LPI Level 1, Exam 102

Linux Professional Institute Exam 102 Objectives are at
http://www.lpi.org/en/obj_102.html

$Revision: 67 $
$Date: 2005-02-07 08:16:41 -0500 (Mon, 07 Feb 2005) $

105: Kernel

105.1: Manage/Query kernel and kernel modules at runtime

See *LPI Linux Certification in a Nutshell*, p. 300. LPI test weight: 4.

Working with modules

- Modules are located in `/lib/modules/`*kernel-version*
- Module dependency file `/lib/modules/`*kernel-version*`/modules.dep`
- `lsmod` – list loaded modules. Displays name, size in bytes, use count, and referring modules.
- `insmod` *modulename* – Insert a module into the running kernel.
  - `insmod -s` – display results to *syslog* instead of `stdout`.
  - `insmod -v` – sets verbose mode.
- `depmod` – Writes module dependency information to `/lib/modules/`*kernel-version*`/modules.dep`
- `modinfo` *modulename* – displays helpful module information
  - `modinfo -a` – displays module author.
  - `modinfo -d` – displays modules's description.
  - `modinfo -P` – displays the typed parameters that module supports.
- `modprobe` *modulename* – determines module dependencies and install prerequisites.
  - `modprobe -r` *modulename* – removes module.
  - `modprobe -s` – display results to *syslog* instead of `stdout`.
  - `modprobe -v` – sets verbose mode.
  - `modprobe -a` – loads all modules.
  - `modprobe -c` – displays a complete module configuration.
  - `modprobe -l` – list modules.
  - `modprobe -t` *tag*– confine to a certain kernel tag such as `net`, `block`, `cdrom`, `fs`, `ipv4`, `misc`.
- `rmmod` *modulename* – removes a loaded module.
  - `rmmod -a` – removes all unused modules.
  - `rmmod -s` – display results to *syslog* instead of `stdout`.

modules.conf file

- Module configuration storage: `/etc/modules.conf` or `/etc/conf.modules`
- `/etc/modules.conf` directives:
  - `#` – Comment lines
  - `keep` – keep default paths
  - `depfile` *depfilelocation* – overrides dependency file location `modules.dep`
  - `path` *path* – directory to search for modules.
  - `options` *modulename opt1=val1 opt2=val2* – specify modules configuration option defaults.
  - `alias` *aliasname targetmodule* – alias a generic name to a specific module.
  - `pre-install` *module shellcommand* – run a shell command before installing module *modulename*
  - `install` *module shellcommand* – override the default module-insertion command `insmod` for another
  - `post-install` *module shellcommand* – run a shell command after installing module *modulename*
  - `pre-remove` *module shellcommand* – run a shell command before removing module *modulename*
  - `remove` *module shellcommand* – override the default module-removal command `rmmod` for another
  - `post-remove` *module shellcommand* – run a shell command after removing module *modulename*

uname

- `uname` – prints system information
  - `uname -a` – All information in the order:
  - `uname -s` – kernel name (Linux)
  - `uname -n` – Hostname (rsifox)
  - `uname -r` – Kernel release (2.6.10)
  - `uname -v` – Kernel version (#1 SMP Tue Dec 28 15:50:14 EST 2004)
  - `uname -m` – Machine (i686)
  - `uname -p` – Processor type (Mobile Intel(R) Pentium(R) 4 - M CPU 2.00GHz)
  - `uname -i` – Hardware platform (GenuineIntel)
  - `uname -o` – Operating system (GNU/Linux)

105.2: Reconfigure, build, and install a custom kernel and kernel modules

See *LPI Linux Certification in a Nutshell*, p. 310. LPI test weight: 3.

