



Curtin College

in association with



Curtin University

# User and Linux Admin

Computer Systems 2000 (CS2000)

Trimester 2 2020



# File & Directory Permissions - Contents

- File and directory permissions in more detail and how these can be changed.
- Ways to examine the contents of files.
- How to find files
- Ways of searching files for text patterns.
- How to sort files.
- Tools for compressing files and making backups.
- Accessing removable media

# File & Directory Permissions

| <u>Permission</u> | <u>File</u>                                 | <u>Directory</u>  |
|-------------------|---|---|
| <u>read</u>       | User can look at the contents of the file   | User can list the files in the directory  |
| <u>write</u>      | User can modify the contents of the file    | User can create new files and remove existing files in the directory  |
| <u>execute</u>    | User can use the filename as a UNIX command | User can change into the directory, but cannot list the files unless (s)he has read permission. User can read files if (s)he has read permission on them. |

three categories of users are user/owner (u), group (g) and others (o)

# File & Directory Permission

- File and directory permissions can only be modified by their owners, or by the superuser (root), by using the chmod (change [file or directory] mode) system utility.

**\$ chmod *options files***

- chmod accepts options in two forms.
  - *Specified as a sequence of 3 octal digits*
  - *Each octal digit represents the access permissions for the user/owner, group and others respectively*

<https://chmodcommand.com/>



# File & Directory Permissions Example

**\$ chmod 600 private.txt**

- *sets the permissions on private.txt to rw- - - - -*
- *(i.e. only the owner can read and write to the file).*

# File & Directory Permissions

- Permissions may be specified symbolically

- *u (user), g (group), o (other), r (read), w (write), x (execute), + (add permission), - (take away permission) and = (assign permission)*

**\$ chmod ug=rw,o-rw \*.txt**

- *sets the permissions on all files ending in \*.txt to rw-rw----*

- -R option

- *recursively modify file permissions.*

**\$ chmod -R go+r play**

- *will grant group and other read rights to the directory play and all of the files and directories within play*



# File & Directory Permissions

- chgrp (change group)
  - **\$ chgrp *group files***
- Can be used to change the group that a file or directory belongs to. It also supports a -R option.
- Example
  - **\$ chgrp -R *staff /office/files***



# Examining Files

- The cat command is a standard Unix program used to concatenate and display files.
- file analyses a file's contents and reports a high-level description of what type of file it appears to be:

**\$ file myprog.c letter.txt webpage.html**

**myprog.c: C program text**

**letter.txt: English text**

**webpage.html: HTML document text**

- *file* can identify a wide range of files but sometimes gets understandably confused (e.g. when trying to automatically detect the difference between C++ and Java code).
- Remember \*nix does not use file extensions





# Examining Files

- `head, tail filename {-lines}`
- head and tail display the first and last few lines in a file respectively.

**`$ tail -20 messages.txt`**

**`$ head -5 messages.txt`**

- tail -f option
  - *continuously monitor the last few lines of a (possibly changing) file.*
  - *This can be used to monitor log files, for example:*

**`$ tail -f /var/log/messages`**

# Examining Files

- `objdump options binaryfile`
  - *objdump can be used to disassemble binary files*
- `od options filename` (octal dump)
  - *od can be used to displays the contents of a binary or text file in a variety of formats, e.g.*
- Example

```
$ cat hello.txt
hello world
$ od -c hello.txt
0000000 h e l l o w o r l d \n
0000014
$ od -x hello.txt
0000000 6865 6c6c 6f20 776f 726c 640a
0000014
```



# Finding Files

- **\$ find *directory* -name *targetfile* -print**

- Search for a file called *targetfile* in any part of the directory tree rooted at *directory*.
- can include wildcard characters.

- Example:

- **\$ find /home -name "\*.txt" -print 2>/dev/null**

- search all user directories for any file ending in ".txt" and output any matching files (with a full absolute or relative path).
- Quotes (") are necessary to avoid filename expansion
- 2>/dev/null suppresses error messages
  - from errors such as not being able to read the contents of directories for which the user does not have the right permissions.

# Finding Files

- ▶ find files by type
  - ▶ *type f* for files
  - ▶ *type d* for directories
- ▶ by permissions/size
  - ▶ *perm o=r* for all files and directories that can be read by others
  - ▶ *size*
- ▶ Execute commands on the returned files

**\$ find . -name "\*.txt" -exec wc -l '{}' ';'**

  - ▶ *counts the number of lines in every text file in and below the current directory. The '{}' is replaced by the name of each file found and the ';' ends the -exec clause.*
- ▶ For more information about find and its abilities, use `man find` and/or `info find`.



# Finding Files

- *which command*

- *find out where command is stored on disk*

**\$ which ls**

**/bin/ls**

- *locate string*

- *find can be slow*

- *The locate command provides a much faster way of locating all files whose names match a particular search string.*



# Finding Files

## **\$ locate ".txt"**

- find all filenames in the filesystem that contain ".txt" anywhere in their full paths.
- One disadvantage of locate is it stores all filenames on the system in an index that is usually updated only once a day.
  - *locate will not find files that have been created very recently.*
  - *may also report filenames as being present even though the file has just been deleted.*
  - *Unlike find, locate cannot track down files on the basis of their permissions, size and so on.*



# Finding Text

- `grep` (General Regular Expression Print)

**`$ grep options pattern files`**

- *searches the named files (or standard input if no files are named) for lines that match a given pattern.*

**`$ grep hello *.txt`**

- *searches all text files in the current directory for lines containing "hello"*

- Options

- *-c (print a count of the number of lines that match)*
- *-i (ignore case)*
- *-v (print out the lines that don't match the pattern)*
- *-n (printout the line number before printing the matching line).*

**`$ grep -vi hello *.txt`**

- `grep` may be combined with `find` using backward single quotes to pass the output from `find` into `grep`.

**`$ grep hello `find . -name "*.txt" -print``**

- *search all text files in the directory tree rooted at the current directory for lines containing the word "hello".*



# Sort

- `sort filenames`

- *sorts lines contained in a group of files alphabetically (or if the `-n` option is specified) numerically.*

- `$ sort input1.txt input2.txt > output.txt`**

- *outputs the sorted concatenation of files `input1.txt` and `input2.txt` to the file `output.txt`.*

- `uniq filename`

- `uniq` removes duplicate adjacent lines from a file.

- *This facility is most useful when combined with `sort`:*

- `$ sort input.txt | uniq > output.txt`**

# File Compression and Backup

- tar (tape archiver)
  - *Creates a single file known as an archive.*
  - *contains other files plus information about them, such as their filename, owner, timestamps, and access permissions.*
  - *tar does not perform any compression by default.*
- To create a disk file tar archive, use  
**\$ tar -cvf archivenam filenames**
  - *c = create*
  - *v = verbose (output filenames as they are archived)*
  - *f = file*
- To list the contents of a tar archive, use  
**\$ tar -tvf archivename**
- To restore files from a tar archive, use  
**\$ tar -xvf archivename**



Successfully extracted a Toyota

# File Compression and Backup

- `cpio`

- *doesn't automatically archive the contents of directories*

**`$ find . -print -depth | cpio -ov -Htar > archivename`**

- This will take all the files in the current directory and the directories below and place them in an archive called *archivename*.
  - *-depth option controls the order in which the filenames are produced and is recommended to prevent problems with directory permissions when doing a restore.*
  - *-o option creates the archive*
  - *-v option prints the names of the files archived as they are added*
  - *-H option specifies an archive format type*



# File Compression and Backup

- To list the contents of a cpio archive

**\$ cpio -tv < *archivename***

- To restore files, use:

**\$ cpio -idv < *archivename***

- *-d option will create directories as necessary*



# Compress and GZip

**\$ compress filename**

**\$ gzip filename**

▀ *In each case, filename will be deleted and replaced by a compressed file called filename.Z or filename.gz.*

▀ To reverse the compression process, use:

