Introduction to Linux

Training Material

License

Contents

Contents

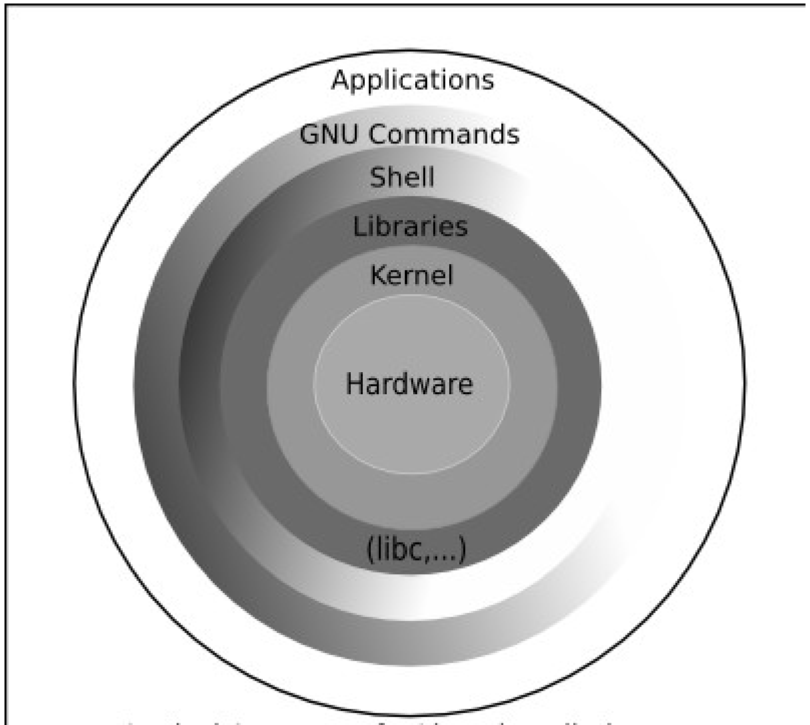
Contents

Contents

Introduction

What is Linux?

- Kernel
  - 2.4.x
  - 2.6.x
- Distributions
  - Red Hat
  - Novell / SuSE
  - Debian
- GNU
  - compilers
  - libraries
  - editors and other tools



Logical Anatomy of a Linux installation

A brief history of Linux

Sep 1983 – Richard Stallman announces the GNU Project

Apr 1991 – Linus Torvalds announces he's working on a hobby OS

Sep 1991 – Linux Kernel 0.01

Mar 1994 – Linux Kernel 1.0 (i386)

Mar 1995 – Linux Kernel 1.2 (Alpha, Mips, Sparc)

June 1996 – Linux Kernel 2.0 (SMP, Tux the Penguin)

Jan 1999 – Linux Kernel 2.2 (64-bit, FAT32, NTFS)

Jan 2001 – Linux Kernel 2.4 (ISA PnP, PA-RISC, USB, PC Card)

Dec 2003 – Linux Kernel 2.6 (IA64, x86_64, em64t, embedded systems, NUMA)

Linux distributions (1/2)

- Red Hat
  - Enterprise Linux (AS, ES, WS, Desktop) v4
  - Enterprise Linux (AS, ES, WS) v3
  - Fedora Core
- Debian
  - Stable (3.1 - *Sarge*)
  - Testing (*Etch*)
  - Unstable (*Sid*)
  - Derivatives (Ubuntu, Xandros, Knoppix, Progeny)

Linux distributions (2/2)

- Novell / SuSE
  - SuSE Linux Enterprise Server 10
  - SuSE Linux Enterprise Server 9 (SP3)
  - SuSE Linux Enterprise Desktop 10
  - Novell Linux Desktop 9
  - OpenSuSE 10.1
- Others
  - Gentoo
  - Mandriva
  - Ubuntu
  - Sun Java Desktop System
  - Slackware
  - Turbolinux

Who is using Linux?

- Dot coms
  - Google
  - Amazon
  - Paypal
- Financial
  - Irish Stock Exchange
  - First Trust Corporation
  - Central Bank of India
- Entertainment
  - Ticketmaster
  - Pixar
  - Industrial Light and Magic

Getting Started

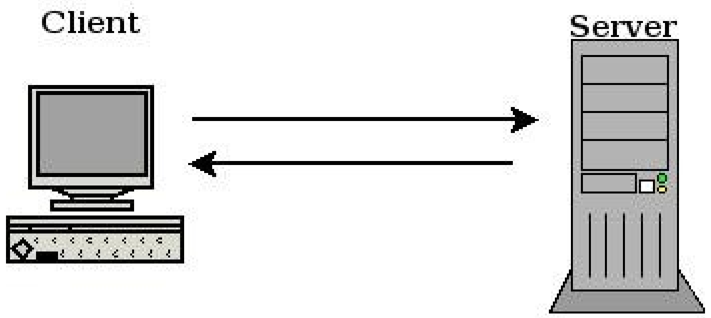
System Accounts

- Users
  - uid
  - username
  - password
- Groups
  - gid
- Files
  - /etc/passwd and /etc/shadow
  - /etc/group and /etc/gshadow
- Other authentication mechanisms
  - LDAP
  - Kerberos

Account Settings

- Passwords
  - choosing good passwords
  - password expiry
  - password security
- Shell
- Home directories

Connecting



- Local
  - the console
- Remote
  - telnet
    - *telnet <hostname> [port]*
  - ssh
    - *ssh <username>@<hostname>*
  - rlogin, rsh
    - *rlogin -l <username> <hostname>*