Kernel versioning

- Kernels are numbered as *major.minor.patchlevel*
- major releases (currently 2.x.x)
- minor releases (currently 2.6.x is stable). Typically even numbered kernels are stable and odd numbered minor kernels are development kernels. The 2.5 kernels were dev kernels.
- patch level (currently at 2.6.10 as stable)

Kernel sources and configuration

- Kernel sources are stored in sub-directories of `/usr/src/`
- You can get a kernel from `http://www.kernel.org` and unpack it in `/usr/src` or download an .rpm or .deb of the kernel-sources.
- Typically the running kernel's source is a symlink `/usr/src/linux`
- The kernel configuration settings are stored in `/usr/src/linux-x.y.z/.config`
- Some systems also have the compiled .config options for the running kernel in `/proc/config.gz`
- Options for configuring a kernel:
  - edit `.config` with a text editor
  - run `make config` in the kernel source directory and answer Y/N/M (yes, no, module) for each option to compile in.
  - run `make oldconfig` and a new .config default will be made, or the last customized .config will be used.
  - run `make menuconfig` – A hacked version of cdialog text UI for configuration.
  - run `make xconfig` – Runs an X-Windows menu UI for configuration.

Steps for kernel compilation

- Run `make dep` – Source file dependencies are checked and a `.depend` file is built.
- Run `make clean` – Removes old output files are prepares for new compilation.
- Run `make bzImage` – Builds a bootable kernel image file.
- Run `make modules` – Builds all module (.o) files.
- Run `make modules_install` – Loadable modules are installed to `/lib/modules/x.y.z`
- `cp /usr/src/linux/arch/i368/boot/bzImage /boot/kernel-x.y.x` to copy file to `/boot` directory.
- Finally modify `/etc/lilo.conf` and run `lilo` or edit `/boot/grub/grub.conf`.

106: Boot, Initialization, Shutdown, and Runlevels

106.1: Boot the system

See *LPI Linux Certification in a Nutshell*, p. 134. LPI test weight: 3.

- kernels can accept parameters on boot. Most used is `root=/dev/hdaX`
- The kernel gives detailed progress on the console during boot and are stored to disk as soon as a disk and logger comes available.
- dmesg – send kernel ring buffer to `stdout`
- dmesg -n 1 – Show only kernel panics on console.
- More logging is shown in `/var/log/messages`

106.2: Change runlevels and shutdown or reboot the system

See *LPI Linux Certification in a Nutshell*, p. 137. LPI test weight: 3.

- init run by kernel – pid 1
- Chnage runlevel with telinit.
- runlevels:
  - 0 – shutdown now
  - 1 – single user maintenance
  - 2 – multiuser no nfs
  - 3 – multiuser text mode (command line)
  - 5 – GUI starts X
  - 6 – Reboot immediately
- nicer way sends messages to users: `shutdown`
- edit files in `\etc\inittab`
- `\etc\inittab` has line with `initdefault`
- Check current runlevel with `runlevel`. Output is *previousRL currentRL*. If just booted *previousRL* will be `N`.

107: Printing

Printing overview

- `/etc/printcap` – Printer capabilities file
- `lpd` – BSD line printer daemon
- `lpr` – Pipe print job to line printer
- `lpq` – Queries and displays queue information
- `lprm` – removes a jobid from the print queue.
- `lpq` – Line printer control (superuser control)

- Even advanced print systems like CUPS provide old BSD compatibility.

107.2: Manage printers and print queries

See *LPI Linux Certification in a Nutshell*, p. 324. LPI test weight: 1.

- `lpq -P`*printername* – query a particular printer
- `lpq -L` – Long format
- `lpq` *username* – query a particular users jobs
- `lprm -` – remove all of your print jobs
- `lprm -P`*printername* – remove all of your print jobs on a particular printer.
- `lpc` has an interactive mode if called w/o command line options.
  - `lpc abort all` – Terminates printing NOW on all printers.
  - `lpc abort` *printername* – Terminates printing NOW on a specific printer.
  - `lpc disable` – Stops any additional jobs from queueing.
  - `lpc enable` – Allows additional jobs to queue.
  - `lpc down` *message* – `disable`, `stop`, and provide message for users as why the printer is down.
  - `lpc restart` – Restarts `lpd` daemon.
  - `lpc stop` – Stops printing after current job.
  - `lpc start` – Start printing queue.
  - `lpc status` – Display queue status.
  - `topq` *name jobs* – Push jobs to the top of the waiting queue *name*
  - `up` – `enable` & `start` a print queue.