**\$ compress -d filename**

**\$ gzip -d filename**



# Removable Media

- mount, umount
- attach the filesystem found on some device to the filesystem tree.
- umount command will detach it again
  - *very important to remember to do this when removing the floppy or CDROM*
- The file /etc/fstab contains a list of devices and the points at which they will be attached to the main filesystem:

**\$ cat /etc/fstab**

```
/dev/fd0 /mnt/floppy auto rw,user,noauto 0 0
```

```
/dev/hdc /mnt/cdrom iso9660 ro,user,noauto 0 0
```



# Removable Media

- mount point for the floppy drive is /mnt/floppy
- mount point for the CDROM is /mnt/cdrom

**\$ mount /mnt/floppy**

**\$ cd /mnt/floppy**

**\$ ls (etc...)**

**\$ umount /mnt/floppy**



# USB Flash (FAT 32)

```
$ mkdir -p /mnt/myusb
```

```
$ mount -t vfat -o rw,users /dev/sda1/mnt/myusb
```

➤ To confirm mount point

```
$ mount
```

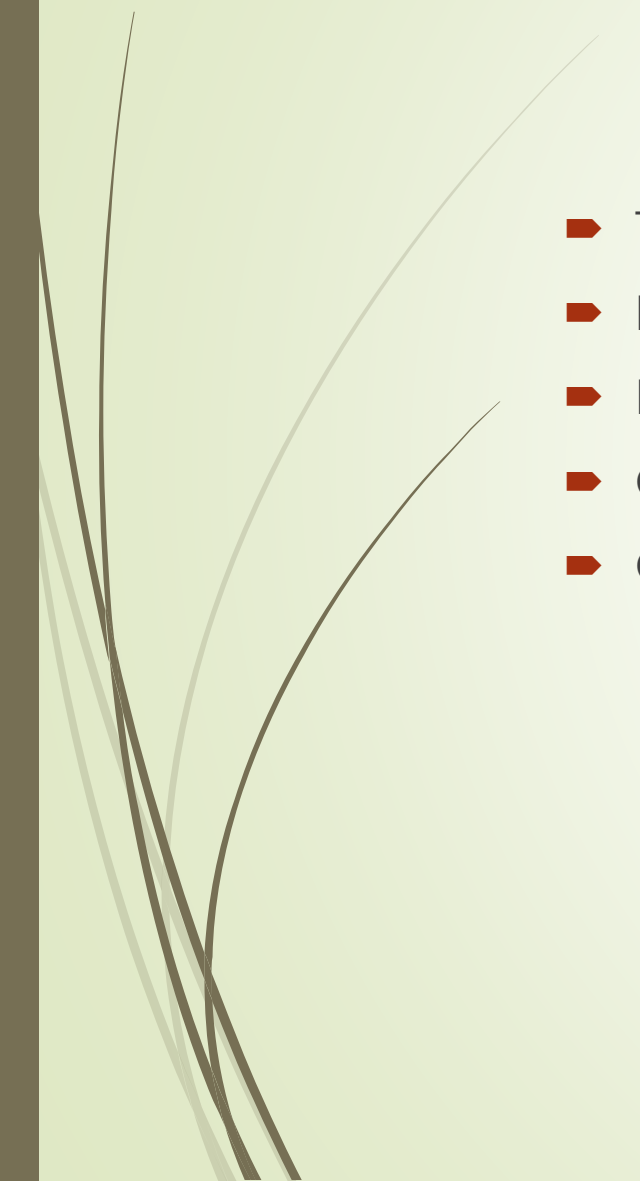
➤ You should see

```
/dev/sda1 on /mnt/myusb type vfat
```

```
(rw,noexec,nosuid,nodev)
```



# Processes and Redirection - Contents

- The concept of a process.
  - Passing output from one process as input to another using pipes.
  - Redirecting process input and output.
  - Controlling processes associated with the current shell.
  - Controlling other processes.
- 



# Processes



- A process is a program in execution.
- Every time you invoke a system utility or an application program from a shell, one or more "child" processes are created by the shell in response to your command.
- All UNIX processes are identified by a unique process identifier or PID.
- `init` is the first process created at system start
  - *has a PID of 1*
  - *All other processes are said to be "descendants" of `init`.*



# Pipes

- | operator is used to pass data directly to one process to another.

**\$ cat hello.txt | sort | uniq**

- *creates three processes (corresponding to cat, sort and uniq)*
- *the output of the cat process is passed on to the sort process which is in turn passed on to the uniq process.*

**\$ cat hello.txt | grep "dog" | grep -v "cat"**

- *finds all lines in hello.txt that contain the string "dog" but do not contain the string "cat".*





# Redirection

- Processes usually write to standard output (the screen) and take their input from standard input (the keyboard).
  - another output channel called standard error, where processes write their error messages; by default error messages are also sent to the screen.*
- To redirect standard output to a file instead of the screen, we use the > operator:

```
$ echo hello
```

```
hello
```

```
$ echo hello > output
```

```
$ cat output
```

```
hello
```

# Redirection

- The contents of the file output will be destroyed if the file already exists.
- To append the output of the echo command to the file, use the >> operator:

```
$ echo bye >> output
```

```
$ cat output
```

```
hello
```

```
bye
```

- To capture standard error, prefix the > operator with a 2 (in UNIX the file numbers 0, 1 and 2 are assigned to standard input, standard output and standard error respectively)

```
$ cat nonexistent 2>errors
```

```
$ cat errors
```

```
cat: nonexistent: No such file or directory
```

# Redirection

- To redirect standard error and standard output to two different files:

```
$ find . -print 1>files 2>errors
```

- or to the same file:

```
$ find . -print 1>output 2>output
```

```
$ find . -print >& output
```

- Standard input is redirected using the < operator

- *input is read from a file instead of the keyboard:*

```
$ cat < output
```

```
hello
```

```
bye
```

# Redirection

- To combine input redirection with output redirection

**`$ cat < output > output`**

- be careful not to use the same filename in both places as this will destroy the contents of the file output.
  - *The first thing the shell does when it sees the > operator is to create an empty file ready for the output.*
- To pass standard output to system utilities that require filenames as "-":

**`$ cat package.tar.gz | gzip -d | tar tvf -`**
- Here the output of the gzip -d command is used as the input file to the tar command.



# Controlling Processes



- Most shells provide sophisticated job control facilities that let you control many running jobs (i.e. processes) at the same time
- Jobs can either be in the foreground or the background.
  - *There can be only one job in the foreground at any time.*
  - *The foreground job has control of the shell*
  - *Jobs in the background do not receive input from the terminal*
- The foreground job may be suspended
  - *Ctrl-Z to suspend fg or bg to continue*
- To initiate a background process append '&' to the command

# Controlling Processes

```
$ find . -print 1>output 2>errors &  
[1] 27501
```

```
$
```

- [1] is returned by the shell to represent the job number of the background process
- 27501 is the PID.

- *To see a list of all the jobs associated with the current shell, type jobs:*

```
$ jobs
```

```
[1]+  Running find . -print 1>output 2>errors &
```

```
$
```

- Note that if you have more than one job you can refer to the job as %n where n is the job number.



# Controlling Processes

**\$ ps**

| <b>PID TTY</b>      | <b>TIME CMD</b>      |
|---------------------|----------------------|
| <b>17717 pts/10</b> | <b>00:00:00 bash</b> |
| <b>27501 pts/10</b> | <b>00:00:01 find</b> |
| <b>27502 pts/10</b> | <b>00:00:00 ps</b>   |

- The PID of the shell (bash) is 17717, the PID of find is 27501 and the PID of ps is 27502.

# Controlling Processes

- The kill command terminates a process or job abruptly
  - *kill allows jobs to be referred to in two ways - by their PID or by their job number*  
**\$ kill %1**  
**\$ kill 27501**
- kill sends the process a signal requesting it shutdown and exit gracefully (the SIGTERM signal)
- -9 option (the SIGKILL signal) kills a process abruptly  
**\$ kill -9 27501**
- kill handles many other types of signals
  - *To see a list of such signals, run kill -l.*

# Controlling Processes

- To use ps to show all running processes  
**\$ ps -fae (or ps -aux on linux machines)**
- **ps -aeH** displays a full process hierarchy (including the init process).
- top provides an interactive way to monitor system activity
- Useful keys in top are:
  - s - set update frequency
  - u - display processes of one user
  - k - kill process (by PID)
  - q - quit
- Note that, for obvious security reasons, you can only kill processes that belong to you (unless you are the superuser).