Copying files over the network

- ftp
  - operation
  - transfer mode
  - anonymous ftp
  - plaintext usernames and passwords
- sftp
  - same interface as ftp
  - encrypted usernames and passwords
- scp
- rcp

System documentation (1/2)

- Distribution-specific documentation
  - Administration and install guides
  - User manuals
  - Reference guides
  - Release Notes
- man pages and the man command
  - *man [section] <page>*
  - *man -k <topic>*
  - *man -f <topic>* or *apropos <topic>*
  - *man man!*

System documentation (2/2)

- GNU info
- <command> -h, --help
- The Linux Documentation Project
  - http://www.tldp.org/
  - FAQs, HOWTOs, Guides
- Project specific documentation for Samba, Apache and others.

Exercise 1.1 – Getting started

1) Start up *telnet* on your local machine.
2) Start up *putty* on your local machine.
3) Login to the server using
   - ssh
   - telnet
4) Upload a test file from your local system to the server using each of the following methods,
   - ftp
   - sftp
   - scp

Exercise 1.2 – Getting started

5) Use man to retrieve information on each of the following:
    - the `mail` command
    - the `passwd` file (not the `passwd` command)
    - the `printf` system call (not the `printf` command)

6) Use info to review the *info primer*.

7) Find the following on TLDP:
    - *The Unix and Internet Fundamentals HOWTO*
    - *Advanced Bash-Scripting Guide*
    - *The Linux FAQ*

The Shell

What is the shell?

- Command-line interpreter
- Scripting environment
- Features
  - built-in commands
  - aliases
  - variables
  - pipes
  - input/output redirection
  - history
  - tab completion
- Shell commands and variables are <u>CASE SENSITIVE</u>

Choosing a shell

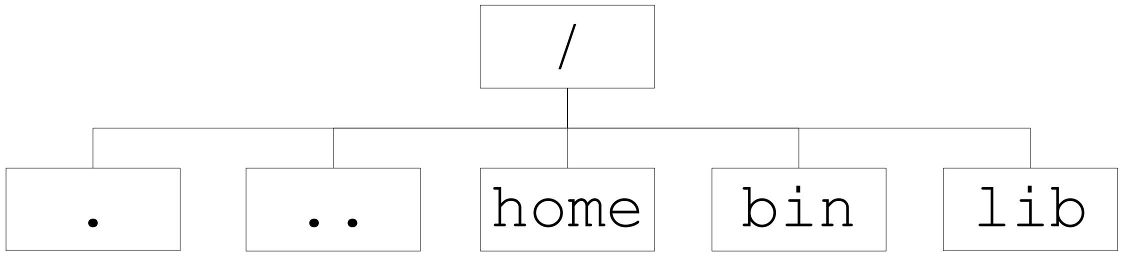
- Bourne shells
  - sh
  - bash
- C shells
  - csh
  - tcsh
- Korn Shell
- Others

Switching to a different shell

- Changing shells
  - `chsh`
  - `/etc/shells`
- Restricted shell
  - changing directory
  - changing shell
  - changing shell variables
- Changing user details
  - `chfn`
  - `.plan`

Navigating the filesystem

- The filesystem tree



- Commands
  - `pwd`
  - `cd`
  - `ls`
- Inodes

Shell Variables

- Shell variables
- Environment variables
- Viewing variables with `echo`
- Listing variables
  - `set`
  - `export`
  - `setenv`
  - `env`

Giving variables values

- In the bourne shell
  - `VAR=value`
  - `export VAR`
  - `export VAR=value`
- In the c-shell
  - `set VAR=value`
  - `setenv ENV value`
  - synchronised variables
- `unset`

Environment Variables

- `PATH`
- `MANPATH`
- `PS1` or `prompt`
- `HOME`
- `PRINTER`
- `TERM`
- `EDITOR`

Paths

- Absolute paths
  - `/bin/ls`
  - `/etc/passwd`
- Relative paths
  - `./ls`
  - `../ls`
  - `ls`
- Security problems with relative paths
- Use the `which` command to verify your PATH
- System administrators and scripts should always use absolute paths!

Some Useful Commands

- Printing output to the screen
  - `echo`
- Viewing files
  - `cat`
  - `more` or `less`
- Identifying files
  - `file`
- Finding commands
  - `which`
- Command-line options
  - short (`-h`)
  - long (`--help`)

Exercise 2.1 – The Shell

1) Navigate to the following directories and list the contents of the directory,
   - `/home`
   - `/usr/bin`
   - `/etc`
2) Use absolute paths to list the contents of those directories <u>without navigating to them first</u>.
3) Display the contents of the following variables and explain what they mean:
   - `SHELL`
   - `USER`
   - `PWD`

Exercise 2.2 – The Shell

4) Use a system command to determine what type of file or directory each of the following is:
   - `/`
   - `/etc/passwd`
   - `/bin/ls`
   - `/lib/libc-2.3.5.so`

5) View the contents of `/etc/passwd` with `cat`, `more` and `less`.

6) Change your `PATH` variable to "/" and try running `cat`, `more` and `less` again. Explain what is happening.

7) Change your `PS1` (`bash`) or `prompt` (tcsh) variable.