107.3: Print files

See *LPI Linux Certification in a Nutshell*, p. 331. LPI test weight: 1.

- `lpr` – Print a file
- `lpr -P`*printername* – Print a file to a specific queue.
- `lpr -#`*count* – Send *count* copies.
- `lpr -s` – symbolic link the print job.

107.4: Install and configure local and remote printers

See *LPI Linux Certification in a Nutshell*, p. 332. LPI test weight: 1.

- Colon seperated options
- `if` – input filter
- `of` – output filter
- `lp` – printer device for local printer
- `rp` – remote spool name
- `rm` – remote machine name
- `mx` – max size of print job in blocks
- `sd` – spool directory for print jobs
- `sh` – suppress header pages

108: Documentation

108.1: Use and manage local system documentation

See *LPI Linux Certification in a Nutshell*, p. 147. LPI test weight: 4.

- Lots of plain text docs in `/usr/share/doc` (`/usr/doc` is a symlink)
- Use `less` pager to look at plain text
- `less` keystrokes
  - space – forward one screen
  - D – forward one-half screen
  - Return – forward one line
  - B – backward one screen
  - U – backward one-half screen
  - Y – backward one line
  - g – top of the file
  - G – tail of the file
  - `\` – Search forwards
  - `?` – Search backwards
  - H – help screen
  - :n – next file on command line
  - :p – previous file on command line
- `man` – display manpages
- `man -a` – display all manpages of that name
- `man -w` – print location of manpages
- `man 3` – find man in a particular section
- man page sections

1	Executable programs programs
8	System Administration commands
2	System calls
3	Library calls
4	Special files (usually in `/dev`
5	File formats and conventions
6	Games
7	Macro packages and conventions
9	Kernel routines
N	Tcl/Tk

- GNU `info` pages

Tab	Move along hypertext links
Enter	Follow hypertext
d	Return to top node
?	List all info commands
p and n	move to previous or next page
u	move up one level in the Texinfo hierarchy
q	quit
h	show `info` primer
/string	string search

- `whatis` – Find a short description (from man pages) for a command.
- `apropos` – Search the whatis database for a string
- `makewhatis` – Build the whatis database

108.2: Find Linux documentation on the Internet

See *LPI Linux Certification in a Nutshell*, p. 155. LPI test weight: 3.

- Linux Documentation Project `http://www.tldp.org`
- Usenet Newsgroups
  - *comp.os.linux*
  - *comp.os.linux.answers*
  - *comp.os.linux.networking*
  - Archives in `http://groups.google.com`
  - Websites of specific projects (such as `http://www.x.org`
- Mailing Lists
- Vendor websites

108.5: Notify users on system-related issues

Not in LPI Linux Certification in a Nutshell LPI test weight: 1.

- `/etc/issue` – pre-login message and identification file for gettys
- `/etc/issue.net` – The file /etc/issue.net is a text file which contains a message or system identification to be printed before the login prompt of a telnet session.
- `/etc/motd` – The contents of /etc/motd are displayed by login(1) after a successful login but just before it executes the login shell.

109: Shells, Scripting, Programming, and Compiling

109.1: Customize and use the shell environment

See *LPI Linux Certification in a Nutshell*, p. 340. LPI test weight: 5.