# Other Useful Commands

- `rlogin`, `rsh`
  - *`rlogin` and `rshare` are insecure facilities for logging into remote machines and for executing commands on remote machines respectively.*
  - *Along with `telnet`, they have been superseded by `ssh`*
- `ssh machinename` (secure shell)
  - *`ssh` is a secure alternative for remote login and also for executing commands on a remote machine*
  - *Provides secure encrypted communications between two untrusted hosts over an insecure network*
  - *X11 connections can also be forwarded over the secure channel*
  - *`ssh` is not a standard system utility (though often included in the “distribution”)*
  - *obtained from <http://www.ssh.org>*

# Other Useful Commands

## ➤ **scp sourcefiles destination (secure copy)**

- *scp* is a secure way of transferring files between computers
- *Same syntax as the UNIX cp command except that the arguments can specify a user and machine as well as files*

**\$ scp bob@deco.ece.curtincollege.edu.au:~/hello.txt .**

- *copy (subject to correct authentication) the file hello.txt from the user account on the remote machine deco.ece.curtin.edu.au into the current directory (.) on the local machine*
- ## ➤ **lynx**
- *provides a way to browse the web on a text-only terminal (why would this be useful???)*

**Apply now to begin your studies in 2014**

## Information for

- \* [Future students](#)
- \* [International future students](#)
- \* [Current students](#)
- \* [Staff](#)
- \* [Alumni](#)
- \* [Researchers](#)
- \* [Visitors](#)
- \* [Media](#)

## Popular

- \* [OASIS Login](#)
- \* [Library](#)
- \* [Teaching & learning](#)
- \* [Scholarships](#)
- \* [Giving to Curtin](#)
- \* [Getting here](#)
- \* [Bookshop](#)
- \* [Academic calendar](#)
- \* [Fees & charges](#)
- \* [Online payments](#)
- \* [Jobs at Curtin](#)

## What's on

[Did you attend Open Day?](#) [Did you attend Open Day?](#)

Complete our online survey about Open Day for your chance to win one of two GoPro Hero 3s!

[Learn more](#)

[Psychology and Speech Pathology info evening](#) [Psychology and Speech Pathology info evening](#)

If you are interested in language, development and psychology come along on 27 August to find out more about the courses and career op

[Learn more](#)

— [press space for next page](#) —

Arrow keys: Up and Down to move. Right to follow a link; Left to go back.  
H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list



# Printer Utils

- CUPS Common Unix Printing System
- **lpr -Pprintqueue filename**
  - *lpr* adds a document to a print queue, so that the document is printed when the printer is available.
  - *Look at /etc/printcap to find out what printers are available.*
- **lpq -Pprintqueue**
- **lpq** checks the status of the specified print queue.
  - *Each job will have an associated job number.*
- **lprm -Pprintqueue jobnumber**
  - *lprm* removes the given job from the specified print queue.
- Note that *lpr*, *lpq* and *lprm* are BSD-style print management utilities.
- SYSV UNIX use the equivalents *lp*, *lpstat* and *cancel*.





# Superuser/Root

- The superuser is a privileged user who has unrestricted access to all commands and files on a system regardless of their permissions.
- log in as root and the root password
  - *Standard practice is to allow root login at the console only, not via network connections*
  - *Normal safeguards that apply to other user accounts do not apply to root.*
  - *Using root for mundane tasks may result in misplaced keystrokes having catastrophic effects e.g. accidentally typing "rm -rf \* .txt" instead of "rm -rf \*.txt"*
- su (switch user)



# Superuser/Root

- If the user account is not specified, root is assumed
  - *su does not change your current directory*
  - *"-" option will run the target user's startup scripts and change into their home directory*
- Note that the root account often displays a different prompt (usually a #).

# User Accounts

- `useradd` (in `/usr/sbin`):
  - Adds new user information to the `/etc/passwd` file and creates a new home directory
  - The password should also be set at creation (using the `-p` option on `useradd`, or using the `passwd` utility)
- `groupadd` (in `/usr/sbin`):
  - creates a new user group and adds the new information to `/etc/group`

**# `groupadd groupname`**
- `usermod` (in `/usr/sbin`):
  - To modify the group permissions of an existing user, use

**# `usermod -g initialgroup username -G othergroups`**
- `groups`
  - You can find out which groups a user belongs to by typing:
  - **# `groups username`**

# Scheduling Jobs

- crond is a daemon that executes commands that need to be run regularly according to some schedule.
- The schedule and corresponding commands are stored in the file `/etc/crontab`.
- Each entry in the `/etc/crontab` file entry contains six fields separated by spaces or tabs in the following form:  
minute hour day\_of\_month month weekday command
- These fields accept the following values:

|                     |                                |
|---------------------|--------------------------------|
| <b>minute</b>       | <b>0 through 59</b>            |
| <b>hour</b>         | <b>0 through 23</b>            |
| <b>day_of_month</b> | <b>1 through 31</b>            |
| <b>month</b>        | <b>1 through 12</b>            |
| <b>weekday</b>      | <b>0 (Sun) through 6 (Sat)</b> |
| <b>command</b>      | <b>a shell command</b>         |
- You must specify a value for each field. Except for the command field

# Scheduling Jobs

- A number in the specified range, e.g. to run a command in May, specify 5 in the month field.
- Two numbers separated by a dash to indicate an inclusive range
- A list of numbers separated by commas, e.g. to run a command on the first and last day of January, you would specify 1,31 in the day\_of\_month field.
- \* (asterisk), meaning all allowed values, e.g. to run a job every hour



The diagram illustrates the five fields of a cron job schedule. Five asterisks (\*) are aligned vertically. Blue arrows point from each asterisk to its corresponding field name and range: the top asterisk points to 'minute (0-59)', the second to 'hour (0-23)', the third to 'day of month (1-31)', the fourth to 'month (1-12)', and the bottom asterisk points to 'day of week (0-6) (Sunday is 0)'. Below the asterisks is the text '<command to execute>'.

```
* * * * * <command to execute>
```

# Scheduling Jobs

- You can also specify some execution environment options at the top of the `/etc/crontab` file:

**SHELL=/bin/bash**

**PATH=/sbin:/bin:/usr/sbin:/usr/bin**

**MAILTO=root**

- To run the `calendar` command at 6:30am. every Mon, Wed, and Fri, a suitable `/etc/crontab` entry would be:

**30 6 \* \* 1,3,5 /usr/bin/calendar**

- *The output of the command will be mailed to the user specified in the `MAILTO` environment option.*
- *No need to restart `crond` after changing `/etc/crontab` - it automatically detects changes.*



# Keeping Processes Alive

- It is important that daemons related to mission critical services are immediately re-spawned if they fail for some reason.

- *add your own entries to the /etc/inittab file.*

**rs:2345:respawn:/home/sms/server/RingToneServer**

- rs is a 2 character code identifying the service
- 2345 are the run levels for which the process should be created.
- The init process will create the RingToneServer process at system startup, and re-spawn it should it die for any reason.



