

If you want to limit the listing to only exported (or environment) variables, use the env command:

```
[matt@pitt:~] env
<snip>
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
LANG=en_CA.UTF-8
USER=matt
HOME=/home/matt
<snip>
```

Important environment variables include the following:

Variable	Description
PATH	A colon (:)-separated list of directories to search for programs when you try to execute something. If the program is not in any of those directories, you will need to provide a full path on the command line.
USER	Your user name.
LANG	You language choice.
HOME	The location of your home directory.

Removing Variables

There are two approaches to cleaning up variables that you no longer want.

One method is to assign an empty or "NULL" value to the existing variable, as follows:

```
[matt@pitt:~] MYVAR=
```

The variable will still exist but the value is an empty string.

To completely remove a variable, including its name, from your shell, use the unset command:

```
[matt@pitt:~] unset MYVAR
[matt@pitt:~] set | grep MYVAR
```

Using Shell Aliases

An alias is a macro or shorthand for a longer command.

Create an alias with:

```
[matt@pitt:~] alias ll='ls -l'
```

Now you can use it as follows:

```
[matt@pitt:~] ll *.pdf -rw-rw-rw- 1 matt matt 277704 2005-04-12 00:27 bootitng.pdf
```

To get a list of your aliases, use the alias command with no arguments:

```
[matt@pitt:~] alias
alias ls='ls --color=auto'
alias ll='ls -l'
alias rm='rm -i'
<snip>
```

To temporarily ignore an alias, use a '\' before the command:

```
[matt@pitt:~] \ll *.pdf
```

Output:
bash: ll: command not found

To unset or remove an alias, use the unalias command:

```
[matt@pitt:~] unalias ll
```

Using Shell Functions

Shell functions are just like functions in other programming languages. Parameters can be passed, multiple statements may be executed, other functions may be called and values can be returned.

To define a function, enter something comparable to the following:

```
[matt@pitt:~] function my_top()

{
        uptime
        free
        ps axuw | head -10
}
```

And use it as you would any regular program or alias:

```
[matt@pitt:~] my_top
13:38:47 up 23:56, 4 users, load average: 0.00, 0.07, 0.09
     total   used   free  shared  buffers   cached
Mem:   515364   507648   7716    0   48140   253568
-/+ buffers/cache:  205940   309424
Swap:     0    0    0
USER    PID %CPU %MEM  VSZ  RSS TTY   STAT START  TIME COMMAND
root    1 0.0 0.1  1564  528 ?    S  Jul21  0:02 init [2]
root    4 0.0 0.0   0   0 ?    S<  Jul21  0:00 [events/0]
root    5 0.0 0.0   0   0 ?    S<  Jul21  0:00 [khelper]
root    6 0.0 0.0   0   0 ?    S<  Jul21  0:00 [kthread]
root    8 0.0 0.0   0   0 ?    S<  Jul21  0:00 [kblockd/0]
root    9 0.0 0.0   0   0 ?    S<  Jul21  0:00 [kacpid]
root   132 0.0 0.0   0   0 ?    S  Jul21  0:00 [pdflush]
```

The "function" keyword is optional and, typically, a "library" of these functions can be written and read into your script or shell with the source or . commands:

```
[matt@pitt:~] . ~/shlib/mylib.sh
              or
[matt@pitt:~] source ~/shlib/mylib.sh
```

Use the unset command to remove a function.

Precedence of Execution

The order of execution precedence for the bash shell is:

- Aliases - The shell will replace the alias with the alias value.
- Keywords - Such as if, for, while, while, until and function.
- Functions.
- Built-in Commands - To save execution time, many useful commands arecompiled into the shell interpreter such as alias, cd, echo. Confirm this compilation with the type command.
- Executables - Finally the PATH environment variable is checked to locate the command that is needed.

BASH configuration files

When you start a shell, some configuration files may be read, depending on the situation.

Configuration File	Purpose
/etc/profile	Global settings and commands for brand new logins.
~/.bash_profile ~/.profile ~/.bash_login	Individual settings for brand new login shells. The first one found in this order will be read.
~/.bash_logout	This file is read and executed when a login shell exits.
/etc/bash.bashrc	Global settings for new interactive shells (non-login).
~/.bashrc	Individual settings for new interactive shells (non-login).
~/.inputrc	Configuration file for how interactive shells read input and provides command line editing.

For shell scripts, none of these files are read.

1.109.2 Customize or write simple scripts (Weight: 3)

Candidates should be able to customize existing scripts or write simple new BASH scripts.

A Simple Shell Script

A simple script looks like:

```
[matt@pitt:~] cat simple.sh
```

Output:

```
#!/bin/sh
# My 'hello world' shell script.
echo "hello $1"
```

The first line indicates to the kernel which shell interpreter to execute. That shell will read in this script and execute the commands.

The second line is a comment. Comments start with a "#" anywhere on the line, but not in quoted strings, and comment out the rest of the line.

The third line uses the echo command to print the string 'hello' followed by whatever (if anything) is the first argument to the script.

Executing the Script

To execute or run the simple.sh script, you can run a bash shell and give it the name of the script as an argument. Any further parameters will be passed to the shell script.

```
[matt@pitt:~] sh simple.sh world
```

Output:
hello world

Alternatively, you can turn on the read and execute permissions for the script and run it as you would any other script or program, as follows:

```
[matt@pitt:~] chmod +rx simple.sh
[matt@pitt:~] ./simple.sh world
```

Output:
hello world

We needed the ./ path information to find the simple.sh script because the home directory is not in the PATH environment variable.

Using Parameters in your Script

All parameters are numbered. To use a parameter higher than $9, you must use curly braces ({ and }) around the number. For example:

```
[matt@pitt:~] cat simple2.sh
```

Output:

```
#!/bin/sh
echo $1
echo ${10}
echo $11
[matt@pitt:~] ./simple2.sh a b c d e f g h i j k
a
j
a1
```

You can also use the shift command to rename higher-numbered parameters to lower-numbered ones. The lower-numbered parameters will disappear, so make sure they are no longer needed. With no numeric value, everything will be shifted by one position.

[matt@pitt:~] cat simple3.sh

```
Output:
#!/bin/sh
echo $1; shift 9
echo $1; shift
echo $1
[matt@pitt:~] ./simple3.sh a b c d e f g h i j k l
a
j
k
```

Other useful variables in parameter passing and shell scripting are:

Variable	Description
$#	Total number of positional parameters.
$* $@	All the current position parameters. For details on how these two variables differ in behaviour, consult the bash(1) man page.
$?	The exit value of the most recent foreground command.
$0	The name of the shell or shell script.
$$	The PID of the current shell.

The read Command

Users can also provide input with the read command. For example:

[matt@pitt:~] cat simple4.sh

```
Output:
#!/bin/sh
echo "provide two numbers and a sentence"
read var1 var2 theRest
echo $theRest: $(($var1 + $var2))
[matt@pitt:~] ./simple4
provide two numbers and a sentence
34 56 my age plus my IQ
my age plus my IQ: 90
```

In this script, the line "provide two numbers and a sentence" is printed to the screen. The user provides input and the two numbers are assigned to the first two variables (var1 and var2). Any remaining input is assigned to the last variable. If there are more variables than values, the remaining variables will be left unassigned.

The $((...)) construct is provided by bash for arithmetic calculations. More detail is provided in the bash (1) man page.

The test Command

The test command is used to compare file information (such as size, timestamps, type and permission), string and numeric comparison.

Some available comparisons are:

Logical groupings	EXPR1 -a EXPR2	EXPR1 and EXPR2 are both true
	EXPR1 -o EXPR2	EXPR1 or EXPR2 is true
String operators	-n STR	the length of STR is nonzero
	STR1 = STR2	the strings are equal
	STR1 != STR2	the strings are not equal
Numeric operators	INT1 -eq INT2	INT1 is equal to INT2
	INT1 -ge INT2	INT1 is greater than or equal to INT2
	INT1 -lt INT2	INT1 is less than INT2
	INT1 -ne INT2	INT1 is not equal to INT2
File operators	FILE1 -nt FILE2	FILE1 is newer (modification date) than FILE2
	-e FILE	FILE exists
	-f FILE	FILE exists and is a regular file
	-r FILE	FILE exists and read permission is granted
	-s FILE	FILE exists and has a size greater than zero
	-w FILE	FILE exists and write permission is granted
	-x FILE	FILE exists and execute (or search) permission is granted

The '[' command is an alternative form of calling the test command. In this form, the command line must end with a ']' as the final argument.

The test command will exit with zero (0) to indicate a true value (just as 0 means success for program exit values). A non-zero value indicates that the expression being evaluated was false.

The $? variable can be inspected to determine the exit value of test. For example:

```
[matt@pitt:~] test -d /no/such/path; echo $?
```

Output:

```
1
[matt@pitt:~] [ -d /tmp ] ; echo $?
0
```

Scripting Control Constructs The if/elif/else/fi Statements

If statements allow for conditional execution based on the truthfulness of the value being tested. The arguments to the if family of statements is a command (including test or []). If the exit value is zero (0), the condition is considered true. Otherwise, the condition is false.

For example:

```
[matt@pitt:~] if test -f /tmp
then
        echo /tmp exists
elif [ -x /sbin/init ]
then
        echo /sbin/init is executable
else
        echo no tests passed
fi
```

produces the output:

```
/sbin/init is executable
```

Only the first condition to evaluate to true will be executed.