- Bash shell environmental variables
  - PATH – List of directories which the shell looks for executables.
  - HOME – Your home directory
  - USERNAME – Your username
  - TERM – Type of terminal
- bash files
  - `~/.bash_history` – History storage
  - `~/.bash_profile` – Bash runs with login instance.
  - `~/.bashrc` – Bash runs with each bash subshell
  - `~/.profile` – Bash runs if no `/.bash_profile`
  - `~/.bash_logout` – Bash runs upon logout.
  - `~/.inputrc` – Defines optional key-bindings for bash.
- To look at current shell variables, run bash built-in `env`
- To create a new shell variable, use equal sign. `#PI=3.141592`
- To get the command line to print a variable, echo it: `echo $PI`
- To use a local variable in other shells, use export: `export PI`
- To alias a command use alias: `alias more='less'`
- To make a function in bash:
  `function MYFUNC(){ command-list; }`
- Modify many many BASH options with `set` builtin.

109.2: Customize or write simple scripts

See *LPI Linux Certification in a Nutshell*, p. 351. LPI test weight: 3.

- Bash scripts should be prepended with `#!/bin/bash`
- Bash Built-in variables

`$1-$N`	Stores the arguments (variables) that were passed to the shell program from the command line.
`$?`	Stores the exit value of the last command that was executed.
`$0`	Stores the first word of the entered command (the name of the shell program.
`$*`	Stores all the arguments that were entered on the command line (`$1` `$2` ...).
`"$@"`	Stores all the arguments that were entered on the command line, individually quoted (`"$1"` `"$2"` ...).

- Status variable `$?` – Graceful exits normally are zeros.
- Builtin `test` *expression* or `[ expression ]`
- The `test` structures:
  - File comparisons:

`-d`	Returns True if file, filename is a directory.
`-f`	Returns True if file, filename is an ordinary file.
`-r`	Returns True if file, filename can be read by the process.
`-s`	Returns True if file, filename has a nonzero length.
`-w`	Returns True if file, filename can be written by the process.
`-x`	Returns True if file, filename is executable.

  - String comparisons:

`str1 = str2` Returns True if str1 is identical to str2.
`str1 != str2` Returns True if str1 is not identical to str2.
`str` Returns True if str is not null.
`-n str` Returns True if the length of str is greater than zero.
`-z str` Returns True if the length of str is equal to zero. (zero is different than null)

- Numeric comparisons:
  `int1 -eq int2` Returns True if int1 is equal to int2.
  `int1 -ge int2` Returns True if int1 is greater than or equal to int2.
  `int1 -gt int2` Returns True if int1 is greater than int2.
  `int1 -le int2` Returns True if int1 is less than or equal to int2
  `int1 -lt int2` Returns True if int1 is less than int2
  `int1 -ne int2` Returns True if int1 is not equal to int2
- Comparing expressions:
  `!expression` Returns true if expression is not true
  `expr1 -a expr2` Returns True if expr1 and expr2 are true. `&&` and operator.
  `expr1 -o expr2` Returns True if expr1 or expr2 is true. `||` or operator.

- if structure
```
if [ expression ]
        then
                commands
fi
```
- if..then...else structure
```
if [ expression ]
        then
                commands
        else
                commands
fi
```
- If..then...else If...else
```
if [ expression ]
        then
                commands
elif [ expression2 ]
        then
                commands
else
                commands
fi
```
- case structure
```
case string1 in
        str1)
                commands;;
        str2)
                commands;;
        *)
                commands;;
esac
```
string1 is compared to str1 and str2. If one of these strings matches string1, the commands up until the double semicolon (; ;) are executed. If neither str1 nor str2 matches string1, the commands associated with the asterisk are executed. This is the default case condition because the asterisk matches all strings.

- for do done loop
```
for var1 in list
do
        commands
done
```
- while loop
```
while [ expression ]
do
        commands
done
```
- until loop
```
until [ expression ]
do
        commands
done
```
- functions with arguments
```
fname2 (arg1,arg2...argN){
        commands
}
```
- `break [n]` – exits from the innermost for while or until loop or from *n* levels of loop.
- `continue [n]` – skip remaining commands in form while or until loop and go to next iteration (or skip *n* loops)
- `echo [string]` – echo string to command line
- `exit [n]` – exit a shell with status *n*. The value of *n* can be 0 (success) or non-zero (failure).
- `read variable1 [variable2]` – Read a line from `stdin` and assign to a variable.
- `shift [n]` – Shift positional parameters down *n* elements.
- `source` – Read and execute the lines in file.
- Mailing results from a script – Just pipe a message to the handy `mail` command (`-s` option provides an optional subject line)
```
echo "Error! Does not Compute!" |mail -s "Problem" root
```

111: Administrative Tasks

111.1: Manage users and group accounts and related system files

See *LPI Linux Certification in a Nutshell*, p. 164. LPI test weight: 4.