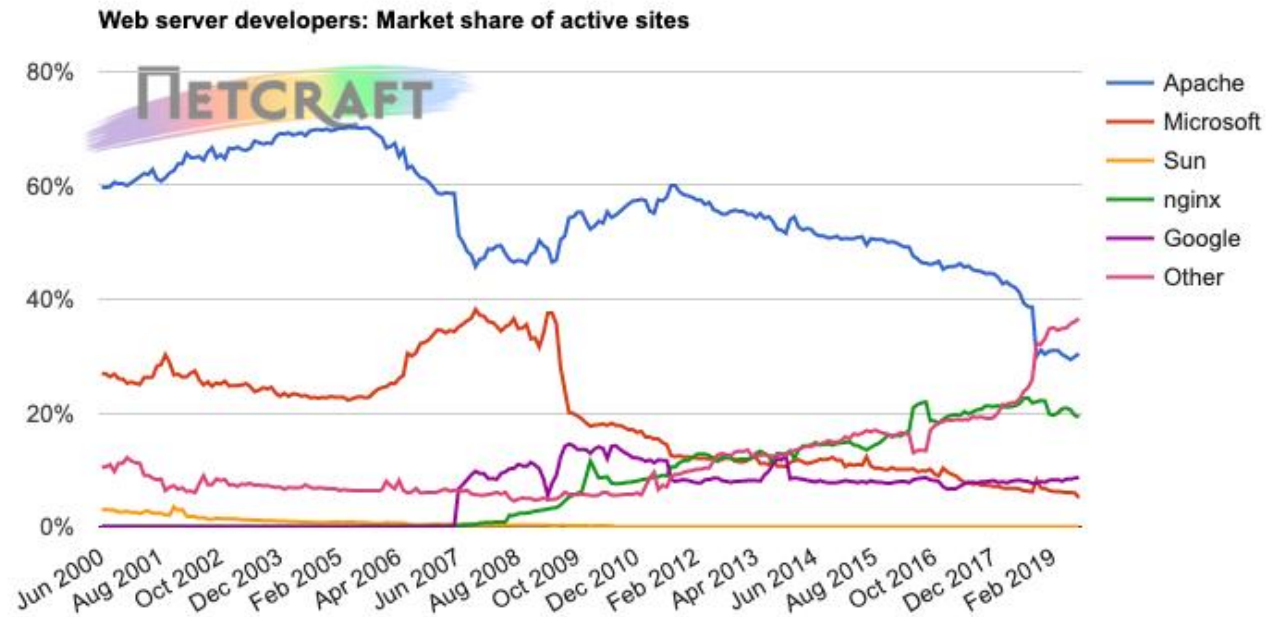


# Web

- Web administration is determining what files to share and how to manage content.
  - IIS
    - Configuration via MMC
    - Supports
      - HTTP
      - FTP
      - SMTP
      - NNTP
      - (and others)
    - Security: lockdown tool and patching
      - <https://support.microsoft.com/en-us/kb/325864>

# Web

- Apache Supports HTTP
- Version 2.x config in `httpd.conf`
- V1.2 - `httpd.conf`, `srm.conf`, `access.conf`
- Version 2.4.43 current



<https://news.netcraft.com/archives/2017/08/29/august-2017-web-server-survey.html#more-25668>



# FTP

- Windows
  - *FTP runs as a sub-service to IIS*
  - *Configuration via Internet Service Manager (MMC to Internet Information Services)*
- Linux
  - *FTP runs more independently*
  - *Configuration through .conf file (for example, using vsftp and vsftpd.conf)*
- Avoid anonymous logins unless specifically needed



# File Systems



- The primary problem is how to give a consistent view of the system across multiple hosts
- Windows network filesystem access
  - *DFS – Combines multiple Windows shares into a single “root” for easy access*
  - *Shares – SMB (CIFS) protocol used to allow access to files on one computer from another*
    - Enabling sharing
    - Creating a share
      - *GUI – Explorer or via MMC*
      - *Command line – net share or rmtshare*



# File Systems



- Viewing available shares
  - *GUI – Explorer or via MMC*
  - *Command line – net share or net view*
- Connecting to a share
  - *GUI - Explorer*
  - *Command line – net use*



# File Systems



- Linux network filesystem access
  - *Network File System - NFS protocol used to allow file sharing*
  - *Enabling NFS*
    - Nfsd
    - NFS also relies on rpc.mountd, rpc.nfsd, and portmap
  - *Creating a share*
    - /etc/exports
      - */etc/exports lists directories that a server exports to its clients.*
      - *Each line in the file specifies a single directory.*



# File Systems

- The syntax of a line in the /etc/exports file is:  
directory [host][option][,option]
  - *The directory is the full path name of the directory.*
  - *Option can designate a simple flag such as ro, rw, sync, or root\_squash.*
  - *The server automatically exports these when the NFS server is started.*
  - *These exported directories can then be mounted by clients.*





# File Systems



- Considering the exportation of a parent directory in a tree that includes one or more child directories.
  - *If you mount the parent directory, would you expect to see the child directories?*
    - In some implementations, you will see the child directories, but with no data beneath them.
    - In others, including RH, you will see the child directories and data
    - Use the `hide` and `no_hide` options if you want to set the entire sub-tree as hidden or visible

# File Systems

***/usr/games***

***box1(ro) comp2(ro) 10.0.1.9(ro)***

***/home***

***box2.external.net(rw,no\_root\_squash)***

***/var/tmp***

***/usr/lib***

***clients \*.internal.net(rw)***

- Entry #1 - */usr/games* can be mounted by the systems named *box1*, *comp2*, and *sys3*. (They can read data/run programs, but they cannot write in the directory)
- Entry #2 - */home* can be mounted by the system *box2* and that root access is allowed for the directory
- Entry #3 - any client can mount */var/tmp* (Note: no access list)
- Entry #4 - specifies an access list designated by the netgroup *clients*. Machines designated as belonging to the netgroup (a NIS feature) *clients* can mount the */usr/lib* directory from this server; also hosts from *internal.net* can access with read and write permissions



# File Systems

*/usr/sbin/exportfs -a*

- Exportfs can also be used to add/remove shares “on the fly”
- *Viewing available shares*
  - showmount -e
- *Diagnostics on messages set via NFS*
  - nfsstat
- *Connecting to a share*
  - Establish local mount point and mount share
    - *mount -t nfs server:/share/mnt/mymntpoint*
  - Use fstab
    - *Mounts during system boot*



# File Systems



- What about users?
  - *Users are dealt with by assuming that UIDs and GIDs are the same on both the server and the client.*
  - *Do you want root on clientbox to be root on serverbox? Do you want user1 on client box to be user1 on serverbox?*
    - root\_squash
      - *Requests from root clients are mapped to the nobody user and group ID so they will only have file privileges associated with other*
    - no\_root\_squash
    - all\_squash
      - *makes every user accessing the exported filesystem to take the user ID of the nobody user*



# Samba

- Samba website [samba.org](http://samba.org)
- 5 Basic Services
  - *file sharing (this is our primary concern)*
  - *network printing*
  - *authentication and authorisation*
  - *name resolution*
  - *service announcement (i.e., Windows browsing).*
- `smbd` and `nmbd` (for NetBIOS name resolution)
  - `smb.conf`
  - `smbstatus`



# Samba

## ➤ SAMBA

- *SMB (Server Message Block, also known as CIFS/Common Internet File System)*
- *Server and Client*
  - *Server allows sharing of file system and/or printers with any system that supports SMB (like both Windows and Linux)*
  - *Client allows for connections to any SMB server*
- *Can act as a Windows Domain Controller*
- *Supports network “browsing”*



# User Management

Chapter 4-5 in Principles of Network and Systems Administration





# Host Management



- The questions
  - *Follow the OS designer's recommended setup? (Often this is insufficient for our purpose)*
  - *Create our own setup?*
  - *Make all machines alike?*
  - *Make all machines different?*
- Vendors support individual hosts
  - *not the network view*



# The Server Room

- Critical hardware needs to be protected from accidental and malicious damage
- Server rooms should have
  - *a lockable door*
  - *cooling equipment to maintain room temperature at or below 20 degrees Celsius*
  - *anti-theft protection*
- Note: backup media should NEVER be stored in the server room.
- Duplicate servers should also be housed in a different physical location



# Further Considerations

- An uninterruptible power supply for all essential equipment
- Single points of failure should be avoided
- Hot standby equipment should be available for minimal loss of uptime in case of failure
- Replaceable hard disks should be considered with RAID protection for continuity
- Protection from natural disasters like fire and floods
- *Note that most countries have regulations about fire control. A server room should be in its own 'fire cell', i.e. it should be isolated by doorways and ventilation systems from neighbouring areas to prevent the spread of fire.*