Some script writers prefer to keep the "then" keywords on the same line as the if/elif keywords. You can keep them on the same line by ending the if/elif statements with a semi-colon, such as:

```
[matt@pitt:~] if [ -x /sbin/init ]; then
        echo should we execute it?
        echo why? it is already running
```

The for Loop

The for loop is used to iterate over a list of values. A simple usage is:

```
[matt@pitt:~] for i in 1 2 3 4 5; do
        echo "i is $i"
done
```

which produces the output:

```
i is 1
i is 2
i is 3
i is 4
i is 5
```

As with the if statement, you can put the do keyword on the next line, such as:

```
[matt@pitt:~] for i in 1 2 3 4 5
do
        echo "i is $i"
done
```

You can also use the output of a command as the list of values. For example:

```
[matt@pitt:~] for i in $(seq 1 2 9); do
        echo "i is $i"
done
```

produces the output:

```
i is 1
i is 3
i is 5
i is 7
i is 9
```

With no list given, the for loop will iterate over all of the scripts positional parameters. For example:

```
[matt@pitt:~] cat simple5.sh
#!/bin/sh
for i; do
        echo "i is $i"
done

[matt@pitt:~] ./simple5.sh a b c d e
```

Output:

```
i is a
i is b
i is c
i is d
i is e
```

The while/until Loop

The while/until loops are provided to continue looping around a set of commands until the tested condition becomes true (an exit value of 0) in the case of the while loop, or false (an exit value of non-zero) in the case of the until loop.

For example:

```
[matt@pitt:~] i=0
[matt@pitt:~] while [ $i -lt 5 ]; do
        echo "i is $i"
        i=`expr $i + 1`
done
```

produces the output:

```
i is 0
i is 1
i is 2
i is 3
i is 4
```

The expr command is the old-timer alternative to the $((...)) construct. Because expr is an external command, scripts will execute somewhat more slowly. However, the scripts will also be more portable to other Unixes with no bash shell.

A similar example with the until loop is:

```
[matt@pitt:~] i=0
[matt@pitt:~] until [ $i -ge 5 ]; do
        echo "i is $i"
        i=$(($i + 1))
done
```

which will produce identical output as the while loop example.

Topic 111: Administrative Tasks

1.111.1 Manage users and group accounts and related system files (Weight: 4)

Candidates should be able to add, remove, suspend and change user accounts.

User accounts are assigned a User ID (UID) and one primary Group ID (GID) and any number of secondary Group IDs. UIDs and GIDs are numeric values from 0 to 65,536.

The /etc/passwd File

The /etc/passwd file is the sole file that links user names and the UID (User ID) number. Each line in the /etc/passwd file is a user account and the format for the line looks like:

```
[matt@pitt:~] grep matt /etc/passwd
```

Output:
matt:x:1000:1000:G. Matthew Rice,,,:/home/matt:/bin/bash

The file contains seven fields separated by colons. The fields, in order, are:

Field Name	Description
login name	The account name.
optional encrypted password	This field may contain an encrypted password but, with /etc/passwd needing to be world readable so normal users can match names to UIDs, this is a security hole. More commonly, the encrypted password is stored in /etc/shadow which is not world readable. An "x" indicates that the password is stored in /etc/shadow.
numerical user ID	The user's numeric UID. This is the number stored in files, for example, to indicate ownership.
numerical group ID	The user's numeric GID. This is the user's default group ID. Additional groups can be assigned to the user via the /etc/group file.
user name or comment field	Also, known as the GECOS field (General Electric Common Operating System). Anything can be placed here but it usually includes the user's full name.
user home directory	The user's home directory. Typically, it is under /home/<username>. If this field is left blank, the user's current directory will be / upon login.
optional user command interpreter	The shell field. If left blank, the user will be given /bin/sh as the shell. It can be any executable program. For example, a mail-only user (no local login) could be assigned /bin/false as the shell.

Account information can be provided from alternate services such as NIS, LDAP, SQL databases, etc. However, doing so is beyond the scope of these exams.

The /etc/shadow File

The /etc/shadow file is used as a secure location for storing encrypted passwords as well as account access controls. A typical line in /etc/shadow looks like:

```
[root@pitt:~] grep matt /etc/shadow
```

Output:
matt:1JG8aDfRgtTgrkkKR/tm7bbJgA7RyBm/:13346:0:99999:7:::

As with the /etc/passwd file (and many others), the fields are separated with colons (:). The fields, in order, are:

Field	Description
login name	This is the user name for the line and ties back to the user with the same name in the /etc/passwd file.
encrypted password	The encrypted password.
days since Jan 1, 1970 that password was last changed	This field is given as an integer number of days.
days before password may be changed	Based on the number of days since the last change, users may be prevented from changing their password. A zero (0) value means that the password may be changed as often as desired.
days after which password must be changed	
days before password is to expire that user is warned	
days after password expires that account is disabled	
days since Jan 1, 1970 that account is disabled	Regardless of password changes, if this field is set and the number of days are reached, the account will be disabled.
a reserved field	

This file should be readable only by the root user (UID 0) to maintain security.

The /etc/group File

The /etc/group file is used to map group names to Group IDs (GIDs). It is also used to assign additional users to a group. A typical line in /etc/group looks like:

```
[matt@pitt:~] grep matt /etc/group
```

Output:
```
adm:x:4:matt
cdrom:x:24:haldaemon,matt
audio:x:29:matt
matt:x:1000:
admin:x:112:matt
```

The fields in this file are also colon (:) separated and the fields, in order, are:

Field	Description
group_name	The name of the group.
password	The (encrypted) group password. If this field is empty, no password is needed. It is not common practice to have passwords on groups.
GID	The numerical group ID.
user_list	All the group members' user names, separated by commas. A user need not be listed with the group if that is his or her primary group as defined in /etc/passwd.

The /etc/gshadow File

The /etc/gshadow file is used to stored sensitive information about groups that should not be world readable.

As with all of the other files, the fields are separated by colons (:) and, in order, are:

- group name
- encrypted password
- comma-separated list of group administrators
- comma-separated list of group members

This file is not used frequently.

Creating New User and Group Accounts

User accounts can be created by manually editing the /etc/passwd and related files; however, that method is error prone. Instead, user accounts should be created with the useradd(8) command, such as:

```
[root@pitt:~] useradd newusername
```

On some distributions, this command will create a user entry in /etc/passwd and /etc/shadow and assign the user to a common group (such as the "users" or "staff" group). Other distributions will also create a group with the name of "newusername" and make that the primary group for the user.

Many useful options should be considered before running the command. They include:

Option	Description
-m	The user's home directory will be created if it does not exist.
-k SKEL	Used in conjunction with -m. This parameter provides a directory with files that should be copied to the new user's home directory. If this option is not used, the /etc/skel directory will be used.
-c comment	Set the GECOS or comment field in /etc/passwd.
-D	Display or assign default values for newly created accounts. This won't affect existing accounts, though.

This command varies slightly from distribution to distribution. Please consult the man page before using the command.

When using the -D option, you can change defaults such as the path to home directories, password and account expiry, default shells and more. These defaults are kept in /etc/default/useradd.

For example, to change the default shell, use the command:

```
[root@pitt:~] useradd -D -s /bin/zsh
```

Use the groupadd(8) command to create new groups. To add a group and assign the GID value yourself, use the -g option as follows:

```
[root@pitt:~] groupadd -g 456 newgroup
```

Assigning Passwords

Once a user or group is created, you must set any required passwords. To change a user password, user the passwd(1) command, as follows:

```
[root@pitt:~] passwd newusername
```

You will be prompted for the new password information. Non-root users can only change their own password. In that case, the passwd(1) command requires no arguments.
Account status information can also be obtained with the '-S" option"

To administer the /etc/group file, use the gpasswd(1) command.

Modifying User and Group Accounts

To modify the user information such as the GECOS field, home directory, password and account expiry, group membership and more, use the usermod(8) command.

The chage(1) command can also be used to modify the user's password expiry information.

To modify a group's GID value or the group name, use the groupmod(8) command.

These commands can only be used by the root user.

Deleting User and Group Accounts

Use the userdel(8) command to delete a user. By default, the home directory, mail spool, and crontabs will be left intact. To delete the home directory, use the "-r" option. The administrator must still clean up mail and cron files as well as anything left behind in /tmp or elsewhere.

Use the groupdel(8) command to delete a group.

1.111.2 Tune the user environment and system environment variables (Weight: 3)

Candidates should be able to modify global and user profiles.

Using the /etc/skel Directory

Any files in /etc/skel will copied into a new user account's home directory. For example:

```
[root@pitt:~] ls -la /etc/skel
```

Output:

```
total 18
-rw-r--r--  1 root root 220 2006-04-21 18:51 .bash_logout
-rw-r--r--  1 root root 414 2006-04-21 18:51 .bash_profile
-rw-r--r--  1 root root 2227 2006-04-21 18:51 .bashrc
```

```
[root@pitt:~] useradd -m tempuser

[root@pitt:~] ls -la /home/tempuser/
```

Output:

```
total 12
-rw-r--r-- 1 tempuser users 220 2006-07-22 17:01 .bash_logout
-rw-r--r-- 1 tempuser users 414 2006-07-22 17:01 .bash_profile
-rw-r--r-- 1 tempuser users 2227 2006-07-22 17:01 .bashrc
```

Add or remove files and directories as required.

Setting Environment Variables

Add environment variables for global users in the /etc/profile file because they are needed from the first login shell.

The ~/.bash_profile or ~/.profile files can be used for per-user settings.

Updating the PATH Environment Variable

The PATH environment variable is useful for finding programs on the system. For example, to add a bin/ directory in your home to the PATH environment variable, enter the following:

```
[matt@pitt:~] PATH="~/bin:$PATH"
[matt@pitt:~] export PATH
```

You need not export the PATH variable if it has already been exported.

1.111.3 Configure and use system log files to meet administrative and security needs (Weight: 3)

Candidates should be able to configure and manage system logs.

System logging with syslogd

Most system services running on a Linux Operating System will report logging information to the syslogd daemon. This daemon will then write the logs to files (typically in the /var/log directory), directly to the screen or even to another server on the network.