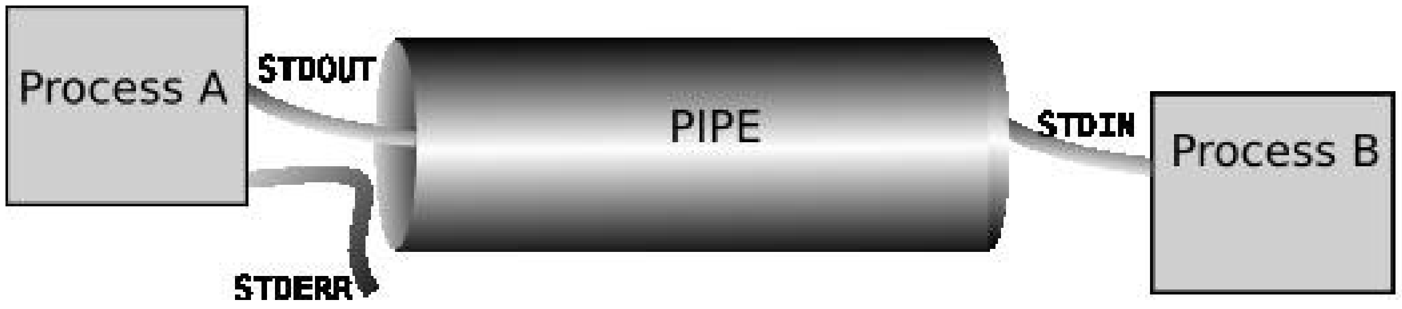
Advanced Shell Topics

I/O Redirection

- Standard input (file descriptor 0, stdin)
- Standard output (file descriptor 1, stdout)
- Standard error (file descriptor 2, stderr)
- Redirection operators
  - `[n]> file`
  - `<[n] file`
  - `[n]>> file`
- Redirections are processed in the order they appear, from left to right.

Pipes

- `|`
- chaining commands
- `xargs`
- named pipes (fifos)



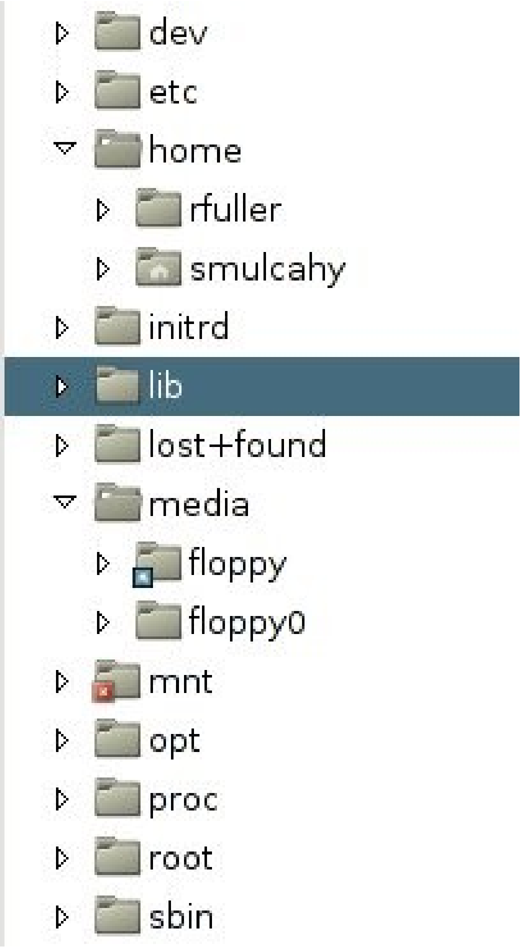
Exercise 3 – Advanced Shell

1) Write a command to list the contents of /etc/passwd to a file in /var/tmp (such as *mytest*).
2) Write a command to list the contents of /etc/group to the same file used in 1, <u>without overwriting the original content</u>.
3) Write a command to list the contents of /etc/passwd to a file in /var/tmp while redirecting any errors to a different file in /var/tmp.
4) Explain what the following will do (you may need to use the *man* command to check what a particular command does):
   - `cat /etc/passwd | tr a-z A-Z`
   - `ls /etc | sort | tr A-Z a-z`

Files

Linux Files

- Everything is a file!
  - Docs, pictures, executables
  - Directories
  - Devices
  - Kernel internals
  - .dot files
  - hard links
  - soft (symbolic) links
- Filesystem Hierarchy Standard (FHS)
- Hard and soft links



File Hierarchy Standard - /

- /
- /bin
- /boot
- /dev
- /etc
- /home
- /lib

File Hierarchy Standard - /

- /mnt (and the mount command)
- /opt
- /root
- /sbin
- /tmp
- /usr
- /var

File Hierarchy Standard - /usr

- /usr/bin
- /usr/include
- /usr/lib
- /usr/local
- /usr/sbin
- /usr/share
- /usr/src