# Further Considerations

- Important computing equipment can be placed in a Faraday cage to prevent the leakage of electromagnetic radiation, or to protect it from electromagnetic pulses (EMP), e.g. from nuclear explosions or other weaponry.
- Access to cabling should be easy in case of error, and for extensibility.
- Humans should not be able to (unintentionally) touch equipment
- No carpeting or linoleum that causes a build up of static electricity should be allowed
- Humidity should also controlled
  - *too high and condensation can form on components*
  - *too low and static electricity can build up*



# Configuring and Customising Workstations

- Operating systems which have grown out of home computers take the view that remaining disk resources are for the local owner to do with as he or she pleases.
  - *This is symptomatic of the idea that one computer belongs to one user*
  - *Not necessarily true in corporate environments*
  - *Some files are better suited to local storage*
    - *E.g. system files, temporary files*
  - *Diskless workstation failed (X-terminals etc.)*
    - *Poor performance, expensive and high bandwidth requirements*
    - *Thin clients are different*



# Partitioning



- In organising disk space it is good practice to separate:
  - *The operating system.*
  - *Shared - made available for all hosts.*
  - *Space which can be used to optimise local work, e.g. temporary scratch space, space which can be used to optimise local performance (avoid slow networking).*
  - *Space which can be used to make distributed backups, for multiple redundancy.*
  - *Local applications*





# Partitions and Backup/Recovery

- ▶ Partition according to backup/recovery plans
  - ▶ *OS rarely changes*
    - ▶ Exception user accounts and password files
  - ▶ *User data changes continuously*
  - ▶ *Applications change occasionally*
  - ▶ *Temporary/scratch space does not require backup*
- ▶ Example partition names (SCSI)
  - ▶ */dev/sd0a - First partition of disk 0 of the standard disk controller. This is normally the root file system*
  - ▶ */dev/sd0b - Second partition of disk 0 on the standard disk controller. This is normally used for the swap area.*
  - ▶ */dev/sd1c - Third partition of disk 1 on the standard disk controller.*



# Structuring Software

- Software should be separated from the operating system's installed files, so that the OS can be reinstalled or upgraded without ruining a software installation.
  - *Unix-like operating systems have a naming convention.*
- Compiled software can be collected in a special area, with a bin directory and a lib directory
  - *keeps the program search PATH variable simple.*
- Home-grown files and programs which are unique to our own particular site can be kept separate from files which could be used anywhere.
  - *Clearly defines the validity of the files and we see who is responsible for maintaining them.*
- Note:
  - *GNU/Linux: New library directories are added to the file /etc/ld.so.conf.*
  - *Run the command ldconfig so that /etc/ld.so.cache is updated*

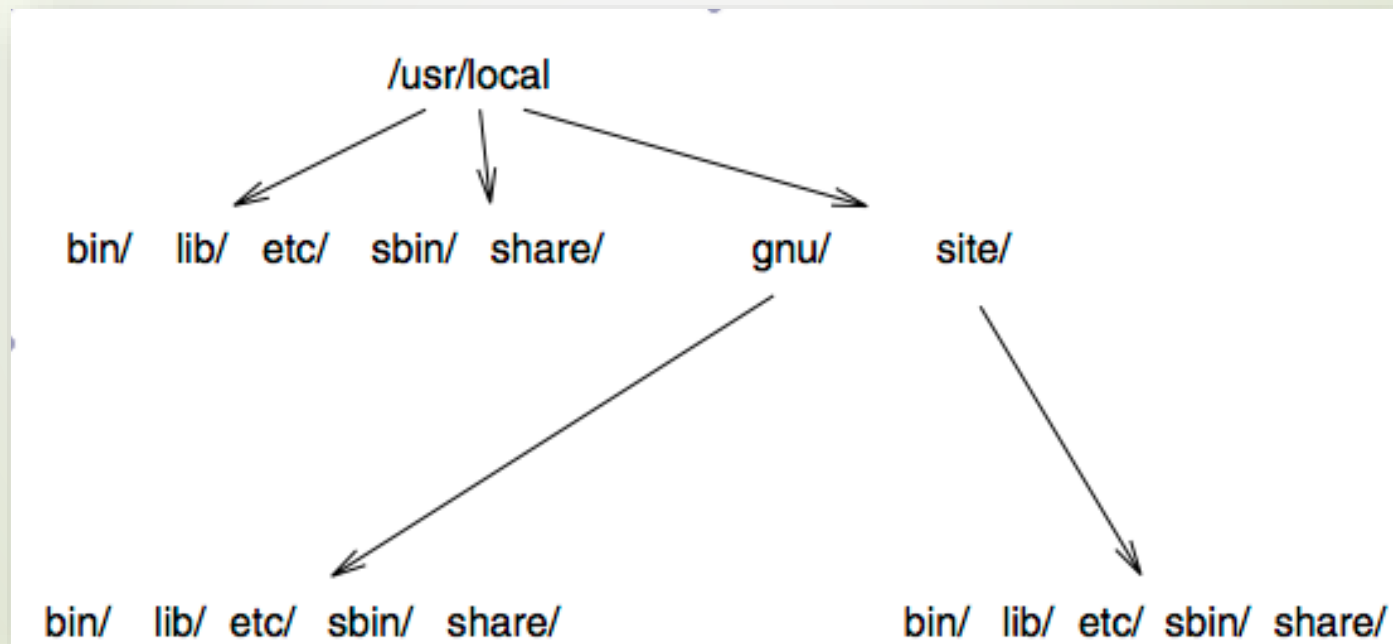


# Structuring Software

- Applications traditionally installed to `/usr/local`.
  - Create subdirectories `/usr/local/bin` and `/usr/local/lib`
  - Avoids “dll hell” that occurs in Windows
- Conventions
  - `bin`- Binaries or executables for normal user programs.
  - `sbin`- Binaries or executables for programs which only system administrators require. Files in `/sbin` are often statically linked to avoid problems with libraries which lie on unmounted disks during system booting.
  - `lib`- Libraries and support files for special software.
  - `etc`- Configuration files.
  - `var`- log files

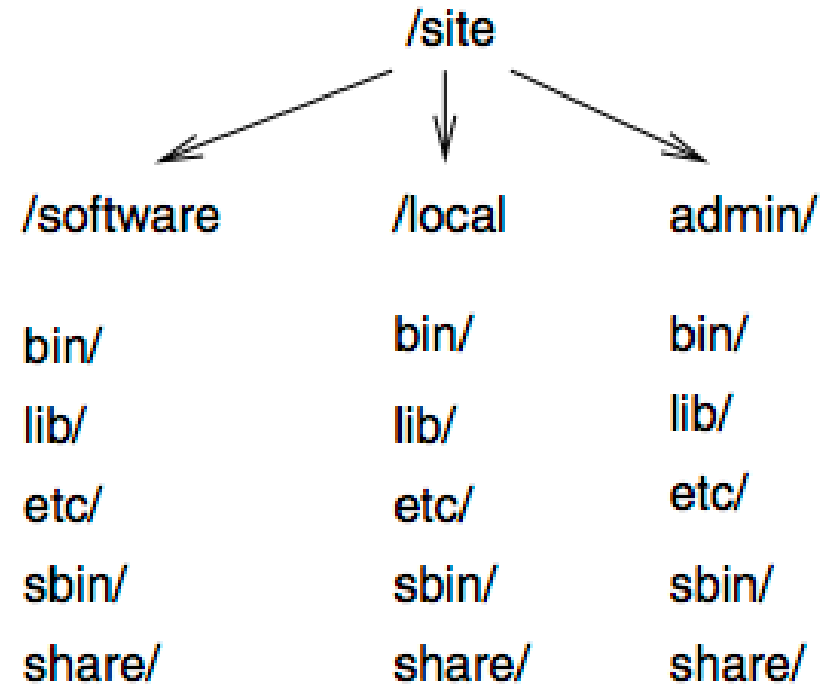
# An Example

- Overkill? What is (possibly) wrong here?