The default system log file is /var/log/messages. You can look through these log files for various events such as login attempts. For example, on an Ubuntu system, login attempts go to /var/log/auth.log:

```
[root@pitt:~] grep LOGIN /var/log/auth.log
```

Output:
Jul 17 20:41:50 otoole login[4900]: ROOT LOGIN on `tty1'

You can also use tail with the "-f" option to watch as new messages are logged to a syslog file. For example:

```
[root@pitt:~] tail -f /var/log/messages
```

Output:
Jul 22 14:16:48 otoole -- MARK --
Jul 22 14:36:48 otoole -- MARK --
Jul 22 14:56:48 otoole -- MARK --
Jul 22 15:16:49 otoole -- MARK --

will print out the last 10 lines of the /var/log/messages file, then wait for additional data to be available. Once more data shows up, it will also be printed to the screen.

The "-- MARK –" lines are in the log file to indicate that syslogd is still working even if no messages have come through from other applications.

Syslog Features

The syslogd daemon reads its configuration from the /etc/syslog.conf file. This file lists the logging functionality, selectors and actions. A selector is a combination of facilities and priorities. A facility is a category or class of message and the priority level varies from informational to emergency or critical.

The syslog facilities are:

Facility	Description
auth authpriv security	Only authpriv should be used now. The others are deprecated.
cron	Clock daemons (cron and at).
daemon	System daemons with no individual facility.
ftp	FTP services.
kern	Kernel messages.
lpr	Printing services.
mail	Mail services.
news	USENET news services.
syslog	Internally generated syslog messages.
user	User level messages (for instance, sent with the logger(1) command).
uucp	Unix-to-Unix Copy Protocol (UUCP) services.
mark	Internal use only. By default, the '-- MARK --' line will be written to the log file every 20 minutes.
local0 to local7	Available for local use.

The priority values are:

Priority	Description
debug	Debug level messages.
info	Informational messages.
notice	Normal condition but not significant.
warning	Warning conditions.
err	Error conditions.
crit	Critical conditions.

Priority	Description
alert	Action must be taken.
emerg	System is unusable.

The possible actions are:

Action	Description
Regular File	Typically, messages are logged to real files. The file has to be specified with a full path, beginning with a slash '/'. A '-' prefix means that syslog should not sync the file (flush it to disk) after every log message.
Named Pipes	Prepending the '|' character to a file indicates that it is a named pipe or fifo (first in, first out). A fifo can be created with the mkfifo command.
Terminal and Console	If the file you specified is a tty, special tty handling is performed; it is also performed with /dev/console. Any user logged into that terminal will see the messages.
Remote Machine	Prepending the '@' symbol indicates that a host name is being provided. All messages for this action will be sent to a different machine.
List of Users	A list of users who will receive the messages, if they are logged on. The list is separated by commas (,). No mail is generated.
Everyone logged on	Using an '*' character indicates that all logged-on users should get the message.

Some specifiers are available to make configuration more simple. They are:

Specifier	Description
*	Used as a priority, this means info level and above (ie. no debug) messages. Used as a facility, this means all facilities.
none	Used to indicate no priority for a given facility.
=	Indicates the only priority that should be logged. The default is that priority and higher.
!	Used to exclude priorities.
,	Used to group multiple facilities with the same priority.
;	Used to group disparate facility and priority settings that go to the same target.

The /etc/syslog.conf File

The following are some samples that may be found in the /etc/syslog.conf file.

Log all facility messages, except uucp, with a priority of warning to the /var/log/warning.log file. The '-' indicates that this file will not be synced after every write.

```
*.=warning;uucp.none     -/var/log/warning.log
```

All critical condition messages will be sent to the host 'loghost'.

```
*.crit                                  @loghost
```

Log all but debug priority messages for mail, news and uucp facilities to any terminals that the root or matt users are logged on to.

```
mail,news,uucp.*                root,matt
```

Log all mail facility messages except emerg to the console.

```
mail.!=emerg                            /dev/console
```

Security Log Files

Some other log files are kept in /var/log (and other directories) which are not managed by the syslogd daemon.

The who, w, last and lastlog commands query them for information.

These files and their purposes are:

File	Purpose
/var/log/lastlog	Records login dates, times and durations per user. Used by the lastlog command.
/var/log/wtmp	Records login times and durations per user. Used by the last and w commands.
/var/run/utmp	Records user information about current logins. Used by the finger, who and login programs.

The logrotate Program

The syslogd daemon has no facility for truncating or removing large log files. They will continue to grow.

To deal with this issue, the logrotate(8) command was created. It reads configuration information from /etc/logrotate.conf and the files in /etc/logrotate.d.

1.111.4 Automate system administration tasks by scheduling jobs to run in the future (Weight: 4)

Candidates should be able to use cron or anacron to run jobs at regular intervals and to use at to run jobs at a specific time.

The at Command Family

The at command is used to schedule a task to run once at some point in the future. An example use is:

```
[matt@pitt:~] at 5pm
at> echo "it is `date`"
at> <EOT>
```

Output:
job 1 at Mon Jul 24 17:00:00 2006

To get the <EOT>, hit the Ctrl-d key sequence.

Some available time formats are:

Time	Description
date	Specify a date and time, such as 5pm (the next 5pm) or '5pm Jan 1' or '17:00 013109'.
midnight	Run at the next 12:00am.
noon	Run at the next 12:00pm.
now + count time units	Run at 'count' 'time units' from now. For example, now + 1 day will run the at job 24 hours later. 'now' can be replaced with other times such as 4pm.
teatime	Run at the next 4:00pm.

Use /etc/at.allow to list users allowed access to the at command. Use /etc/at.deny to list users restricted from access to the at command. If neither file exists, only root has access. If only one file exists, all missing users are assumed to have the opposite access to what the file provides.

Use 'at -d' or 'atrm' to remove a pending at job.

Use 'at -l' or 'atq' to list the pending jobs for a user.

Use "batch" to execute a command when the system load level permits. Typically, this means once the load average drops below 1.5.

At and batch jobs are queued in the /var/spool/at/ or /var/spool/cron/atjobs/ directory.

The cron Service

The cron service allows scheduling of tasks, as do at and batch. The chief difference with cron is that these are tasks that are expected to run periodically.

Any time that a cron job is run, any output created by the job will be mailed to the user. Also, if the job exits with a non-zero value, the user will receive an e-mail notification.

The list of cron jobs (or crontabs) are kept in the /var/spool/cron/ or /var/spool/cron/crontabs/ directory.

Use /etc/cron.allow to list users allowed access to the cron service. Use /etc/cron.deny to list users restricted from access to the cron service. If neither file exists, only root has access. If only one file exists, all missing users are assumed to have the opposite access to what the file provides.

Systemwide jobs can be placed in the /etc/crontab file. Also, hourly, daily, weekly and monthly jobs can be scheduled by placing scripts or programs in /etc/cron.hourly, /etc/cron.daily, /etc/cron.weekly and /etc/cron.monthly, respectively. These jobs will run as root.

The crontab Command

The list and scheduling of the cron jobs are managed with the crontab(1) command. The important options to crontab are:

Option	Description
-e	Edit the crontab.
-r	Remove the entire crontab.
-l	List the crontab.
-u	Specify a user other than the one running the command (only for root).

A sample crontab file looks like the following:

```
MAILTO=matt@somedomain.com
30 5,15    * * *        /some/script/or/program
```

This example will run at 5:30am and 3:30pm (15:30) every day of every month. Any errors or output will be mailed to matt@somedomain.com (the default is to the user scheduling the cron job).

The line has six fields which, in order, indicate:

Field	Values
Minute	0-59 (an * means every minute)
Hour	0-23 (an * means every hour)
Day	1-31 (an * means every day unless the Day Of Week Field is set)
Month	1-12 (an * means every month, names are allowed)
Day Of Week	0-7 (0 and 7 mean Sunday, names are allowed)
Command	The actual command and arguments to execute

Time fields can also have multiple values separated by the following:

Separator	Description
,	Indicates multiple distinct values (eg. 3,6,8 means 3, 6 and 8)
-	Indicates a range (eg. 3-5 means 3, 4, and 5)
/	Indicates a step (eg. */20 would mean every twenty minutes starting at 0, 1-5/2 means 1,3,5)

The anacron Command

An alternative system to cron is anacron. Anacron will schedule commands periodically but takes computer downtime into account. So, if a job would have run but the computer was shut off, the job will run when the system restarts.

The default configuration file is /etc/anacrontab and the spool directory is /var/spool/anacron.

1.111.5 Maintain an effective data backup strategy (Weight: 3)

Candidates should be able to plan a backup strategy and back up filesystems automatically to various media.

The gzip, gunzip, bzip2, bunzip2, compress and uncompress commands provide compression and uncompression functionality.

The cpio command has three important options:

Option	Description
-o or --create	Add listed files to an archive.
-i or --extract	Copy files from the archive to the file system.
-p or --pass-through	Copy files from one directory tree to another, combining the copy-out and copy-in steps without actually using an archive.

File names are supplied as input to the command. The following command will create a new cpio archive of all .txt files under the matt user's home directory:

```
[matt@pitt:~] find . -name '*.txt' | cpio -o >newarchive.cpio
```

The tar command is used for the same functionality as cpio—namely, creating archives. The most important options are as follows:

Option	Description
-c	Create an tar archive
-t	List contents of a tar archive
-x	Extract files from a tar archive
-u	Update files in a tar archive
-v	Provide verbose output on stderr
-z	Use gzip compression on the tar archive
-j	Use bzip2 compression on the tar archive
-f	Specify the file name of the archive (this should usually be the last option)

Create a tar archive with a command such as:

```
[matt@pitt:~] tar cvf newarchive.tar
              or
[matt@pitt:~] tar zcvf newarchive.tar.gz
```

Extract all files from a tar archive with a command such as:

```
[matt@pitt:~] tar xvf newarchive.tar
              or
[matt@pitt:~] tar zxvf newarchive.tar.gz
```

The dd and dump commands can be used to create backups of an entire filesystem. Use dd and restore, respectively, to recover these backups. However, these commands are useful only on unmounted filesystems. Live or mounted filesystems will probably generate inconsistent backups.