File Hierarchy Standard - /var

- /var/cache
- /var/lib
- /var/lock
- /var/log
- /var/mail
- /var/opt
- /var/run
- /var/spool
- /var/tmp

ls command

```
ls -la /bin
```

```
total 3404
drwxr-xr-x   2 root root   4096 Oct 13 18:46 .
drwxr-xr-x  25 root root   4096 Sep  3 20:05 ..
-rwxr-xr-x   1 root root   2684 Dec 24  2002 arch
-rwxr-xr-x   1 root root  82312 Apr  3  2002 ash
-rwxr-xr-x   1 root root 511400 Apr  8  2002 bash
-rwxr-xr-x   1 root root  16504 Jul 16 12:37 cat
-rwxr-xr-x   1 root root  31404 Jul 16 12:37 chgrp
-rwxr-xr-x   1 root root  31212 Jul 16 12:37 chmod
-rwxr-xr-x   1 root root  34572 Jul 16 12:37 chown
-rwxr-xr-x   1 root root  51212 Jul 16 12:37 cp
-rwxr-xr-x   1 root root  49092 Nov 24  2001 cpio
```

ls command

permissions

```
total 3404
drwxr-xr-x   2 root root   4096 Oct 13 18:46 .
drwxr-xr-x  25 root root   4096 Sep  3 20:05 ..
-rwxr-xr-x   1 root root   2684 Dec 24  2002 arch
-rwxr-xr-x   1 root root  82312 Apr  3  2002 ash
-rwxr-xr-x   1 root root 511400 Apr  8  2002 bash
-rwxr-xr-x   1 root root  16504 Jul 16 12:37 cat
-rwxr-xr-x   1 root root  31404 Jul 16 12:37 chgrp
-rwxr-xr-x   1 root root  31212 Jul 16 12:37 chmod
-rwxr-xr-x   1 root root  34572 Jul 16 12:37 chown
-rwxr-xr-x   1 root root  51212 Jul 16 12:37 cp
-rwxr-xr-x   1 root root  49092 Nov 24  2001 cpio
```

ls command

owner and group

```
total 3404
drwxr-xr-x   2 root root   4096 Oct 13 18:46 .
drwxr-xr-x  25 root root   4096 Sep  3 20:05 ..
-rwxr-xr-x   1 root root   2684 Dec 24  2002 arch
-rwxr-xr-x   1 root root  82312 Apr  3  2002 ash
-rwxr-xr-x   1 root root 511400 Apr  8  2002 bash
-rwxr-xr-x   1 root root  16504 Jul 16 12:37 cat
-rwxr-xr-x   1 root root  31404 Jul 16 12:37 chgrp
-rwxr-xr-x   1 root root  31212 Jul 16 12:37 chmod
-rwxr-xr-x   1 root root  34572 Jul 16 12:37 chown
-rwxr-xr-x   1 root root  51212 Jul 16 12:37 cp
-rwxr-xr-x   1 root root  49092 Nov 24  2001 cpio
```

ls command

size in bytes

```
total 3404
drwxr-xr-x   2 root root   4096 Oct 13 18:46 .
drwxr-xr-x  25 root root   4096 Sep  3 20:05 ..
-rwxr-xr-x   1 root root   2684 Dec 24  2002 arch
-rwxr-xr-x   1 root root  82312 Apr  3  2002 ash
-rwxr-xr-x   1 root root 511400 Apr  8  2002 bash
-rwxr-xr-x   1 root root  16504 Jul 16 12:37 cat
-rwxr-xr-x   1 root root  31404 Jul 16 12:37 chgrp
-rwxr-xr-x   1 root root  31212 Jul 16 12:37 chmod
-rwxr-xr-x   1 root root  34572 Jul 16 12:37 chown
-rwxr-xr-x   1 root root  51212 Jul 16 12:37 cp
-rwxr-xr-x   1 root root  49092 Nov 24  2001 cpio
```

ls command

modification date (normally!)

```
total 3404
drwxr-xr-x   2 root root   4096 Oct 13 18:46 .
drwxr-xr-x  25 root root   4096 Sep  3 20:05 ..
-rwxr-xr-x   1 root root   2684 Dec 24  2002 arch
-rwxr-xr-x   1 root root  82312 Apr  3  2002 ash
-rwxr-xr-x   1 root root 511400 Apr  8  2002 bash
-rwxr-xr-x   1 root root  16504 Jul 16 12:37 cat
-rwxr-xr-x   1 root root  31404 Jul 16 12:37 chgrp
-rwxr-xr-x   1 root root  31212 Jul 16 12:37 chmod
-rwxr-xr-x   1 root root  34572 Jul 16 12:37 chown
-rwxr-xr-x   1 root root  51212 Jul 16 12:37 cp
-rwxr-xr-x   1 root root  49092 Nov 24  2001 cpio
```

ls command

file name

```
total 3404
drwxr-xr-x   2 root root   4096 Oct 13 18:46 .
drwxr-xr-x  25 root root   4096 Sep  3 20:05 ..
-rwxr-xr-x   1 root root   2684 Dec 24  2002 arch
-rwxr-xr-x   1 root root  82312 Apr  3  2002 ash
-rwxr-xr-x   1 root root 511400 Apr  8  2002 bash
-rwxr-xr-x   1 root root  16504 Jul 16 12:37 cat
-rwxr-xr-x   1 root root  31404 Jul 16 12:37 chgrp
-rwxr-xr-x   1 root root  31212 Jul 16 12:37 chmod
-rwxr-xr-x   1 root root  34572 Jul 16 12:37 chown
-rwxr-xr-x   1 root root  51212 Jul 16 12:37 cp
-rwxr-xr-x   1 root root  49092 Nov 24  2001 cpio
```

File permissions and ownership (1/2)

- Setting permissions
  - Symbolic mode

    ```
    chmod <groups> <add/remove/set> <permissions> <file>
    e.g.
    chmod u=rwx,g=rx,o=rx foo
    chmod u=r,g=r,o= foo
    ```

  - Octal mode

    ```
    chmod <mode> <file>
    e.g.
    chmod 0755 foo
    chmod 0440 foo
    ```

File permissions and ownership (2/2)