# Another Example

➤ Better?





# Configuration Security



- Be on the lookout for software which is configured, by default, to install itself on top of the operating system.
  - *check the destination before committing to an installation.*
    - `make -n install`
  - *Programs which are replacements for standard operating system components often break the principle of separation.*
    - Software originating in BSD Unix is often an offender, since it is designed to be a part of BSD Unix, rather than an add-on, e.g. sendmail and BIND.
  - *Again never work with root privilege unless necessary*
    - Compile source packages from a user account





# Configuration Security



- Services which are run by the system:
  - *Daemons which carry out essential services run with a user ID which is independent of who is logged on to the system*
  - *Often, such daemons are started as root or the Administrator when the system boots*
  - *In many cases, the daemons do not need these privileges and will function with ordinary user privileges after changing the permissions of a few files.*



# Configuration Security



- *Unix setuid programs:*
  - Unix has a mechanism by which special privilege can be given to a user for a short time, while a program is being executed. Software which is installed with the Unix setuid bit set, and which is owned by root, runs with root's special privileges
- *To do this create a special user in the password database, with no login rights (this just reserves a UID).*
- Used by Apache and mysql – creates users www and mysql.



# Kernel Customisation

- Why?
- Unique applications or hardware
  - *Speakup, boot from USB etc.*
- Always get source from kernel.org
- back up the old kernel
  - *look at how to recover this using ln -s*  
**\$ cp /boot/vmlinuz /boot/vmlinux.old**  
**\$ cd /usr/src**  
**\$ tar xzf /local/site/src/linux-2.2.9.tar.gz**  
**\$ ln -s linux-2.2.9 linux**

# Kernel Customisation

- For each patch file:  
`$ zcat /local/site/src/patchX.gz | patch -p0`
- Choose the correct target architecture  
`$ cd /usr/include`  
`$ rm -rf asm linux scsi`  
`$ ln -s /usr/src/linux/include/asm-i386 asm`  
`$ ln -s /usr/src/linux/include/linux linux`  
`$ ln -s /usr/src/linux/include/scsi scsi`
- Prepare the configuration:  
`$ cd /usr/src/linux`  
`$ make mrproper`



# Kernel Customisation

- Set Kernel parameters
  - make config***
  - or
  - make xconfig***
  - or
  - make menuconfig***
- *make xconfig* requires you to run an X11 applications as root, which is a potential security faux pas.
- The customisation procedure has defaults
  - *The choices are Y to include an option statically in the kernel*
  - *N to not include*
  - *M to include as module support (loadable after boot)*
- Capitalisation denotes default
  - *Note default for processor type*

# Kernel Customisation

- After completing the (very long) configuration sequence
  - *build the kernel:*
    - # **make dep**
    - # **make clean**
    - # **make bzImage**
  - *Move it into place:*
    - # **mv arch/i386/boot/zImage /boot/vmlinuz-2.2.9**
    - # **ln -s /boot/vmlinuz-2.2.9 /boot/vmlinuz**
    - # **make modules**
    - # **make modules-install**
  - *This keeps track of which version is running, while still having the standard kernel name.*
- To alter kernel parameters on the fly, Linux uses a number of writable pseudofiles under /proc/sys  
e.g. `echo 1 >/proc/sys/vm/overcommit_memory` `cat /proc/sys/vm/overcommit_memory`
  - *This can be used to tune values or switch features.*



# Compilation Errors

- ▶ A previous configuration might have been left lying around
  - ▶ ***make clean***
  - ▶ ***make distclean***
- ▶ Check dependencies
- ▶ Errors at the linking stage about missing functions are usually due to missing or un-locatable libraries.
  - ▶ *Check that the LD LIBRARY PATH variable includes all relevant library locations.*
  - ▶ *Are any other environment variables required to configure the software?*



# Compilation Errors

- Sometimes an extra library needs to be added to the Makefile. To find out whether a library contains a function
- C-shell trick:
  - *host% cd /lib*
  - *host% foreach lib ( lib\* ) > echo*
  - *Checking \$lib -----*
  - *> nm \$lib | grep function*
  - *>end•*
- Carefully try to patch the source code to make the code compile.
- Check in news groups whether others have experienced the same problem.
- Contact the author of the program.
  - *Believe it or not this works! e.g. openbiblio*

# User Management

USER FRIENDLY by Illiad

Did you call Mr. Niffle about his connection problem?

Yep.

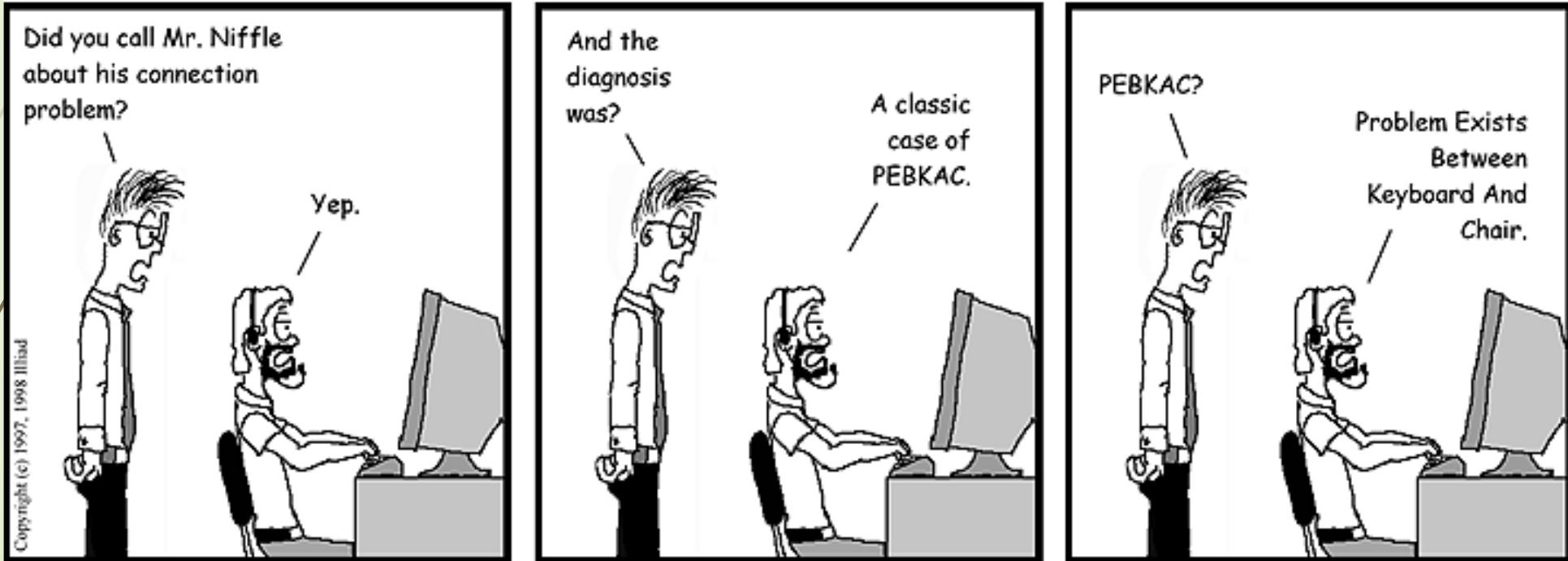
And the diagnosis was?

A classic case of PEBKAC.

PEBKAC?

Problem Exists Between Keyboard And Chair.

Copyright (c) 1997, 1998 Illiad





# User Management



- User management is about interfacing humans to computers.
  - *Accounting: registering new users and deleting old ones.*
  - *Comfort and convenience.*
  - *Support services.*
  - *Ethical issues.*
  - *Trust management and security.*
- Some of these (account registration) are technological, while others (support services) are human issues.