1.111.6 Maintain system time (Weight: 4)

Candidates should be able to properly maintain the system time and synchronize the clock via NTP.

Setting System Time

The system time can be set with the date command such as:

```
[root@pitt:~] date 123115302006.05
```

This command will set the system clock to 3:30:05pm Dec 31, 2006. The format is MMDDhhmm[[CC]YY][.ss].

Use hwclock to read or assign the CMOS clock on the computer. To read the CMOS clock and assign it to the system time, use:

```
[root@pitt:~] hwclock --hctosys
```

To assign the system time to the CMOS (or hardware clock), use:

```
[root@pitt:~] hwclock --systohc
```

All time zone information is stored in files in the /usr/share/zoneinfo directory. To select a default time zone, link the appropriate file to /etc/localtime. For example, the following system is using Eastern Time (Eastern United States/Toronto, Canada) time zone.

```
[matt@pitt:~] ls -l /etc/localtime
```

Output:
lrwxrwxrwx 1 root root 35 12:43 /etc/localtime -> /usr/share/zoneinfo/America/Toronto

Some systems also put the time zone string in /etc/timezone.

Using the NTP Service

NTP, the Network Time Protocol, can be used to set the system clock based on information provided by other time sources.

These sources are categorized as follows:

Tier	Description
Tier 1	Top-level time providers. Usually, these providers are physically connected to devices such as atomic clocks.
Tier 2	Hosts that connect to Tier-1 providers and are expected to provide time information to lower levels.
Tier 3	A local time server at the user's company or ISP.

A NTP daemon (ntpd) must run to keep track of drift from the real time as a computer system runs. The configuration file /etc/ntp.conf controls this daemon's behaviour, including which time servers it connects with, to verify the correct time.

The file /etc/ntp.drift is use by ntpd to track clock drift. The ntpd daemon will slow or speed the system clock to adjust for drift. If the time is more than 1000 seconds off, ntpd will not start. Adjust the time manually, then restart ntpd.

The ntpdate(1) command can be used, similar to the date command, to set the system clock. In this case, the remote timeservers will be consulted for the correct time and it will immediately take effect.

Topic 112: Networking Fundamentals

1.112.1 Fundamentals of TCP/IP (Weight: 4)

Candidates should demonstrate a proper understanding of network fundamentals.

Important Network Concepts

Term	Description
IP Address	Assigned statically or through a dynamic server such as DHCP.
Network Mask	The portion of the IP address used to determine the network address (as opposed to the client address).
Broadcast Address	The network address plus all client address bits turned on. Used to send a message to all hosts on a network at once.
Gateway Address	The address of the host, router or other device that can forward network packets from your local network to the next.

An IP address is represented by 32 bits (4 8-bit bytes) for IPv4. The first (or higher order bits) indicate the network address while the lower order bits indicate the client address on that network.

An IP address of 192.168.2.12 and a network mask (or netmask) of 255.255.255.0 mean that the first 24 bits (3 8-bit bytes) are for the network address and the last 8 bits (1 8-bit byte) are for the client address.

Bits in an 8-bit byte have the values 128, 64, 32, 16, 8, 4, 2, 1 in left-to-right order. Add the values for each bit turned on to get the mask value (all bits on totals 255).

The address and network mask together can be specified as 192.168.2.12/255.255.255.0 or the shorter 192.168.2.12/24 (for a 24 bit network).

To compute the IP address, netmask, etc from the basic bits:

Network Addr: 11000000.10101000.00000010.10000000 (netmask==25 bits)
Client Addr: 00000000.00000000.00000000.01000000 (lower 7 bits)

Add the bits together in different combinations to get:

Network Addr: 11000000.10101000.00000010.10000000 (192.168.2.128)
Network Mask: 11111111.11111111.11111111.10000000 (255.255.255.128)
IP Address: 11000000.10101000.00000010.11000000 (192.168.2.192)
Broadcast: 11000000.10101000.00000010.11111111 (192.158.2.255)

In the above example, the last seven bits are used to provide client addresses, thereby producing 2^7 (or 128) addresses. However, all bits off means the network address and all bits on mean the broadcast. So, in fact, there are only 126 usable IP addresses on this subnet.

The IP Addresses range from:

Network Addr: 11000000.10101000.00000010.10000000 (aka, /25)
Lowest Addr: 00000000.00000000.00000000.00000001
Highest Addr: 00000000.00000000.00000000.01111110

When the lowest and highest addresses are combined with the network bits, the addresses 192.168.2.129 and 192.168.2.254 are determined.

Networks are sometimes referred to by "class:"

Class	Description
Class A	A /8 or 255.0.0.0 netmask. Also, formally used by the addresses starting with the number 1 to 126. There are 126 Class A networks; each can have 16,777,216 (2^24) host addresses.
Class B	A /16 or 255.255.0.0 netmask. Also, formally used by the addresses starting with the number 128 to 191. There are 16,382 Class B networks; each can have 65,536 host (2^16) addresses.
Class C	A /24 or 255.255.255.0 netmask. Also, formally used by the addresses starting with 192 to 233. There are 2,097,150 Class C networks; each can have 254 (2^8 -2) host addresses.
Class D	From 224 to 239. This range is reserved for activities such as multicast and is not usually available for host addresses.
Class E	From 240 to 254. Reserved for future use.

Port Numbers

Sending a packet to a remote system is insufficient. To determine which application gets the data, that application must be listening on a "port" for connections and data. Ports are numbered and a list of assigned port numbers is in the /etc/services file.

Commonly used port numbers from /etc/services are:

Port	Assigned Use	Port	Assigned Use
20	FTP-DATA (for data transfer)	21	FTP (command channel)
22	SSH	23	TELNET
25	SMTP	53	DNS
80	HTTP	110	POP3
119	NNTP (USENET)	143	IMAP2
161	SNMP	443	HTTPS

Data Transport Protocols

Packets come in many formats depending on their purpose:

Type	Purpose
IP	Internet Protocol - The lower-level protocol upon which more complex protocols are based.
ICMP	Internet Control Message Protocol - This protocol defines a small number of messages used for diagnostic and management purposes.
UDP	User Datagram Protocol - An unreliable, connectionless protocol. Unreliable really means that there is no guarantee that a sent packet will arrive in order or even arrive.
TCP	Transmission Control Protocol - A reliable service that ensures that the receiving application is given the data in the order that it is sent. It is connection based.

To send an ICMP packet to another host (and receive an ICMP packet in response), use the ping(8) command:

```
[matt@pitt:~] ping -c 2 -n 127.0.0.1
```

Output:

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.052 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.048 ms
--- 127.0.0.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.048/0.050/0.052/0.002 ms
```

No response means that the round trip was not successful. Something could be blocking the packet (although blocking is discouraged) or the remote machine may not be turned on.

The traceroute command is used to determine where along the network packets are being dropped as well as some performance indicators:

```
[matt@pitt:~] traceroute -n 10.10.10.2
```

Output:
```
traceroute to 10.10.10.2 (10.10.10.2), 30 hops max, 40 byte packets
 1 172.17.0.1 18.884 ms 20.327 ms 20.096 ms
 2 10.10.10.2 19.769 ms 22.522 ms 19.831 ms
```

To watch IP traffic passing on your network interfaces, use the tcpdump command:

```
[root@pitt:~] tcpdump
```

Output:
```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ath0, link-type EN10MB (Ethernet), capture size 96 bytes
16:34:10.390247 IP fibs.com.4321 > 192.168.2.192.38840: P 1420461721:1420461855(134) ack
3245961853 win 5792 <nop,nop,timestamp 1170903922 67293304>
16:34:10.573788 IP 192.168.2.192.38840 > fibs.com.4321: . ack 134 win 15948
<nop,nop,timestamp 67295730 1170903922>
...
```

1.112.3 TCP/IP configuration and troubleshooting (Weight: 7)

Candidates should be able to view, change and verify configuration settings and operational status for various network interfaces.

Viewing and Setting Network Interface Information

Use the ifconfig command to view all active interfaces:

```
[matt@pitt:~] ifconfig
```

Output:
```
eth0 Link encap:Ethernet HWaddr 00:05:4E:51:3C:44
inet addr:192.168.2.192 Bcast:192.168.2.255 Mask:255.255.255.0
inet6 addr: fe80::205:4eff:fe51:3c44/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          ...
```

Important fields in this output are the HWaddr (MAC address), inet addr, Bcast (broadcast address) and Mask (network mask).

Use the -a option to see all available interfaces even if not configured. The ifconfig command can also be used to assign network interface settings:

```
[root@pitt:~] ifconfig eth0 192.168.33.33
[root@pitt:~] ifconfig eth0
```

Output:

```
eth0   Link encap:Ethernet HWaddr 00:15:58:09:CC:CE
inet addr:192.168.33.33 Bcast:192.168.33.255 Mask:255.255.255.0
...
```

Normally, configuration information is placed in configuration files and init scripts configure everything automatically.

On Red Hat-based systems, configuration files are placed in /etc/sysconfig/network scripts/ifcfg-<device> files. On a Debian based system, the one file that configures all devices is /etc/network/interfaces. Once configured, the ifup and ifdown scripts will make the devices and their configuration active and inactive, respectively.

Routing Information

To view your routing table, use:

```
[matt@pitt:~] route -n
              or
[matt@pitt:~] netstat -rn
```

Output:

```
Kernel IP routing table
Destination  Gateway   Genmask    Flags Metric Ref  Use Iface
192.168.2.0  0.0.0.0   255.255.255.0  U   0   0   0 eth0
0.0.0.0    192.168.2.1 0.0.0.0     UG  0   0   0 eth0
```

The 0.0.0.0 (or default keyword without the -n option) indicates where packets go if they do not match any other destination. In this example, 192.168.2.1 is the gateway device for getting packets out of the local network.

On Red Hat systems, the gateway device is configured as a setting in one of the /etc/sysconfig/network-scripts/ifcfg-<device> files or in /etc/sysconfig/network. On Debian-based systems, this information is also stored in /etc/network/interfaces.

To change the default route from 192.168.2.1 to another network interface, use the commands:

```
[root@pitt:~] route del default
[root@pitt:~] route add default gw 192.168.2.2
[root@pitt:~] route -n
```

Output:

```
Kernel IP routing table
Destination  Gateway   Genmask    Flags MSS Window irtt Iface
192.168.2.0  0.0.0.0   255.255.255.0  U   0   0   0 eth0
0.0.0.0    192.168.2.2 0.0.0.0     UG  0   0   0 eth0
```

Configure a DHCP Client

To configure DHCP use on a Red Hat system, use the setting 'BOOTPROTO=dhcp' in the /etc/sysconfig/network-scripts/ifcfg-<device> file. On Debian, use a line such as 'iface eth0 inet dhcp' in /etc/network/interfaces.

Common dhcp client programs include dhcpcd (the best), dhclient and pump.

Looking Up Computers by Name

The name of your computer is stored in /etc/HOSTNAME or /etc/hostname (system dependent). It is set with the hostname and dnsdomainname commands at boot time. The hostname and domainname commands are also used to get host and DNS domain name information.

Host name lookups may refer to a number of sources. A local cache of host name to IP addresses are kept in /etc/hosts (networks can be named in /etc/networks but is not used by many programs). The DNS (domain name service) facility can be used as well (and is preferable because of the many computers using the Internet).

Originally, the /etc/host.conf file was used to specify the order of look up for host names. Now, the /etc/nsswitch.conf file is used. The "hosts" entry in nsswitch.conf determines the order:

```
[matt@pitt:~] grep "^hosts:" /etc/nsswitch.conf
```

Output:
```
hosts:    files dns
```

The output of the above command indicates that /etc/hosts (files) will be checked first, then DNS.

In order for DNS queries to work, the /etc/resolv.conf file must be properly configured. If DHCP is used, this information should be provided automatically. The /etc/resolv.conf file will resemble the following:

```
[matt@pitt:~] cat /etc/resolv.conf
```

Output:
```
search starnix.com
nameserver 192.168.2.1
nameserver 192.168.2.2
```

The "search" line means that any query that fails will be retried with the .starnix.com string added to the host name. Also, two DNS servers will be queried, in order, for a response.

The commands host, dig and nslookup can be used to test your host name lookup configuration:

```
[matt@pitt:~] host www.linux.com
```

Output:
```
www.linux.com has address 66.35.250.177
www.linux.com mail is handled by 10 mail.osdn.com.
```

Use the "-t" option to restrict the query to certain DNS record types.

1.112.4 Configure Linux as a PPP client (Weight: 3)

Candidates should understand the basics of the PPP protocol and be able to configure and use PPP for outbound connections.

PPP stands for Point-to-Point Protocol and is a method of providing IP-based networking across devices (such as modems) which do not inherently support IP.

The chat Program

The pppd daemon must be run on both sides. When it is used with modems, a dialer program must be provided. A typical program used for dialing is chat(8). A sample chat script is a set of send-expect pairs and looks like:

```
[matt@pitt:~] chat "" ATZ OK ATDT4165551212 CONNECT "" ogin: user ssword: pass
```

Read in sequence, this means:

- "" - send an empty string to wake up the modem.
- ATZ OK - send an ATZ command to the modem to reset it. Expect the OK response.
- ATDT5551212 CONNECT - dial the modem and wait for a CONNECT string back in response.
- "" ogin: - send nothing but wait for the string 'ogin:' (The 'l' in login is often capitalized. This script avoids the issue).
- user ssword: - send the user name (substitute your remote user name for the string "user), then expect the 'ssword:' string in response.
- pass - send the password.

Running the pppd Daemon

The pppd daemon can be started on the command line with options such as:

```
[root@pitt:~] pppd connect 'chat -f /etc/pppd/chatscript' /dev/ttyS0 57600 crtscts defaultroute
```

to specify /dev/ttyS0 as the physical device with a baud rate of 57600. Also, we define the program and arguments for making a connection, that we want hardware flow control and a default route added to the system routing table.

Other useful options include:

Option	Description
call name	Read additional options from the file /etc/ppp/peers/name.
debug	Enables connection debugging facilities.
demand	Initiate the link only on demand, i.e. when data traffic is present.
holdoff	Specifies how many seconds to wait before re-initiating the link after it terminates.
idle n	Specifies that pppd should disconnect if the link is idle for n seconds.
lock	Specifies that pppd should create a UUCP-style lock file for the serial device to ensure exclusive access to the device.
persist	Do not exit after a connection is terminated; instead try to reopen the connection.

Global options can also be placed in the /etc/ppp/options file and specific per-connection information can be placed in files in the /etc/ppp/peers directory (see the call option).

Helper Programs

wvdial(1) is an alternative tool to the chat program and is much more intelligent.

wvdialconf(1) can help build a configuration file for use with wvdial. The default location is /etc/wvdial.conf

The /etc/ppp/ip-up and /etc/ppp/ip-down scripts are run by pppd after the connection is brought up or down. These scripts are used to reconfigure routes, interfaces or anything that is required.

Topic 113: Networking Services

1.113.1 Configure and manage xinetd, inetd and related services (Weight: 4)

Configure and manage xinetd, inetd and related services.

Configuration for inetd is in the /etc/inetd.conf file. Configuration for xinetd is in the /etc/xinetd.conf file and files in the /etc/xinetd.d directory. These programs are used to listen on ports on behalf of infrequently accessed services. It keeps resource use to a minimal amount until the service is actually required.

The inetd Service

To add a service to inetd, enter a line such as:

pop3 stream tcp nowait root /usr/sbin/tcpd /usr/sbin/ipop3d

The fields are:

Field	Description
service name	The name of the service. The port number is determined by a corresponding entry in /etc/services.
socket type	stream, dgram, raw, rdm or seqpacket. stream and dgram are the most commonly used.
protocol	A valid protocol from /etc/protocols. tcp and udp are commonly used.
flags	wait or nowait.
user	The user name to run the command as. The root user is commonly used.
command	The command (with full path) plus any arguments.

The above example is for a pop3 service. The actual command being run by inetd is the /etc/sbin/tcpd program. This is the TCP wrappers program. It inspects the /etc/hosts.allow and /etc/hosts.deny files to determine additional access restrictions.

The xinetd Service

This is the eXtended inetd service. Each service being offered through xinetd gets its own configuration file in /etc/xinetd.d/. A sample xinetd configuration resembles the following:

[matt@pitt:~] cat /etc/xinetd.d/telnet

Output:

```
# default: on
# description: The telnet server serves telnet sessions; it
#    uses unencrypted username/password pairs for authentication.
service telnet
{
   flags      = REUSE
   socket_type   = stream
   wait      = no
   user      = root
   server     = /usr/sbin/in.telnetd
   log_on_failure += USERID
              cps                        = 25 30
   disable     = yes
}
```

The settings are similar to inetd with several exceptions (too numerous to mention all of them):

Field	Description
instances	Total number of daemon instances allowed.
cps	Maximum number of connections per second (the first number). The second number indicates how long to disable the service before allowing more connections.
disable	A "yes" means that this service is not available. A "no" makes the service available.
log*	What is logged plus when and the behaviour can be configured.
only_from access_times	Access control (similar to the add-on TCP wrappers).
port	Explicitly set the port number.

After editing the xinetd or inetd configuration file, restart it with either:

```
[root@pitt:~] /etc/init.d/xinetd restart
              or
[root@pitt:~] killall -HUP xinetd
```

The latter sends a HUP signal to the server indicating that the configuration should be reread. Comparable commands can be run for inetd.

1.113.2 Operate and perform basic configuration of Mail Transfer Agent (MTA) (Weight: 4)

Candidates should be able to operate and perform basic configuration of a MTA. Advanced custom configurations not included.

The particular internals of sendmail are not covered in the exam, but common functionality between MTAs is covered.

Sendmail configuration files are /etc/sendmail.cf (or /etc/mail/sendmail.cf). It is usually built from the /etc/mail/sendmail.mc file with a number of macro files and the m4 macro language.

Many additional configuration files such as /etc/mail/sendmail.cw are available for specifying which domains the MTA will accept incoming e-mail.

A few mail-specific acronyms are as follows:

Acronym	Description
MTA	Mail Transfer Agent - Commonly used MTAs in the Linux world are sendmail, postfix, exim and qmail.
MDA	Mail Delivery Agent - Separate programs such as procmail and mail which are used to complete the local delivery of e-mail for an MTA.
MUA	Mail User Agent – An end-user client for reading and sending e-mail such as evolution, kmail, elm, mutt and pine.

Creating Aliases

Add entries to the /etc/aliases file such as:

```
alias:      recipient1, recipient2
```

This addition will cause e-mail to alias@localdomain to be sent to recipient1 and recipient2. The recipients can be local users, fully qualified e-mail addresses and other aliases.

Once updated, use one of the two commands to notify the system of the new aliases:

```
[root@pitt:~] newaliases
        or
[root@pitt:~] sendmail -bi
```

The latter example is sendmail-specific but most sendmail alternatives will provide a newaliases command.

An individual can also "alias" or forward his or her e-mail address by placing an e-mail address in a .forward file in the home directory, as follows:

```
[matt@pitt:~] cat ~/.forward
```

Output:
matt@otherdomain.com

The .forward can also be used to pipe the e-mail into filter programs.

Interacting with Sendmail

Sendmail queues its outgoing mail in /var/spool/mqueue. Locally delivered mail is typically stored in /var/spool/mail/<username>.

To view the messages in the queue, use the command:

```
[root@pitt:~] mailq
```

Stop, start or restart the mail service with:

```
[root@pitt:~] /etc/init.d/sendmail stop
[root@pitt:~] /etc/init.d/sendmail start
        or
[root@pitt:~] /etc/init.d/sendmail restart
```

1.113.3 Operate and perform basic configuration of Apache (Weight: 4)

Candidates should be able to operate and perform basic configuration of Apache. Advanced custom configurations not required.

Starting Apache

To start apache, use the supplied apachectl or apache2ctl (for Apache version 2.x) programs or the init scripts, like:

```
[root@pitt:~] apachectl start
[root@pitt:~] apache2ctl start
        or
[root@pitt:~] /etc/init.d/apache start
```

This script will start a main apache process which runs as the root user. It will spawn child processes to handle actual requests. On Red Hat-based systems, the children typically run as the apache user. On Debian-based systems, the children typically run as the www-data user. As with other services, "stop" and "restart" commands also work.

The apache configuration files are in /etc/httpd, /etc/apache or /etc/apache2 directories, depending on the version of apache and the Linux distribution. The main apache configuration file is httpd.conf. Although srm.conf and access.conf files are available, their use has been deprecated.

A Sample of Configuration Options

Much of the configuration looks like XML directives. Some important apache directives (configuration options) are as follows:

Directive	Description
ServerAdmin	Sets the email address of the administrator.
DocumentRoot	Sets the location of the HTML files.
ServerRoot	Sets the local of the server configuration files.
MinSpareServers	Set the minimum number of idle child processes available in case of a surge of requests.
MaxSpareServers	Set the maximum number of idle child processes available in case of a surge of requests.
StartServers	The number of child processes to create upon startup.
MaxClients	The maximum number of requests to handle concurrently.
Port	Specify the port(s) to listen on for requests. The default is port 80.
Alias	Redirect requests to alternate directories for content.
DirectoryIndex	A list of files to search for if the user requests only a site or directory with no file name.

1.113.4 Properly manage the NFS and SAMBA daemons (Weight: 4)

Candidates should know how to manage the NFS, smb and nmb daemons.

Using NFS Filesystems

Six daemons make up the NFS service. They are:

Daemon	Description
portmap	A common facility used by services like NFS and NIS.
rpc.nfsd	The main NFS server daemon.
rpc.mountd	Handles mount requests.
rpc.rquotad	Used to support quotas on NFS.
rpc.lockd	Used to support file locking.
rpc.statd	Used by rpm.lockd to manage locks that remain on the server side after a crash.

To export a filesystem through NFS, add lines to /etc/exports like:

```
/pub      (ro)        *.linux.com(rw)
/home     192.168.2.0/24(rw)
```

The first line will export everything under /pub. The files will be read-only for everyone coming from a linux.com host. The second line will export the /home directory to anyone coming from the 192.168.2.0/24 network. More options are documented in the exports(5) man page.

After editing the /etc/exports file, let the running nfs service know about the change with:

```
[root@pitt:~] exportfs -a
```

On the client side, mount the filesystems with commands such as:

```
[root@pitt:~] mount -t nfs server:/pub /some/path
```

or add it to the /etc/fstab file with a line such as:

```
server:/pub /some/path nfs rw 0 0
```

To unmount the NFS filesystem, use:

```
[root@pitt:~] umount /some/path
```

just as you would unmount any filesystem.

Using Samba Filesystems

There are two main daemons with Samba:

- smbd - the SMB/CIFS file and printer sharing server. Typically, this is used by Windows clients. However, Linux clients can also access resources.
- nmbd - the name resolution and browsing server. This daemon includes all name resolution, browsing and WINS support that a Windows client would need.

The main configuration file is smb.conf or samba.conf, depending on the distribution. Thee configuration directory also varies depending on the distribution. The most common directories used are /etc and /etc/samba. The file is divided into three main sections: Global Settings, Homes and Printers. More than 300 configuration options exist, so many people use SWAT (Samba Web Admin Tool) for configuration. Other configuration files include /etc/samba/smbpasswd and /etc/samba/smbusers.

Many supplementary programs are available. They include:

Program	Description
smbclient	A command line tool for ftp-like access to SMB/CIFS resources.
testparm	Checks the smb.conf file for internal correctness.
smbpasswd	Update a user's SMB password.
smbadduser	Used to add a SMB user (including setting a password).
smbstatus	Report on current Samba connections.

1.113.5 Setup and configure basic DNS services (Weight: 4)

Candidates should be able to configure basic DNS services.

DNS server configuration is spread across multiple files:

File	Description
/etc/named.conf	The main configuration file for the server. May be in a directory other than /etc.
zone files	These files define the name to IP address mappings for a given domain name.
reverse zone files	These files define the IP address to name mappings.
hints files	These files provide root server information. The root servers can be used to perform queries if the local files or upstream DNS servers cannot determine an answer.

There are three main ways to configure a name server, as follows:

Type	Description
Master	The server will hold a zone file for the domain.
Slave	The server will cache a copy of the zone file provided by the master.
Cache	Does not have zone files. It performs DNS queries and caches the result for later use.

A simple named.conf file which defines one non-default option, one domain zone (as a master) and one reverse zone file, looks like:

[matt@pitt:~] cat /etc/named.conf

Output:

```
options {
        directory "/etc/named.d";
};
zone "linux.com" {
        type master;
        file "db.linux.com";
};
zone "2.168.192.in-addr.arpa" {
        type master;
        file "db.2.168.192";
};
```

A simple zone file looks like:

[matt@pitt:~] cat /etc/named.d/db.linux.com

Output:

```
$TTL 1800
@       IN  SOA   ns.linux.com. dns.linux.com. (
            2003060601   ; serial (version)
            1800     ; refresh
            600      ; retry
            3600000    ; expire
            600 )     ; minimum
      IN   NS   ns.linux.com.
      IN   NS   ns2.linux.com.
      IN   MX   10 titan.starnix.com.
@       IN  A    10.42.42.2
www      IN  A    10.42.42.45
```

A simple reverse zone file looks like:

```
[matt@pitt:~] cat /etc/named.d/rev.linux.com
```

Output:

```
$TTL 86400
@       IN SOA ns.linux.com. dns.linux.com. (
          2003060601   ; serial
          10800     ; refresh
          3600     ; retry
          604800    ; expire
          86400     ; default_ttl
          )
       IN NS    ns.linux.com.
       IN NS    ns2.linux.com.
1       IN PTR    callisto.linux.com.
254     IN PTR    dmz2.linux.com.
```

Read the named.conf(5) man page for details. What should be noted here are that there are many different record type for entries in the zone files. They include:

Type	Description
SOA	A Start Of Authority record to define the zone and the domain or sets of domains that it covers.
NS	A Name Server record which defines what the correct names servers are for future queries.
A	An Address record. The most general type, it maps a name to an IP address.
CNAME	A Canonical NAME record. It is used to provide the equivalent of aliases for other records.
PTR	A Pointer record that is used for reverse DNS mappings (IP to name).
MX	A Mail eXchanger record for routing SMTP traffic.
TXT	A Text record for not keeping.

1.113.7 Set up secure shell (OpenSSH) (Weight: 4)

Candidates should be able to obtain and configure OpenSSH.

A number of programs are related to the secure shell. They include:

Program	Description
sshd	The ssh daemon
ssh	The ssh client program used for remote logins and running remote commands similar to the rsh and telnet programs
scp	A secure copy program similar to rcp
ssh-agent	An end-user program that will securely cache a user's authentication and provide it upon request
ssh-add	The means of loading a user's key(s) into an ssh-agent process
ssh-keygen	A program used to create private/public keys for use with ssh

Many configuration files are also available in the /etc/ssh directory; however, the most important one for server (sshd) configuration is /etc/ssh/sshd_config. The /etc/ssh/ssh_config is a global client configuration file. Per-user configuration is kept in the ~/.ssh directory.

To use ssh to run an ls(1) command on another server, use:

```
[matt@pitt:~] ssh -l mat remotehost ls
```

Output:

```
-rw-rw-r--  1 mat   mat    321 Sep 23 2004 1033.pdf
-rw-rw-r--  1 mat   mat    351 Sep 23 2004 1034.pdf
-rw-rw-r--  1 mat   mat    927 Sep 23 2004 1035.pdf
```

To copy files to the remotehost, use:

```
[matt@pitt:~] scp lpi-117.txt mat@remotehost:tmp
```

Output:

```
lpi-117.txt          100% 8890   8.7KB/s  00:00
```

This command will copy the lpi-117.txt to the ~mat/tmp directory on the remotehost machine. If you wanted to copy it to the /tmp directory, use:

```
[matt@pitt:~] scp lpi-117.txt mat@remotehost:/tmp
```

This command is required because paths not beginning with a '/' are considered relative to the user's home directory.

Topic 114: Security

1.114.1 Perform security administration tasks (Weight: 4)

Candidates should know how to review system configuration to ensure host security in accordance with local security policies.

Configure TCP wrappers

Use the /etc/hosts.allow and /etc/hosts.deny files. To allow a certain domain in (and all subdomains and machines) but deny everyone else, use a command such as:

```
[matt@pitt:~] cat /etc/hosts.allow
```

Output:
ALL: .foobar.edu

```
[matt@pitt:~] cat /etc/hosts.deny
```

Output:
ALL: ALL

Finding Files that are SUID or SGID

To find any files on your system with any special bits set (SUID, SGID or the sticky bit), use the command:

```
[root@pitt:~] find / -perm +7000
```

Using nmap and netstat Commands

The nmap command is useful for discovering open ports on a remote system. A command such as:

```
[root@pitt:~] nmap localhost
```

Output:
Interesting ports on localhost (127.0.0.1):
PORT STATE SERVICE
631/tcp open ipp
1665/tcp open netview-aix-5
Nmap finished: 1 IP address (1 host up) scanned in 0.245 seconds

The nmap command has many features including OS detection, scan ordering and techniques, firewall evasion and spoofing and much more.

The netstat command is useful if you have login access to the system. The following command lists all open TCP ports on the system:

```
[root@pitt:~] netstat -tl
```

Output:

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address    Foreign Address   State
tcp    0   0 *:netview-aix-5   *:*         LISTEN
tcp    0   0 localhost:ipp    *:*         LISTEN
```

Configuring Firewalling with iptables

The iptables system is for IP packet filtering, inspection and NAT (Network Address Translation).