- Changing file ownership
  - chown
- Changing file group
  - chgrp
- Changing file attributes recursively with `-R`

- Special permissions
  - The sticky bit
  - The setuid/setgid bit
  - File owner

File & directory commands

- File creation and timestamp updates
  - `touch`
- Copying and moving files
  - `cp`
  - `mv`
- Removing files and directories
  - `rm`
- Directories
  - `mkdir`
  - `rmdir`

Filename substitution

- Metacharacters
  - *
  - ?
- Character ranges
  - [...]
  - -
  - ^
- Home directory shortcut
  - ~

Filename substitution (examples)

1. `cd ~`
2. `ls /usr/bin/b*`
3. `ls /usr/bin/*zip*`
4. `ls /usr/bin/b[abc]`
5. `ls /usr/bin/b[a-c]`
6. `ls /usr/bin/?grep`

Monitoring free space and inodes

- Files and directories
  - du
- Mounted filesystems
  - df
- -h option improves readability

Exercise 4.1 – Files

1) Create a file in your home directory containing a listing of all files in /bin that have filenames starting with “b”.
2) Create a file in your home directory containing a listing of all files in /bin that have filenames not starting with “b”.
3) Create a number of test files in your home directory with the following permissions (use a naming convention such as exercise3a, exercise 3b and so on):
   a) readable and writeable by you only
   b) readable and executable by you only
   c) readable, writeable and executable by you and readable and writeable by everyone else.

Exercise 4.2 – Files

4) Explain the permissions on the following files:
   - /bin/ls
   - /boot
   - /var/mail
5) Report the space used by your home directory and the free space available on your filesystems.
6) Where should large temporary files be placed according to the FHS?
7) Where would you expect a 3rd party application following the FHS to install its files to?

Processes

Processes and Threads

- Heavyweight process
- Lightweight process
- Speed of context switch
- Speed of creation
- Ease of sharing data
- Security
- NPTL

Shell job control

- foreground jobs
- background jobs (&)
- listing jobs
- switching
- suspending
- interrupting

Listing processes

```
# ps aux

USER          PID %CPU %MEM   VSZ  RSS TTY        STAT START   TIME COMMAND
 root           1  0.0  0.0  1492  104 ?           S    Aug28   0:04 init [2]
 root           2  0.0  0.0     0    0 ?           SW   Aug28   0:00 [keventd]
 root           3  0.0  0.0     0    0 ?           SWN  Aug28   0:00 [ksoftirqd_CPU0]
 root           4  0.0  0.0     0    0 ?           SW   Aug28   6:05 [kswapd]
 root           5  0.0  0.0     0    0 ?           SW   Aug28   0:24 [bdflush]
 root           6  0.0  0.0     0    0 ?           SW   Aug28   0:10 [kupdated]
 root         222  0.0  0.0     0    0 ?           SW   Aug28   0:10 [kjournald]
 root         223  0.0  0.0     0    0 ?           SW   Aug28   0:00 [kjournald]
 root         294  0.0  0.0     0    0 ?           SW   Aug28   0:00 [khubd]
 daemon      1508  0.0  0.0  1604   64 ?           S    Aug28   0:00 /sbin/portmap
 root        1564  0.0  0.2  1628  360 ?           S    Aug28   0:56 /sbin/syslogd
 root        1603  0.0  0.0  2152   80 ?           S    Aug28   0:02 /sbin/klogd
 root        1607  0.0  1.8 12644 2324 ?           S    Aug28   0:00 /usr/sbin/named
```

Process listing variations

- `ps -elf`
- `ps`
- `ps u`
- `ps ux`
- `ps x -o user,pid,ppid,cmd`

Process states

- Runnable (R)
- Sleeping (S)
- Uninterruptible Sleep (D)
- Traced or stopped (T)
- Defunct or zombie (Z)
- Additional BSD status codes
  - No resident pages (W)
  - High priority process (<)
  - Low priority process (N)
  - Process with pages locked in mem (L)

Monitoring processes

- top cpu processes
- cpu state information
- similar details to ps
- process priority and nice

Signals

- What are they?
- What do they do?
- Sending signals via the keyboard
- Sending signals with the kill command
- Sending signals with system calls
- Common signals

Exercise 5 – Processes

1) Review the man page for ps and do the following,
   - display only your processes
   - display all processes
2) Identify some high priority tasks running on the system.
3) Identify some processor and memory intensive tasks.
4) Start a simple process of your own and experiment with running it in the background and bringing it to the foreground (see notes for example process to run).
5) Kill the process while it is running in the background.

Working With Files

OS File differences

- EOL
- EOF
- Managing
  - ascii transfer mode
  - binary transfer mode
  - `todos` / `fromdos`
  - `dos2unix` / `unix2dos`
  - `tr` command

`find` – finding files on filesystems

- `find` command
  - searching on filenames
  - searching on file properties
    - type
    - age
    - size
    - permissions
  - doing something with what you find
    - exec
- `locate` command

grep - finding strings in files

- basics
- counting
- inverted matches
- searching many files

Regular Expressions

- String matching patterns
- Easier to write than to read!

