The Top 50 Things to Know to Pass the LPIC Exam 101
This Page was created by " **MM Linux** User Group" Feb 2005

The Linux Professional Institute offers three levels of certification, with each level consisting of two exams that you must pass. The exams are administered by VUE only. The upper two levels are still in development, but the first level is now available. The two exams at the first level are 101 and 102. The following list of fifty key points to know is excerpted from the upcoming book, *LPI General Linux 1 Exam Cram (Exam 101),* to be published in March 2001 by The Coriolis Group (ISBN: 1-57610-923-2). Used with permission.

GNU and Unix Commands

1. The shell is the interpreter between the operating system and the user. Most versions of Linux offer more than one shell, and users can choose the one that best suits them. Among the common shells found in most implementations of Linux are:

- bash - the Bourne Again Shell (most heavily tested upon with this exam)
- tcsh – Tom's C Shell

2. The **cat** command is the simplest way to view all of a text file (or in reverse order with **tac**). If you only want to view the top, you can use the **head** command and if you only want to see the bottom, you can use the **tail** command.

3. The **cut** command can pull fields from a file, and they can be combined using either **paste** or **join.** The latter offers more features than the former, and can be used conditionally.

4. The **expand** command can change tabs into spaces, while the **fmt** command can crop text to fit a line's character limitation and **pr** can work with pagination. Lines can be numbered in the display with the **nl** command, and **sort** can place items in alphabetic or numeric (using the -n parameter) order.

5. The **od** command can display portions of non-text files by doing an octal dump. **tr** can convert one character set to another, while **wc** can count the number of lines, words, and characters in a file.

6. Files can be copied using **cp,** or moved using **mv.** Files can be deleted with **rm,** and directories (which are created with **mkdir**) can be removed with **rmdir**. In all situations, you can specify target and source items using either absolute or relative addresses, and your wildcards as well. To change directories, use the **cd** command (when used without parameters, this will move you to your home directory), and to see what directory you are presently working in, use the **pwd** (present working directory) command.

7. Standard input (**stdin**) is traditionally the keyboard and standard output (**stdout**) is traditionally the monitor. Both can be redirected, as can standard error (**stderr**). Methods of redirection include using the symbols >, >>, <, and |. You can also utilize the **tee** command to send output to two locations.

8. Commands can be joined on the command line by the semicolon (;), and each command will run independent of one another. Command history allows you to run commands over again without needing to retype them, and allows for editing prior to execution. When run, every command spans at least one process, and processes can be viewed with **ps** or **top** (which continues to dynamically update the display).

9. Jobs running in the foreground can be suspended by pressing ^Z (Ctrl+Z). They can be moved to the background with the **bg** command and background jobs can be brought to the foreground with the **fg** command. Running jobs can be listed with the jobs command, and processes can be eradicated with the use of **kill**.

10. The priority level of a job can be altered before it starts with the **nice** command. After the job has started, only **renice** can change its priority.

11. The **sed** editor can be used to change text as it is displayed, using a defined set of parameters it is given. The **grep** utility (and its counterparts, **egrep** and **fgrep**) can be used to find matches for strings within files.
Devices, Linux File Systems, Filesystem Hierarchy Standard

12. The **fdisk** command is used to create partitions, which can then be formatted using **mkfs**. The **mkfs** utility allows you to format a number of different filesystem types, including msdos, and others.

13. The filesystem contains an inode (a unique entry) for every entity stored upon it. This inode (number visible with **ls -i**) holds information about the entity including permissions, owner, group, and associated dates for creation/access/modification (which can be changed with the **touch** command).

14. The **du** command can show how much of the disk is used, while the **df** command shows how much is free. The main troubleshooting tool for disks is **fsck** which can check filesystem structure, including inodes.

15. The root filesystem must always be mounted, but remote ones can be mounted with the **mount** command, or unmounted with the **umount** command. To have mounting automatically occur at startup, the entries are added to /etc/fstab.

16. Quotas can restrict the amount of disk space users or groups can utilize. Quotas are initialized with the quota command, and toggled on and off with **quotaon** and **quotaoff**. They can be changed/modified with **edquota** and reports can be generated with **repquota**.