Several tables may be defined; each table contains built-in chains plus optional user-defined chains. A chain is a list of rules which have criteria for matching a packet and a target.

The main targets are:

Target	Description
ACCEPT	let the packet through
DROP	drop the packet on the floor
QUEUE	pass the packet to userspace
RETURN	stop traversing the current change and return to the previous chain

The three default tables are:

Name	Purpose
filter	The default table. It contains built-in INPUT, FORWARD and OUTPUT chains.
nat	Used when creating a firewall where all of the internal machines need access to the Internet or another network.
mangle	Used to modify packets.

To inspect the default (filter) table and its chains, use the command:

```
[root@pitt:~] iptables -L
```

Output:

```
Chain INPUT (policy ACCEPT)
target   prot opt source       destination
Chain FORWARD (policy ACCEPT)
target   prot opt source       destination
Chain OUTPUT (policy ACCEPT)
target   prot opt source       destination
```

The policy indicates what to do with a packet if none of the rules match.

To add a rule preventing packets originating from a specific IP address from reaching your system, use a command like:

```
[root@pitt:~] iptables -A -s 192.168.5.6 -j DROP
```

There are too many options to list here. Please consult the iptables(8) man page.

1.114.2 Setup host security (Weight: 3)

Candidates should know how to set up a basic level of host security.

Things to be considered when hardening a system:

- Turn off services not required, including services in /etc/inittab, /etc/rcN.d/, /etc/xinetd.conf (or /etc/xinetd.d/), and /etc/inetd.conf.
- Adjust password aging with the passwd or chage commands.
- Set up shadow password. The pwconv, pwunconv, grpconv and grpunconv commands will perform the conversion.
- Forward the root account's e-mail to another system.
- Make use of the /etc/nologin file when performing sensitive tasks. This file will prevent non-root users from logging in to the system.
- Possibly configure syslogd to log important security-related messages to another syslogd service on a separate host.

1.114.3 Setup user level security (Weight: 1)

Candidates should be able to configure user-level security. Tasks include limits on user logins, processes and memory usage.

Use the ulimit(1) command to list or set user limits. To get a list of current user limits, enter the following:

```
[matt@pitt:~] ulimit -a
```

Output:

```
core file size    (blocks, -c) 0
data seg size     (kbytes, -d) unlimited
max nice          (-e) 20
file size         (blocks, -f) unlimited
...
```

A 0 block size for core files is useful because core files leave a trace of what a user's process was doing. The 0 size means that users will not generate a core file on errors in programs.

To set a limit (for debugging) on the core file size, use:

```
[matt@pitt:~] ulimit -c 5000
```

The ulimit command is documented in the bash man page because it is a built-in command.

Practice Questions

Chapter 1 Kernel

1. Which of the following commands will unload the vfat kernel module, including all modules that are only needed to support the vfat module?

   Select the best answer.

   - ❍ A. lsmod vfat
   - ❍ B. rmmod vfat
   - ❍ C. modprobe -r vfat
   - ❍ D. modprobe vfat
   - ❍ E. insmod vfat

2. Which make targets are valid choices when attempting to configure options for kernel compilation?

   Select the 2 best answers.

   - ❑ A. oldconfig
   - ❑ B. reconfig
   - ❑ C. guiconfig
   - ❑ D. menuconfig

Chapter 2 Boot, Initialization, Shutdown and runlevels

1. Which of the following is the name of the file that contains the Linux operating system's default runlevel setting?

   Select the best answer.

   - ❍ A. /etc/runlevel
   - ❍ B. /etc/inittab
   - ❍ C. /sbin/init
   - ❍ D. /etc/defaults

2. Which of the following programs may be used to properly shut down the Linux operating system?

   Select the 2 best answers.

   - ❑ A. shutdown
   - ❑ B. stop
   - ❑ C. halt
   - ❑ D. suspend

Chapter 3 Printing

1. Which of the following are names for the printer services in common use today?

   Select the 2 best answers.

   - ❑ A. LPD
   - ❑ B. LP
   - ❑ C. CUPS
   - ❑ D. IPC

2. Which of the following commands will take the output of a previous command, called cmd, and send it to the default printer queue?

   Select the best answer.

   - ❍ A. cmd && lpr
   - ❍ B. cmd|lpr
   - ❍ C. cmd || lpr
   - ❍ D. cmd >lpr

Chapter 4 Security

1. Someone from the 192.168.5.6 IP address is flooding your mail server with spam and trying to attack your web server. You decide to use iptables to completely ignore the traffic coming from that address. Which of the following commands would you use to accomplish this task?

   Select the best answer.

   - ❍ A. iptables -A -src 192.168.5.6 -j REJECT
   - ❍ B. iptables -A -s 192.168.5.6 -j REJECT
   - ❍ C. iptables -A INPUT -s 192.168.5.6 -j DROP
   - ❍ D. iptables -A -src 192.168.5.6 -j DROP

2. Which of the following commands can be used to convert legacy /etc/passwd files to the more secure /etc/passwd and /etc/shadow combination?

   Select the best answer.

   - ❍ A. shadowconvert
   - ❍ B. shadconv
   - ❍ C. pwconv
   - ❍ D. passwdconvert

Chapter 5 Documentation

1. You want to read the section 5 man page for the crontab command and not the section 1 man page for the crontab command. Which of the following commands would you use to accomplish this task?

   Select the best answer.

   - ❍ A. man crontab 5
   - ❍ B. man --next crontab
   - ❍ C. man crontab
   - ❍ D. man 5 crontab

2. Which of the following web sites is the official home of many HOW-TO, MINI HOW-TO and other assorted Linux documentation efforts?

   Select the best answer.

   - ❍ A. www.tldp.org
   - ❍ B. www.thelinuxdoc.org
   - ❍ C. www.linux.com
   - ❍ D. www.linuxtoday.com

3. You want to leave a message for the people that successfully log on to the computer at the terminal. Which of the following files is the most appropriate file for this message?

   Select the best answer.

   - ❍ A. /etc/issue.net
   - ❍ B. /etc/issue
   - ❍ C. /etc/motd
   - ❍ D. /etc/msg

Chapter 6 Shells, Scripting, Programing and Compiling

1. You want to assign the output of a command to a bash shell variable. What is the correct syntax for this task?

   Select the 2 best answers.

   - ❑ A. VAR=`cmd`
   - ❑ B. VAR="cmd"
   - ❑ C. VAR=$(cmd)
   - ❑ D. VAR=cmd
   - ❑ E. VAR = `cmd`

2. You want to print out the process ID of your shell script from inside the script itself. Which of the following commands will accomplish this task?

   Select the best answer.

   - ❍ A. echo $$
   - ❍ B. echo $?
   - ❍ C. echo $0
   - ❍ D. echo $pid

Chapter 7 Administrative Tasks

1. Which of the following files stores the group names and secondary group assignments for the Linux operating system?

   Select the best answer.

   - ❍ A. /etc/groups
   - ❍ B. /etc/grp
   - ❍ C. /etc/group
   - ❍ D. /etc/passwd

2. Which of the following commands is used to create normal user accounts in the Linux operating system?

   Select the best answer.

   - ❍ A. acctadd
   - ❍ B. useradd
   - ❍ C. usercreate
   - ❍ D. acctcreate

3. Which of the following directories is commonly used for syslog message log files?

   Select the best answer.

   - ❍ A. /var/syslogd
   - ❍ B. /var/syslog
   - ❍ C. /var/log
   - ❍ D. /var/logs

4. Which of the following services is used to schedule jobs that are required to run periodically?

   Select the best answer.

   - ❍ A. cron
   - ❍ B. at
   - ❍ C. batch
   - ❍ D. jobs

5. Which of the following commands can be used to archive a directory and its files?

   Select the 2 best answers.

   - ❑ A. tar
   - ❑ B. gzip
   - ❑ C. cpio
   - ❑ D. bzip2
   - ❑ E. backup

6. Which of the following commands can be used to read the time in the Hardware Clock and set the time in the kernel's system clock?

   Select the best answer.

   - ❍ A. clock --systohc
   - ❍ B. clock --hctosys
   - ❍ C. hwclock --systohc
   - ❍ D. hwclock --hctosys

Chapter 8 Networking Fundamentals

1. You have an IP address of 192.168.2.128 and a 21 bit netmask. Which of the following values is the correct network address?

   Select the best answer.

   - ❍ A. 192.168.2.0
   - ❍ B. 192.168.0.0
   - ❍ C. 255.255.248.0
   - ❍ D. 255.255.255.0

2. Which of the following commands can be used to display the network routing table for the Linux operating system?

   Select the 2 best answers.

   - ❑ A. telnet
   - ❑ B. route
   - ❑ C. netstat
   - ❑ D. routes

Chapter 9 Networking Services

1. Which of the following files is the default configuration file for the xinet service?

   Select the best answer.

   - ❍ A. /etc/xinet.conf
   - ❍ B. /etc/inetd.conf
   - ❍ C. /etc/inet.conf
   - ❍ D. /etc/xinetd.conf

2. After updating the aliases file on a system, which of the following commands is used to rebuild the aliases database file?

   Select the best answer.

   - ❍ A. aliases
   - ❍ B. buildaliases
   - ❍ C. newaliases
   - ❍ D. updatealiases

3. Which of the following is the name of the main configuration file for the Apache server program?

   Select the best answer.

   - ❍ A. httpd.conf
   - ❍ B. access.conf
   - ❍ C. srm.conf
   - ❍ D. apached.conf

4. Which of following is the type of DNS record used for directing SMTP (e-mail) traffic?

   Select the best answer.