# User Registration



- One of the first issues on a new host is to issue accounts for users.
  - *Tools provided by operating systems for this task are, at best, primitive and are rarely suitable for the task without considerable modification.*
  - *Systems like Windows server and Netware have an advantage in forcing a particular administration model onto the hosts in a network, they can provide straightforward delegation of user registration to anyone with domain credentials.*
  - *Registration of single users under Windows can be performed remotely from a workstation, using*

**`net user username password /ADD /domaincommand`**



# User Registration

- Unix Accounts
  - *Find a unique username, user-id (uid) number and password for the new user*
  - *Update the system database of user accounts*
    - add a line to the file `/etc/passwd` for Unix (or on the centralised password server of a network)
    - Create a login directory (home directory) for the user
    - Choose a shell for the user (if appropriate)
    - Copy some configuration files like `.cshrc` or `.profile` into the new user's directory
- User registration requires different tools and techniques in almost every case.
  - *where should users' home directories be located?*
- `useradd` script assumes that the user will be installed on the local machine under `/home/user`
  - *many users belong to a network and their disk space lies physically on a different host which is mounted by NFS*
- Unix - three different password file formats which increase the awkwardness of distributing passwords

# Password Files

- Traditional

**`mark:Ax7Wc1Kd8ujo2:123:456:Mark Burgess:/home/mark:/bin/tcsh`**

- Shadow (conceals the encrypted form from users)

**`mark:x:123:456:Mark Burgess:/home/mark:/bin/tcsh`**

***In /etc/shadow***

**`mark:Ax7Wc1Kd8ujo2:6445::::::`**

- Blank fields are used for password ageing, history and other expiry mechanisms
- The number is the time at which the password was last changed, measured in the number of days since 1. Jan. 1970.

- BSD 4.4 derived operating systems.

**`mark:Ax7Wc1Kd8ujo2:3232:25::0:0:Mark Burgess:/home/mark:/bin/tcsh`**



# User Groups

- A group is an association of usernames which can be referred to collectively by a single name.
  - File and process permissions can be granted to a group of users.
  - Groups are defined statically by the system administrator.
  - All users belong to the users group (100 in this case)
  - Defined in the `/etc/group` file

**`users::100:user1,mark,user2,user3`**

- The second empty field provides space for a seldom used password

- default groups are defined by the system

**`root::0:root`**

**`other::1:`**

**`bin::2:root,bin,daemon`**

- The names and numbers of system groups vary with different flavours of Unix.
- The root group has superuser privileges



# User Groups

- Unix groups can be created for users or for software
  - e.g. *www-data*
- In addition to the names listed in the group file, a group also accrues users from the default group membership in field four of */etc/passwd*.
  - *If the group file had the groups:*
    - users::100:**
    - mysql::36:**
    - ftp::99:**
    - www::500:www**
    - www-data::501:www,toreo,mark,geirs,sigmunds,mysql,ulfu,magnem**
    - privwww::502:**
  - *Users group would contain every user on the system*
  - *the group ftp contains no members at all*
    - to be used only by a process which the system assigns that group identity,
    - whereas *www-data* contains a specific named list



# Account Policy



- Users are the foremost danger to a computing system
  - *s o the responsibility of owning an account should not be dealt out lightly*
- What should an account policy contain?
  - *Rules about what users are allowed/not allowed to do.*
  - *Specifications of what mandatory enforcement users can expect*
  - *Password strength/expiry*

# Disabling Accounts

- Change the default shell in /etc/passwd to a script

```
#!/bin/sh
```

```
echo "/local/bin/blocked.passwd was run" | mail sysadm /usr/bin/last -10
```

```
| mail sysadm
```

```
message=' Your account has been closed because your password was found  
to be vulnerable to attack. To reopen your account, visit the admin  
office, carrying some form of personal identification. '
```

```
echo "$message"sleep 10
```

```
exit 0
```

- Will not block xdmlogin manager

- A more secure method is to simply replace their encrypted password with \*, which prevents them from being authenticated.



# Environment



- Everything in Unix is configurable
  - *'dot' files for setting defaults*
  - *every file has its own syntax*
  - *.cshrc*
    - supply a default 'read commands' file for this shell
    - This should set a path which searches for commands, a terminal type and any environment variables that a local system requires.
  - *.profile or .bashrc*
    - set a PATH variable, terminal type and environment variables that the system requires.



# Environment



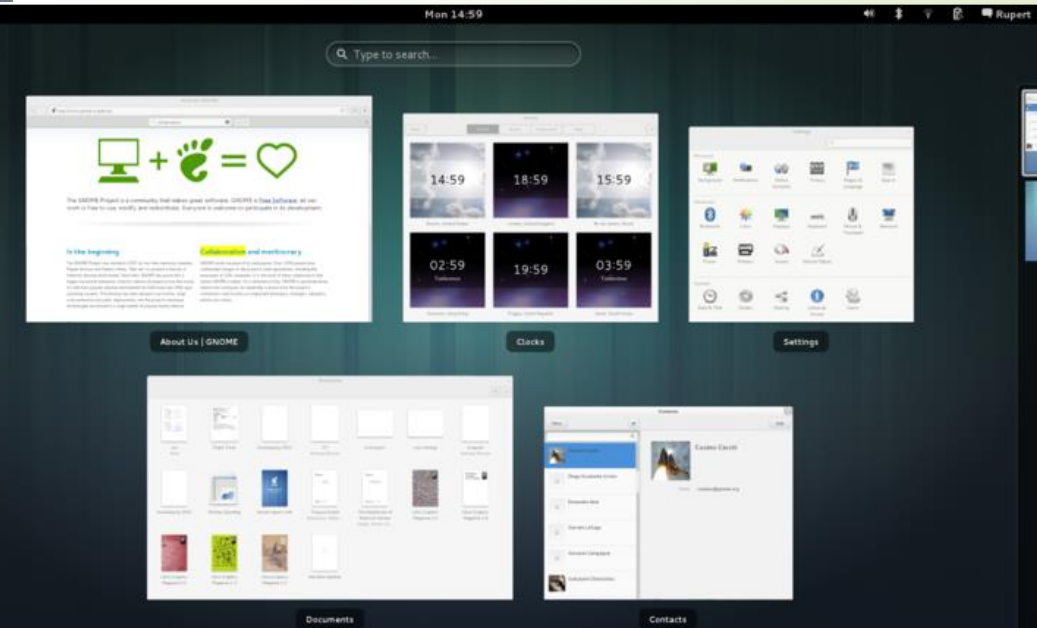
## ■ `.xsession`

- This file is read by the Unix xdm login service.
- Specifies what windows and what window manager will be used when the Xwindows system is started.
- The file is a shell script which should begin by setting up applications in the background and executing a window manager in the foreground.
- xdm provides a login prompt window. GNU/Linux makes use of the obsolete command `startx` which starts the X-windows system from a `tty-shell`.
  - *The older `startx` system used the `.xinitrc` file, whereas xdm uses `.xsession`.*

## ■ `.mwmrc`

- This file configures the default menus etc. for the mwm window manager and the Common Desktop Environment (CDE) window manager







# Environment

- ▶ A shell setup defines
  - ▶ *a terminal type*
  - ▶ *default prompt*
  - ▶ *appropriate environment variables, especially a command path*
- ▶ global shell configuration files which are read for every user are located in /etc or /etc/default
  - ▶ *Use overridable includes rather than global files*
    - # cshrc file (for tcsh)**
    - source ../setupfiles/cshrc-global**
    - # Place own definitions below**
    - alias f finger**
    - alias ed emacs**
  - ▶ *sourcedirective used to read in a file of global C-shell definitions*



# Environment



- a local shell configuration is to set up a command path and a library path for software.
- Since the command path is searched in order, we can override operating system commands with local solutions, simply by placing site-dependent binaries at the start of the path.