- Regular expressions consist of 2 parts

  (single character matches) + (repetition characters)

- Widely supported
  - Shell
  - Perl
  - Java
  - C/C++

Regular Expression Examples

1. `a+`
2. `.*`
3. `[foo]+`
4. `Mo?zilla`
5. `[a-zA-Z]{2,5}`
6. `[Mozilla]{4,}`

Exercise 6 – Files and Regex

1) **Create a text file on your Windows system and transfer it to the Linux system using ftp in ascii and binary transfer modes. Repeat using sftp.**
2) **Count the occurrences of the word "kernel" in all files starting with the letter "m" (hint: find and grep).**
3) **Write a single regular expression to match the following sequences in a file:**
   - **aaa**
   - **bbbb**
   - **bbb**
   - **aaaa**

Other Useful Commands

Date and time

- `date`
  - format specifiers
- `cal`
- `time` *<command>*
  - real time
  - user time
  - system time
- `sleep` *<time to pause for>*

More on viewing files

- Looking at the start or end of files
  - `head`
  - `tail`
- Monitoring files as they grow
  - `tail -f`
  - `less + SHIFT-f`
- File statistics
  - `wc`
- Sorting file contents
  - `sort`

Packing files

- `tar`
  - `tar -cvf`
  - `tar -xvf`
  - `tar -tvf`
- `cpio`
  - `cpio -v`
  - `cpio -id`

Compressing files

- `gzip`
  - standard
  - pretty good compression in short time
- `bzip2`
  - pretty standard
  - very good compression in medium time
- `compress`
  - traditional unix tool
- `zip` (or `jar`)
  - traditional dos/windows tool
- Combining `tar` with `gzip` or `bzip2`

Scheduling

- Running jobs once at some future time
  - `at`
- Running jobs once when the system is under-used
  - `batch`
- Running jobs regularly
  - `cron`

```
0 7 * * * ~/bin/daily-backup.sh
0 7 * * 1 ~/bin/weekly-backup.sh
0 7 1 * * ~/bin/monthly-backup.sh
```

Exercise 7 – Commands

1) **Time the ls command and explain the various output fields.**
2) **Create a tar-file of your home directory.**
3) **Time both gzip and bzip2 compressing this tar-file and comment on the differences found.**
4) **Schedule a one-off job removing the compressed files in 5 minutes time.**
5) **Explain the following cron entries**
   - `25 4 * * 1 w`
   - `* * * * * /bin/monitor.sh`
   - `0 * * * * /bin/wall /0.txt`
   - `5 9-17 * * 1-5 /bin/work`

Editing files

Introduction

- Typical editors on Linux systems
  - vi / vim
  - emacs / xemacs
- Starting vi
- Buffers
- vi modes
  - command mode
  - insert mode
  - switching modes
- Quitting vi

Navigation in vi

- Traditional navigation
- Cursor keys
- Paging
- Advanced navigation
  - start of line / end of line
  - next word
  - previous word
  - start of file / end of file
  - specifying particular lines

Cut and paste in vi

- Copy (yank)
  - single lines
  - multiple lines
- Paste
- Cut (delete)
  - lines
  - characters
- Inserting files

Search and replace in vi

- Simple search
  - repeating
  - backwards
- Simple find and replace
- Advanced search
- Regular expressions

Advanced vi

- Undo
- Starting editing on a particular line
- Replace mode

Exercise 8 - Editing

1) Create a text file containing a few sentences of text.
2) Copy a few lines of the file.
3) Use search and replace to replace all occurrences of the word “the” in your file with the string “xxxx”.
4) Copy /etc/passwd to a file in /var/tmp and practice navigating around it.
5) Replace all non-letter characters in a copy of /etc/passwd with the letter 'X.

[Additional *Commands* exercise]

- Schedule a regular job using cron to back up your home directory to a compressed archive in /var/tmp. This job should run at 0500 every day except Sunday. Ensure that no-one else can view or extract the file.

Scripting

Introduction

- When to use shell scripts
  - system maintenance
  - batch operations
  - easily automated repetitive tasks
- When not to use them
  - webserver scripts (CGI)
  - high security jobs
  - heavily loaded systems
- Alternatives
  - Perl
  - Python

Your first shell script

- #!/bin/sh
- Comments
- External commands
- Shell builtins
  - alias, bg, cd, echo, ...
- Shell constructs
  - if, for, while, ...
- Execute permissions
- &&
- ||

hello worlds

```
#!/bin/sh
# this is a hello world script
echo “hello world”
```

```
#!/bin/sh
# this is another hello world
# script
/bin/echo “hello world”
```

Running a script

- #!
- Running scripts through the shell command
- Debugging scripts
  - -x
  - -v

Shell variables

- Setting
- Using
- Rules for naming
- All variables of type string

Shell variables & quoting

- Single quote - '
- Double quote - "
- Back quote - `

Special Variables

- $0
- $1 - $n
- $#
- $*
- $@
- Backslash - \

Loops

-
  ```
  for VARIABLE in LIST
   do
       BODY
   done
  ```

-
  ```
  while EXPRESSION
  do
    BODY
  done
  ```

-
  ```
  until EXPRESSION
  do
    BODY
  done
  ```

The if statement

```
if CONDITION
then
  BODY
elif CONDITION
then
  BODY
else
  BODY
fi
```

case and test

- case
- test (or [..])
  - STRING1 = STRING2
  - STRING1 != STRING2
  - INTEGER1 -eq INTEGER2
  - INTEGER1 -ne INTEGER2
  - -f FILE
  - ...