   - ❍ A. TXT
   - ❍ B. NS
   - ❍ C. MTA
   - ❍ D. MX

Answers and Explanations

Chapter 1

1. Answer: C

Explanation A. Incorrect. This command is used to list loaded kernel modules. This is also an incorrect use of the command.

Explanation B. Incorrect. This will unload the vfat module but not any dependent modules.

Explanation C. Correct. This command will unload the vfat module along with any dependencies that are not required by other loaded modules.

Explanation D. Incorrect. This command will attempt to load the vfat module along with all dependencies.

Explanation E. Incorrect. This will only attempt to load the vfat module. No dependencies will be checked.

2. Answers: A, D

Explanation A. Correct. This will import a previous configuration as a starting point.

Explanation B. Incorrect. This is not a valid target for the kernel make file.

Explanation C. Incorrect. This is not a valid target for the kernel make file.

Explanation D. Correct. This target will start a text-based configuration tool.

Chapter 2

1. Answer: B

Explanation A. Incorrect. There is no such file. However, there is a runlevel command which will report the current runlevel.

Explanation B. Correct. This is the correct file. The initdefault action defines the default runlevel.

Explanation C. Incorrect. This is the init daemon which will read the /etc/inittab configuration file.

Explanation D. Incorrect. There is no such file.

2. Answers: A, C

Explanation A. Correct. This is the most versatile command for shutting down the operating system.

Explanation B. Incorrect. There is no such command. When running individual init scripts, the 'stop' argument will cause the service to shut down or stop.

Explanation C. Correct. This command will call shutdown unless the Linux operating system is already in runlevel 0 or 6. In runlevel 0 or 6, halt will tell the kernel to shutdown, poweroff or reboot.

Explanation D. Incorrect. This is a bash shell command to suspend the currently running shell process until it receives the SIGCONT signal.

Chapter 3

1. Answers: A, C

Explanation A. Correct. LPD stands for the Line Printer Daemon.

Explanation B. Incorrect. This is not a service name.

Explanation C. Correct. The CUPS printer service is the newer technology. CUPS stands for the Common UNIX Printing System.

Explanation D. Incorrect. This is not a valid service. IPC simply stands for InterProcess Communication. This is a generic term for communication between processes.

2. Answer: B

Explanation A. Incorrect. This will run the cmd command and, if successful (a 0 exit value), will run the lpr command. The lpr command will be expecting input on its standard input, because no arguments have been provided.

Explanation B. Correct. This is the correct answer. The output of the 'cmd' command will be used as the standard input of the 'lpr' command.

Explanation C. Incorrect. This will run the cmd command and, if unsuccessful (a non-zero exit value), will run the lpr command. The lpr command will be expecting input on its standard input, because no arguments have been provided.

Explanation D. Incorrect. This will run the cmd command and write the output to a file called lpr.

Chapter 4

1. Answer: C

Explanation A. Incorrect. -src is not a valid flag for the iptables command. Also, it is more typical for full-word options to require two minuses before the option (ie. --src).

Explanation B. Incorrect. The REJECT option will cause the packets to be ignored but will send error packets back to the attacker.

Explanation C. Correct. This is the correct answer.

Explanation D. Incorrect. -src is not a valid option to iptables. Also, it is more typical for full-word options to require two minuses before the option (ie. --src).

2. Answer: C

Explanation A. Incorrect. This command does not exist.

Explanation B. Incorrect. This command does not exist.

Explanation C. Correct. There is a pwunconv command for the reverse operation. Also, there is the pwck command for verifying the new configuration files.

Explanation D. Incorrect. This command does not exist.

Chapter 5

1. Answer: D

Explanation A. Incorrect. The section number comes before the man page name.

Explanation B. Incorrect. There is no --next option for the man command.

Explanation C. Incorrect. This will show the first crontab man page. In this case, it will be the section 1 man page. The section 5 man page on crontab will not be shown.

Explanation D. Correct. This is the correct answer. The man page section's number must come before the topic of interest.

2. Answer: A

Explanation A. Correct. This is the correct answer.

Explanation B. Incorrect. No such web site exists.

Explanation C. Incorrect. This is a Linux portal and news web site.

Explanation D. Incorrect. This is a Linux news web site.

3. Answer: C

Explanation A. Incorrect. This file is used for messages to people logging in over the network (except ssh), and the message is presented before the login prompt is displayed.

Explanation B. Incorrect. This is the file that contains the contents to display before printing the login prompt. This information will be presented to anyone attempting to log in.

Explanation C. Correct. This is the correct answer.

Explanation D. Incorrect. There is not such file.

Chapter 6

1. Answers: A, C

Explanation A. Correct. This is the 'traditional' method of executing a command and having that command's output assigned to a variable.

Explanation B. Incorrect. This will assign the string cmd to the variable.

Explanation C. Correct. While this is a correct answer, it is, however, bash centric and may not work in other shells.

Explanation D. Incorrect. This will assign the string cmd to the variable.

Explanation E. Incorrect. There can not be any spaces between the = sign and the value or the variable name. Otherwise, this would have been a correct response.

2. Answer: A

Explanation A. Correct. This is the correct answer. The actual variable is '$'. The preceding '$' symbol is used to de-reference the variable to get the value.

Explanation B. Incorrect. This will print out the exit value of the last command executed by the shell script.

Explanation C. Incorrect. The variable '0' contains the name of the shell script being executed.

Explanation D. Incorrect. This is not a shell variable. If it was a shell variable, it would either be a symbol, number or use upper case letters.

Chapter 7

1. Answer: C

Explanation A. Incorrect. This is not a valid file. However, the groups command will print out a list of groups to which a user belongs.

Explanation B. Incorrect. This is not a valid file.

Explanation C. Correct. This is the correct answer. All secondary group memberships are listed in this file, along will the complete group definitions.

Explanation D. Incorrect. This is the user account information file. It does store the primary group ID for each user. This file does not contain group names and secondary group assignments.

2. Answer: B

Explanation A. Incorrect. This is not a valid command.

Explanation B. Correct. There is also an older adduser command floating around, but it shouldn't be relied upon.

Explanation C. Incorrect. This is not a valid command.

Explanation D. Incorrect. This is not a valid command.

3. Answer: C

Explanation A. Incorrect. This directory does not exist on a default Linux installation.

Explanation B. Incorrect. This directory does not exist on a default Linux installation.

Explanation C. Correct. Most log files, even if not managed with syslog, are written in this directory or in a subdirectory of /var/log.

Explanation D. Incorrect. This answer is incorrect. This directory does not exist on a default Linux installation.

4. Answer: A

Explanation A. Correct. The cron daemon is for automating periodic and repeating tasks.

Explanation B. Incorrect. This service will generally only run the command once.

Explanation C. Incorrect. Similar to the at command, this job will generally only run once. The main difference is that batch will wait until the system load drops to an acceptable level. There is no commitment on when the actual command will run.

Explanation D. Incorrect. This is a shell built-in command mostly used for listing background and suspended processes that are being managed by the shell.

5. Answers: A, C

Explanation A. Correct. Tar stands for Tape Archiver. This program can write the archive to regular files and standard output, as well as to tape devices.

Explanation B. Incorrect. This is a compression utility. It is not a program for creating archives.

Explanation C. Correct.

Explanation D. Incorrect. This is a compression utility. It is not a program for creating archives.

Explanation E. Incorrect. This is not a valid command.

6. Answer: D

Explanation A. Incorrect. Clock is not a valid command.

Explanation B. Incorrect. Clock is not a valid command.

Explanation C. Incorrect. This command and option will read the kernel's System Clock time and set the Hardware Clock's time.

Explanation D. Correct. This is the correct answer. It will take the time in the Hardware Clock and set the kernel's system clock to that time.

Chapter 8

1. Answer: B

Explanation A. Incorrect. It would be correct for a 23-or 24-bit network mask.

Explanation B. Correct.

Explanation C. Incorrect. This value is the netmask corresponding to a /21 network. It is not a network address.

Explanation D. Incorrect. This value is a netmask corresponding to a /24 network. It is not a network address.

2. Answers: B, C

Explanation A. Incorrect. This is the command for connecting to services listening on TCP ports, including the telnet port for remote logins.

Explanation B. Correct. This command is also used to manipulate the system routing table.

Explanation C. Correct. This command is also used for displaying other network information. In order to see the routing table, use a command like: netstat -r

Explanation D. Incorrect. This is not a valid command.

Chapter 9

1. Answer: D

Explanation A. Incorrect. This file does not exist.

Explanation B. Incorrect. It is the default configuration file for the inet service.

Explanation C. Incorrect. This file does not exist.

Explanation D. Correct. This is the correct configuration file. Additionally, there is an /etc/xinetd.d directory containing additional configuration files and information.

2. Answer: C

Explanation A. Incorrect. This command does not exist. It is similar to the shell command for listing aliases. That command is alias.

Explanation B. Incorrect. This command does not exist.

Explanation C. Correct. The newaliases command is supported by most commonly-used mail server software. All mail server programs that require an aliases database (generally for performance reasons) will have their own command as well.

Explanation D. Incorrect. This command does not exist.

3. Answer: A

Explanation A. Correct. Some Linux distributions use apache.conf. On a real LPI exam, both answers would be acceptable.

Explanation B. Incorrect. The access.conf file is deprecated and no longer used.

Explanation C. Incorrect. The srm.conf file is deprecated and no longer used.

Explanation D. Incorrect. This file does not exist. However, some distributions use the apache.conf file.

4. Answer: D

Explanation A. Incorrect. This is the record type for comments about another record. It is also being used by SPF for fighting spam.

Explanation B. Incorrect. This is the record type for DNS name servers.

Explanation C. Incorrect. This is not a record type. It is an acronym for Mail Transfer Agent.

Explanation D. Correct. It stands for Mail eXchanger.