- usernames, password, uid, gid, User's full name, home dir, and default shell saved in `/etc/passwd` colon delimited file.
- Shadow password normally used and represented by x in `/etc/passwd` but actually stored in `/etc/shadow`.
- uid, gid are non-negative integers
- Group information is stored in `/etc/group`
- group name, grouppassword, gid, member list are saved in `/etc/group` in colon delimited format.
- Normal user programs must be able to read `/etc/passwd` and `/etc/group` so passwords are put in the root-read-only `/etc/shadow` and `/etc/gshadow` files.
- Manual editing of `/etc/passwd` is with `vipw`
- Manual editing of `/etc/shadow` is with `vipw -s`
- Manual editing of `/etc/group` is with `vigr`
- Manual editing of `/etc/gshadow` is with `vigr -s`
- `useradd` *user* – Add account for *user* on the system.
  - `useradd -c "Edward Corrado" ecorrado` – comment field
  - `useradd -d /home/ecorrado ecorrado` – home directory
  - `useradd -S` – display (or change) useradd defaults
  - `useradd -s /bin/zsh ecorrado` – change shell
  - `useradd -m ecorrado` – build a home directory from `/etc/skel`
- `usermod` *username* – modify *username's* account
  - `usermod -L ecorrado` – Lock user's password preventing login
  - `usermod -U ecorrado` – Unlock the user's password, enabling
  - `usermod -s tcsh ecorrado` – change user's shell the user to log into the system.
- `userdel` *username* – delete *username's* account
  - `userdel -r` *username* – delete *username's* account and his home directory.
- `groupadd` *groupname* – add a group to the system
- `groupmod` *groupname* – Modify a group entry
  - `groupmod -n` *newname oldgroupname* – Rename a groupname
- `groupdel` *groupname* – Delete a group. (Rarely done)
- `passwd` *username* – Interactively set the password a user.
  - `passwd -l` *username* – lock a password (su only)
- `gpasswd` *groupname* – Interactively set a group password.
- `pwconv` – convert a standard `/etc/passwd` file to a shadow password combination
- `pwunconv` – revert from shadow pw to a standard password file.
- `grpconv` – convert to a group and shadow group config.
- `grpunconv` – revert from a shadow group configuration to a standard group file.
- `chage` *user* – modify password aging and expiration settings for *user*.
  - `chage -E` *expiredate user* – Set expire date (format is MM/DD/YY or MM/DD/YYYY).
  - `chage -l` *user* – display pw expiry for a user.

111.2: Tune the user environment and system environment variables

See *LPI Linux Certification in a Nutshell*, p. 174. LPI test weight: 3.

- `/etc/profile` – System wide shell configuration script for bash (default PATH is here).
  - an excellent place to set and export a default PATH, PS1, HOSTNAME, HISTSIZE, HISTFILESIZE, USER, LOGNAME, MAIL, INPUTRC
  - an excellent place to put a default umask
- `/etc/skel` – Skeleton user directory for new users. Typically has a `.bash_profile`, `.bashrc`, `.bash_logout`.

111.3: Configure and use system log files to meet adminstrative and security needs

See *LPI Linux Certification in a Nutshell*, p. 176. LPI test weight: 3.