17. File permissions can be expressed in terms of symbolic, or numeric values. When first creating files, the default permissions are equal to 666 minus any **umask** value. The default permissions on directories are equal to 777 minus any **umask** value. When computing numeric permissions, the ability to read is worth 4, to write is worth 2, and to execute is worth 1.

18. File and directory permissions can be changed with the **chmod** command (which accepts numeric and symbolic values). Adding 1000 to the value turns on the "sticky bit", while adding 2000 turns on the SGID permission, and 4000 turns on the SUID permission.

19. The owner of a group can be changed with the **chown** command, while the **chgrp** command allows changing the group associated with the file.

20. Links are created with the **ln** command. A "hard" link is nothing more than an alias to the file, and all aliases share the common inode. A symbolic link is created with **ln -s** and is an actual file (of very small size) with its own inode. The symbolic link contains a pointer to the original file and can span across filesystems (while a hard link cannot).

21. There are a number of directories created by default during the installation of Linux. Among them:

- /bin - holds binary (executable) files accessible by all users
- /dev - holds device definitions
- /etc - is used for files that differ from machine to machine (configuration files)
- /home - holds the users' home directories
- /mnt - is used for mount points
- /root - the home directory for the root user
- /tmp - is used for temporary files that need not survive a reboot
- /usr - holds documentation and other entries that do not change often. Among the subdirectories are those holding additional executables
- /var – where data that changes is kept. This includes log files, spools, and the like.

22. The **find** command can be used to search the system for file/directories that meet any number of criteria. When found, the **xargs** command can be used to look deeper within those entities for other values (such as in conjunction with **grep**).

23. The **which** command will tell where the first executable by a given name can be found by logically moving through your path statement. The **locate** command looks for matching entries in the locatedb database and can help you find files on your system. This database can be updated by running the **updatedb** command (which can be automatically configured via the /etc/updatedb.conf file).

Boot, Initialization, Shutdown, Run Levels

24. The Linux Loader (lilo) is used to boot the system, and can allow you to choose other operating systems to boot into. lilo can be configured through the /etc/lilo.conf files and it can load additional modules identified in either /etc/conf.modules or /etc/modules.conf.

25. The "system" log file is /var/log/messages and this is where the majority of events are written to by the system log daemon (syslogd). Messages related specifically to lilo can be viewed with the **dmesg** command.

26. The init daemon is responsible for maintaining proper running of daemons at specified runlevels. The system attempts to go to the runlevel specified as the default in the /etc/inittab file upon each boot.

27. Runlevels can be changed with the **init** and **shutdown** (also known as **halt**) commands. The valid runlevels defined as standards are:

- 0 - powered off
- 1 - single user mode
- 2 - multiple user mode without NFS
- 3 - multiple user mode with NFS
- 5 - X environment
- 6 - reboot

Documentation

28. Manual files exist for most standard utilities and can be read using the **man** command. The pages are displayed one screen at a time using **more**, or **less**, or any similar utility defined by the PAGER variable.

29. There are a number of subdirectories beneath /usr/man that hold the manual pages. The most important are:

- /man1 - holds definitions for shell utilities and user commands
- /man2 - holds pages for system calls
- /man3 - libc calls
- /man4 - device descriptions
- /man5 - descriptions of configurable files, such as those found beneath /etc

- /man6 - manual pages for games
- /man7 - Linux system files and conventions
- /man8 - administrative utility definitions

30. To see a man page other than in the first directory a match is found, you can use the man command followed by the subdirectory number. For example: **man 2 nice**

31. Third party software should write documentation in the /usr/doc directory; creating a subdirectory that is the name of the utility, followed by a hyphen (-), and the version number.

32. The **whatis** command will show what manual pages are available for an entry (apropos allows you to find noncomplete matches), while **whereis** will show the location of the file and all related files (including any manual pages).
33. Many standard utilities allow you to enter the executable name followed by "**--help**" to obtain help only on the syntax. The **info** command will show help files stored beneath /usr/info.