Exit codes,functions

- Exit status
  - checking
  - exit
  - $?
- ${VARIABLE}
- Functions
  - arguments
  - returning status

Special devices

- Useful special devices
  - /dev/null
  - /dev/zero
  - /dev/random and /dev/urandom

sed & awk

- Both perform text transformations
- Operate on standard input (from a pipeline) or files
- Can also be used to build scripts in their own right
- Come in different flavours
- Support regular expressions

```
sed 's/regexp/replacement text/{flags}'
    e.g. sed 's/o/_/g' foo.txt

awk -F <chars>  ' { print $n  } ' filename
    e.g. awk -F, '{print $3, $2, $1}' csv.txt
```

Shell configuration files

- Bourne shells
  - /etc/profile
  - ~/.profile
  - ~/.bash_profile
  - ~/.bash_login
- C shells
  - /etc/csh.login
  - /etc/csh.cshrc
  - ~/.tcshrc
  - ~/.cshrc
- .bashrc (interactive and non-interactive shells)
- . and source

Exercise 9.1 – Scripting

1. Write a script which takes and processes the following options. The script should display an error message when the arguments to the script are incorrect,
   - -h displays a help message
   - -l performs an ls (with each line preceded by the command name)
   - -d displays the date in the form “22:34 25-Dec-2004”
2. Write a script which takes a filename as an argument and renames that file to an all lowercase version of the original filename.
3. Write a script that displays the following message

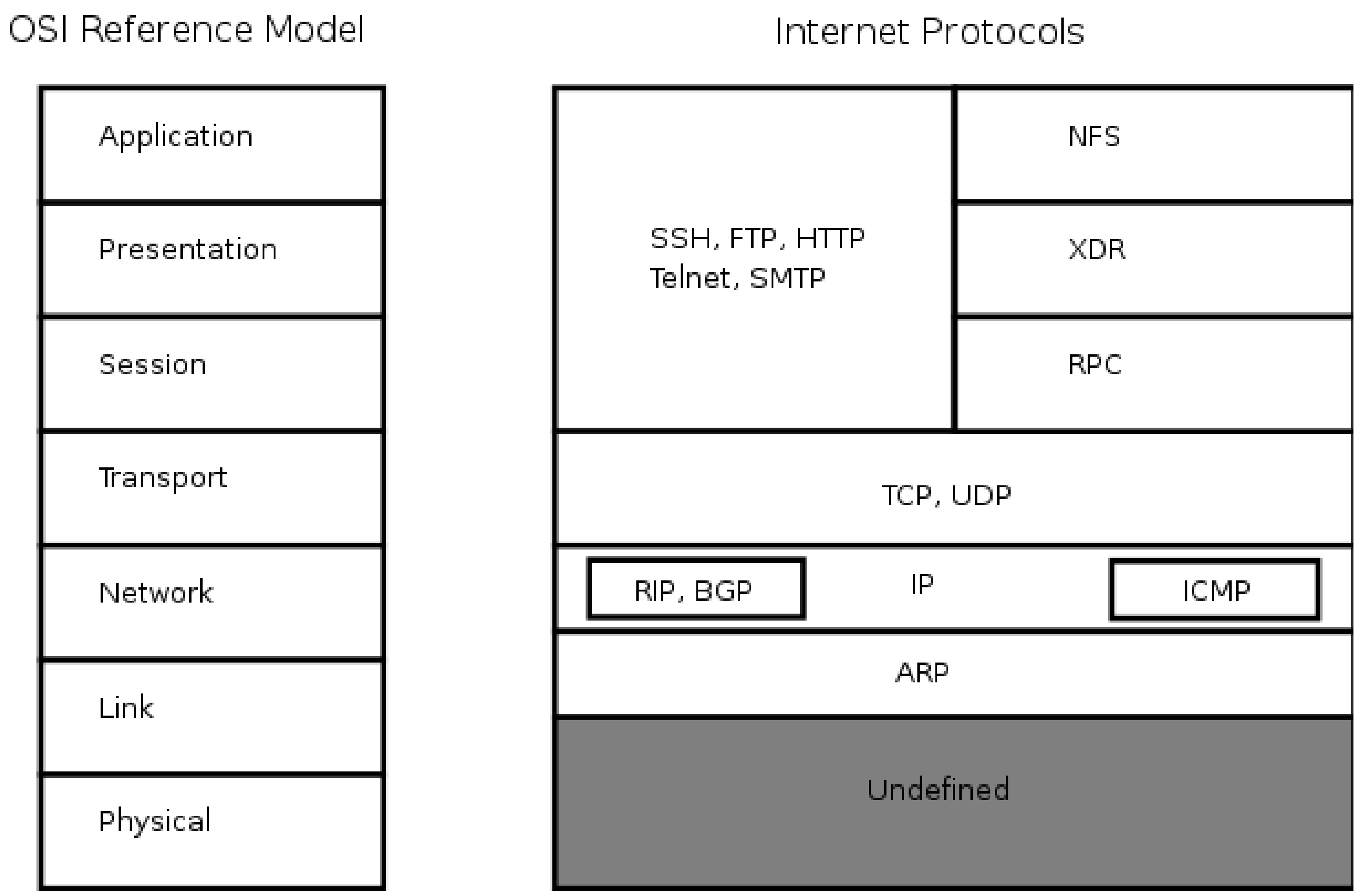
   Hello *username*, today is *Day of week*