- `syslogd` – System log daemon displays and records system messages.
- Most log files live in `/var/log`
- `/etc/syslog.conf` – configuration file for syslogd
- Each line of `syslog` has entries *facility.level action*.
- Facility – the creator of the message: `auth`, `authpriv`, `cron`, `daemon`, `kern`, `lpr`, `mail`, `mark`, `news`, `syslog`, `user`, `local0`, `local1`, ... `local7`.
- Level – the urgency of the message: `debug`, `info`, `notice`, `warning`, `err`, `crit`, `alert`, or `emerg`. Special level `none` disables a facility. A `*` can denote all levels.
- action – Destination for the messages that match *facility.level*|. Options: a filename, a hostname preceded by the `@` sign, or a comma

seperated list of users (or an `*`) to alert.
- use the `logger` command to send a log message to syslogd. Ex: `logger -p local5.info "Script complete. Sleep well, admin."`
- `logrotate` – utility (typically called by cron) to rotate log files.
- `/etc/logrotate.conf` – configuration file that describes specific files to rotate.
- Examine log files with less, or follow them with tail -f, or search them with grep.

111.4: Automate system administration tasks by scheduling jobs to run in the future

See *LPI Linux Certification in a Nutshell*, p. 180. LPI test weight: 4.

- the `crond` daemon allows programs to be run periodically.
- `crontab` – Allows users to modify their cron table.
  - `crontab -e` – interactive editing of crontab
  - `crontab -l` – list the user's crontab
  - `crontab -r` – remove the crontab file
  - `crontab -u jdoe` – operate on another user's crontab.
  - crontab fields: *minute hour day month dayofweek command*
    - minute (0-59)
    - hour (0-23)
    - day of the month (1 to 31)
    - day of the week (0 to 6 from Sun to Sat). Mneumonic Sunday is Nonesday.
    - Command with options
  - `/etc/crontab` holds system crontabs, which include a username before the command.
    minute hour day month dayofweek runasuser command
  - System crontab often used to run admin scripts in `/etc/cron.hourly/`, `/etc/cron.daily/`, `/etc/cron.weekly/`, `/etc/cron.monthly/` directories.
  - `at` –

111.5: Maintain an effective data backup strategy

See *LPI Linux Certification in a Nutshell*, p. 184. LPI test weight: 3.

111.6: Maintain system time

Not in LPI Linux Certification in a Nutshell LPI test weight: 4.

112: Networking Fundamentals

112.1: Fundamentals of TCP/IP

See *LPI Linux Certification in a Nutshell*, p. 389. LPI test weight: 4.

112.3: TCP/IP configuration and troubleshooting

See *LPI Linux Certification in a Nutshell*, p. 400. LPI test weight: 7.

112.4: Configure Linux as a PPP client

See *LPI Linux Certification in a Nutshell*, p. 414. LPI test weight: 3.

113: Networking Services

113.1: Configure and manage inetd, xinetd, and related services

See *LPI Linux Certification in a Nutshell*, p. 425. LPI test weight: 3.

113.2: Operate and perform basic configuration of sendmail

See *LPI Linux Certification in a Nutshell*, p. 429. LPI test weight: 4.

113.3: Operate and perform basic configuration of Apache

See *LPI Linux Certification in a Nutshell*, p. 432. LPI test weight: 4.

113.4: Properly manage the NFS, smb, and nmb daemons

See *LPI Linux Certification in a Nutshell*, p. 434. LPI test weight: 4.

113.5: Setup and configure basic DNS services

See *LPI Linux Certification in a Nutshell*, p. 439. LPI test weight: 4.

113.7: Set up secure shell (OpenSSH)

Not in LPI Linux Certification in a Nutshell LPI test weight: 4.

114: Security

114.1: Perform security administration tasks

See *LPI Linux Certification in a Nutshell*, p. 446. LPI test weight: 4.

114.2: Setup host security

See *LPI Linux Certification in a Nutshell*, p. 458. LPI test weight: 3.

114.3: Setup user level security

See *LPI Linux Certification in a Nutshell*, p. 460. LPI test weight: 1.

`$Revision: 67 $, $Date: 2005-02-07 08:16:41 -0500 (Mon, 07 Feb 2005) $.`
http://www.drfox.com