34. A great deal of additional documentation can be found on the Internet at the Linux Documentation Project and other sites (including individual vendors). You can also find help to problems by subscribing to newsgroups and mailing lists.

35. It is important to document fully all changes you make to the system and to make documentation, and other forms of support available to all users.
Administratative Tasks

36. In order for a user to access the system, they must have a username and password. The username is stored in the /etc/passwd file as the first field. Other fields of this file are:

- second - this can hold the user's password if it has not been moved to /etc/shadow
- third - the user's unique ID number. The root user is always 0, and system accounts typically have low numbers. Standard users typically start at 500
- fourth - the group ID number. The root group is number 0, and system groups have low numbers
- fifth - free text describing the user that is returned by finger queries and similar utilities
- sixth - the user's home directory
- seventh - the user's shell. If left blank, the default shell is applied

37. Passwords can be moved to the /etc/shadow file for greater security (with the **pwconv** utility - the opposite of which is **pwunconv**). When this is done, an "x" appears in the second field of the /etc/passwd file. Passwords are changed with the **passwd** command.

38. Users can be added by manually editing the configuration files, or using the **useradd** command (and they can be removed by using **userdel**). Adding a user in this way will also create their home directory and copy to it the template files found beneath /usr/skel.

39. Group definitions exist in the /etc/group file. Passwords can also be there, or they can be moved to /etc/gshadow for greater security (with the **grpconv** utility - the opposite of which is **grpunconv**). The **groupadd** utility can be used to create groups (and avoid manual editing of the configuration files), and **groupdel** can be used to remove groups. Groups can be modified with **groupmod**, and users can change between groups with the **newgrp** command.

40. The /etc/profile file holds variables that you want to apply to all users. When the users login, this file runs and sets up their environment before going to any individual .profiles that may exist beneath their home directories. To see the last time each user logged onto the system, use the **lastlog** command, which extracts its data from the lastlog file. To just see who logged on most recently, and may currently be on (the latter of which **who** will show), you can use the **last** command.

41. The system log daemon (syslogd) logs most events to the log files, with the main file being /var/log/messages. The **logrotate** command can be configured to automatically archive these logs and perform maintenance as configured in the /etc/syslog.conf file. You can manually write entries to the log files using the **logger** command.

42. To schedule a job to run in unattended mode one time only, you can use the **at** command. Scheduled jobs can be viewed with the **atq** command and deleted prior to execution with **atrm**.

43. Restrictions can be placed on who can use the at service (atd) by creating an at.allow file and only placing valid user names beneath there. The opposite of this is that you can create an at.deny file and place within it only the names of users who cannot use the service (meaning that everyone else can).

44. If you need to schedule an unattended job to run at any sort of regular interval, you can create a crontab (cron table) entry for it. The crontab files are read by the cron service, which looks every minute to see if any jobs need run.

45. Restrictions can be placed on who can use cron by creating a cron.allow file and only placing valid user names beneath there. The opposite of this is that you can create a cron.deny file and place within it only the names of users who cannot use the service (meaning that everyone else can).

46. There are six fields to each entry in the cron tables:

- the minute the job is to run (0-59)
- the hour the job is to run (0-23)
- the day of the month (1-31)
- the month of the year (1-12)
- the day of the week (0-6, with 0 equal to Sunday)
- the path to the executable that is to run

47. Backups must be done in order to provide insurance for your data in the event of a catastrophe. Backups can be done with the **tar** command (which can combine multiple files into a single archive) or **cpio** which can copy files in and out between media.

48. In addition to archiving, you can also compress files with the **compress** command (counter is **uncompress**), **gzip** (counter is **gunzip**), or **pack** (counter is **unpack**).

49. Backups can be full, or partial. If they are partial, they can be incremental (everything since the last full or incremental backup) or differential (everything since the last full backup). Differentials and incrementals cannot be mixed and matched, and you must use one or the other.
50. As important as backups are, it is equally important that you verify that restores can be done on a regular basis. Only by testing the quality of the backups, and the restore procedure can you verify that you will be ready when an emergency does occur.