Exercise 9.2 – Scripting

4. **Write a script to back up any user's home directory ($HOME) using tar and gzip. Print a message and stop if it fails at any point ($?). Set permissions on the backup file so only the user can access the file. Name the file backup-*USERNAME-YYYYMMdd*.tar.gz**
5. **Write a script that takes two files as arguments (-m message.txt and -r recipients.txt). The script should email the contents of message.txt to each email address in recipients.txt and should also tell each user what time it is. Start each message with 'Dear *first name*'.**
6. **Write a script to add numbers e.g. add.sh 2 2 will display 4 on output (hint: expr and read).**
7. **Write a script to convert miles to km.**

Networking

Networking Concepts



IP Addresses

- Address (network part and host part)
- Netmask (used to identify the network part)
- Network
- Broadcast (address all hosts on a network)

Address	**192.168.0.1**	11000000.10101000.00000000. 00000001
Netmask	**255.255.255.0 = 24**	11111111.11111111.11111111. 00000000
Host part	**0.0.0.255**	00000000.00000000.00000000. 11111111
Network	**192.168.0.0/24**	11000000.10101000.00000000. 00000000
First Host	**192.168.0.1**	11000000.10101000.00000000. 00000001
Last Host	**192.168.0.254**	11000000.10101000.00000000. 11111110
Broadcast	**192.168.0.255**	11000000.10101000.00000000. 11111111

Devices and Tools

- Network devices
  - eth0, eth1, ...
- Tools
  - ifconfig
  - ping
  - telnet
  - traceroute
  - route
  - ipcalc

Domain Name System - DNS

- Overview
- nsswitch.conf
- /etc/hosts
- /etc/resolv.conf
- host
- dig

  dig <hostname>

  dig -x <IP address>
- nslookup

  nslookup <hostname>

  nslookup <IP address>

Exercise 10 – Networking

1. **Find the range of addresses for the 10.10.x.x network (using a netmask of 255.255.0.0).**
2. **Suggest a netmask and network for splitting a 10.10.10.x address into 4 separate networks.**
3. **Describe the fields in the network configuration of the first ethernet device on the Linux server.**
4. **Verify that the other server is running.**
5. **Identify the network path to that server.**
6. **Verify that there is a DNS entry for both servers (what is it?).**
7. **Verify if the mail service is running on the server (port 25).**

The System

The super-user account

- The `root` account
- Typical super-user activities
  - Filesystem maintenance
  - Software installation
  - Reviewing log-files
- `su`
- `sudo`
  - `/etc/sudoers`

System log files

- `syslogd`
- `/var/log`
  - `messages`
  - `syslog`
  - `apache/`
  - `...`
- Log rotation
- Reviewing
- `dmesg`

Services

- Starting/stopping
- Adding new ones
- `/etc/init.d`
- Run-levels
  - `/etc/inittab`
  - `/etc/rc`*`n`*`.d`

Software packages

- Overview
  - Red Hat (rpm)
  - Novell / SuSE (rpm)
  - Debian (deb)
  - Slackware (tgz)
- Dependencies
- Versions
- Package contents
  - Packaged software
  - Installation software
  - Package information
  - Dependency information

RPM

- Installing
  rpm -ivh <*package*>.rpm
- Listing
  rpm -qa
- Removing
  rpm -e <*package*>
- Advanced uses
  rpm -qi <package>
  rpm -qR
- Alternatives
  - yum
  - red carpet

Exercise 11 – The System

1. **Write a script that sends an email to some address containing a list of todays log messages from the ssh daemon.**
2. **Write a script to remove all log-files that meet the following criteria:**
   - **older than 10 days**
   - **larger than 1 Megabyte**
3. **Check the status of the webserver service.**
4. **Check the current runlevel. List the services started and stopped for that runlevel.**

Developing on Linux

C on Linux

- Compilers
  - GCC
    - C, C++
    - Objective-C
    - Fortran
    - Ada
  - Intel
    - C, C++
    - Fortran (F77, F95)
  - The Portland Group
    - C, C++
    - Fortran (F77, F95)

Java on Linux

- JRE vs. JDK
- Installing
- What version of Java?
- GCJ
- Environment Variables
- IDEs
  - Eclipse
  - Netbeans

Other scripting languages

- Perl
  - powerful text manipulation
  - commonly used for system administration tasks
  - syntax similar to shell and C
- Python
  - newer scripting language
  - more emphasis on OO
  - emphasises readability of code
  - uses indentation rather than curly braces to delimit blocks
- Ruby
  - very strong OO emphasis
  - intended to be easy to learn
  - *Ruby on Rails* - web application framework

Exercise 12 – Developing on Linux

1. **Check what version of gcc is installed.**
2. **Write a simple hello world program in C and compile it up with gcc.**
3. **Check what version (if any) of Java is installed.**

Advanced SSH topics

Keys

- Uses
  - Eliminates passwords
  - Automation (remote commands)
  - Restrict certain commands to certain users and hosts
- HOWTO

  1) Generate your own set of keys

     (local host) `ssh-keygen -t rsa`

  1) Copy the public key to the system and account you want to access

     (local host) `scp ~/.ssh/id_rsa.pub user@hostname:.`

     (remote host) `cat id_rsa.pub >> ~/.ssh/authorized_keys`

  1) Verify you can login to that account without a password

     (local host) `ssh user@hostname`

Tunnelling

```
ssh -L portA:hostA:portB username@hostB
```

(open a tunnel between portB on hostB and portA on hostA)

- Useful for securely using insecure services through firewalls.
- Can tunnel any protocol.
- Can chain multiple tunnels.

In closing ...

- Summary
- Next steps
- Questionaire

Thank you and well done!