# Root Account

- A privileged account has potentially dangerous consequences for the system
- Ok to give a powerful shell to root but...
  - *Make sure the tcsh or bash shell is physically stored on the root partition*
    - When a Unix system boots, only the root partition is mounted
    - Referencing a shell which is not available will render the host unbootable.
- The superuser's PATH variable should never include '.'
  - *current directory, may lead to executing wrong commands*
- Root should never
  - *run X11*
  - *log in directly (unless the system is in single user mode or on the console)*
  - *read or reply to its own E-mail (this should be sent to a system administration group)*
  - *log in over an unencrypted channel.*



# User Support



- Level of support offered to users is a matter of policy
  - *Self sufficient?*
  - *or Support?*
- Usually a mix
  - *Training users*
  - *Helping users*
  - *Documenting and providing FAQs*



# Controlling User Resources

- Every system has a mixture of passive and active users
  - *Passive users utilise the system often minimally;*
    - Accepting the choices which have been made for them
    - Are often not even aware of what files they have
    - Seldom make demands other than when things go wrong
    - Can be a security risk, unaware of the consequences of their actions



# Controlling User Resources

- ▶ *Active users*
  - ▶ Follow details of system development
  - ▶ Frequently find errors in the system
  - ▶ Contact system administrators frequently, demanding upgrades of their favourite programs
  - ▶ Can be of great help to a system administrator in testing
  - ▶ Can help the passive users.
  - ▶ Note: active users are not authorised staff
- ▶ *Active users need to understand that, while their skills are appreciated, they do not decide system policy: they must obey it.*





# Resource Consumption



- Users almost never throw away files unless forced
  - *Keep a separate partition for problem users' home directories and/or enforce strict quotas*
  - *Some files are temporary by definition.*
    - e.g. \*.o files, files which can easily be regenerated from source like TeX \*.dvi files, cache files etc
  - *Some files can be defined as temporary as a matter of policy.*
    - \*.mp3, video formats etc
  - *Core dumps*
- Create scripts to handle these



# Killing Old Processes

- Processes sometimes do not get terminated when they should.
  - *Users forget to log out*
  - *Poorly written software does not properly kill its processes when a user logs out.*
  - *Background programs simply crash or hang.*
  - *One way to clean up processes is to look for user processes which have run for more than a day*
    - (Note that the assumption here is that everyone is supposed to log out each day and then log in again the next day – that is not always the case.)
  - *Hung processes can be seen by running `ps -aux`*



# Deleting Users



- Users who leave an organisation eventually need to be deleted from the system.
  - *It is advisable to keep old accounts for a time*
- Before deleting a user
  - *Backup the data*
  - *Remove the following:*
    - Account entry from the password database
    - Personal files
    - E-mail and voice mail and mailing lists
    - Removal from groups and lists (e.g. mailing lists)
    - Removal of cron and batch tasks
    - Revocation of smartcards and electronic ID codes.
  - *So script/use LDAP*



# Computer Usage Policy



- Should make it clear what is considered acceptable behaviour
- Avoid law-suits
- Policy should include:
  - *What all parties should do in case of dismissal*
  - *What all parties should do in case of security breach*
  - *What are users' responsibilities to their organisation?*
  - *What are the organisation's responsibilities to their users?*
  - *The policy has to take special care to address the risks of using insecure operating systems (Windows 95, 98, ME and Macintosh versions prior to MacOSX)*
- Example policy available on Moodle



# Configuration and Maintenance

- Two overlapping issues
  - *How to make systems operate as intended*
  - *How to maintain this state*
- Configuration methods
  - *Configuration text file*
    - Sendmail
  - *XML file*
    - plists
  - *Database format (registry)*
    - Windows
  - *Transmitted protocol (ASN.1)*
    - SNMP



# Configuration and Maintenance



- Systems tend to a state of disorder unless a disciplined policy is maintained, because they are exposed to random noise through contact with users.
- Deviation from a system's ideal state can be smoothed out by a counteractive response. If these two effects are in balance, the system will stay in equilibrium.
  - *Equilibrium is a 'fixed point' of the system behaviour. System administration is about finding such fixed points and using them to develop policy. The time scales over which errors occur and which repairs are made are clearly important.*





# System Configuration Policy



- System administration is often a collaborative effort between several administrators.
  - *Necessary to have agreed policies for working so that everyone knows how to respond to 'situations' which can arise*
    - e.g. patching SVN kills Apache
  - *Summarises the attitudes of an organisation to its members*
    - often embodies security issues.
  - *As Howell\* cites 'We have met the enemy, and he is us!'*
  - *A system policy should contain the issues we have been discussing and addressed at each level: network level, host level, user level.*

B. Howell and B. Satdeva. We have met the enemy. An informal survey of policy practices in the internetworked community. Proceedings of the Fifth Large Installation Systems Administration Conference (LISA V) (USENIX Association: Berkeley, CA), page 159, 1991.



# Change Management



- Can upgrades, redesign and replacement be done without disruption to service?
- Planning changes of infrastructure can be dealt with using two general strategies:
  - *Deconstruction followed by reconstruction.*
  - *Change of policy description followed by convergence to a new state.*
- Change management then becomes a reconstruction of infrastructure.
- The amount of work required to perform large changes through differential adjustment grows significantly with the magnitude of the change



# Change Management



- Many system administrators feel more comfortable with starting from scratch when large changes need to be made
- Why would people go over to the new, if the old still works?
  - e.g. *XP to Vista Windows 7 8 10*
- Temporary redundancy of service is a sensible precaution in a missioncritical environment.
- Securing predictability during a change is a tricky business as the conditions under which a system is performing its function are changing
- Change management can thus be viewed as a problem in risk or fault management



# Change Management



- Change management is about planning the timing and deployment of upgrades and overhauls to the system.
  - *How will a change propagate into the rest of the system?*
- Checklist for change management
  - 1. *Decide on the change.*
  - 2. *Map out the repercussion network (dependencies) of the change, as far as possible.*
    - *Some dependencies might be hidden or be beyond your control, e.g. operating system upgrade changes.*
  - 3. *Revise policy for each affected component in the system to reflect the change.*
  - 4. *Inform users of the impending change and wait for comments.*
  - 5. *Incorporate any user comments into the policy change.*
  - 6. *Lock the system to prevent incomplete reconfiguration from being a hazard to the system.*
  - 7. *Make the changes.*
  - 8. *Unlock the system.*
  - 9. *Inform users that the change has been implemented.*

# Clock Synchronisation

- One of the most fundamental tasks in a network is to keep the clocks on all hosts synchronised.
- Many security and maintenance issues depend on clocks being synchronised correctly.
  - *The clocks on cheap PC hardware tend to drift very quickly*
  - *The rdate command sets the local clock according to the clock of another host.*
  - *If missing it may be simulated by a script:*

```
#!/bin/sh
```

```
# Fake rdate script for linux - requires ssh access on server
```

```
echo Trying time
```

```
serverDATE='/bin/su -c '/usr/bin/ssh time-server date' remote-user'
```

```
echo Setting date string...
```

```
/bin/date --set="$DATE"
```

# Clock Synchronisation

- Better to use the NTP protocol
  - *The network time protocol daemon xntpd is used to synchronise clocks from a reliable time-server.*
  - *Two configuration files are needed*
    - */etc/ntp.conf and /etc/ntp.drift.*
    - */etc/ntp.conf contains the following, where the IP address is that of the master time-server:*

***driftfile /etc/ntp.drift***

***authdelay 0.000047***

***server 128.39.89.10***

- */etc/ntp.drift* file must exist, but its contents are undetermined.





# Automation of Configuration



- Tivoli is probably the most advanced and wide-ranging product available.
- It is a Local Area Network (LAN) management tool based on CORBA and X/Open standards
- Advertised as a complete management system to aid in both the logistics of network management and an array of configuration issues.
- It addresses the problems of system administration from the viewpoint of the business community, rather than the engineering or scientific community.



# Automation of Configuration


- Disadvantage:
  - *focus on application-level software rather than core system integrity.*
  - *Also it lacks abstraction methods for coping with real-world variation in system setup.*
- Utilises encrypted communications and client-server interrelationships to provide functionality including software distribution and script execution.
- Tivoli can activate scripts but the scripts themselves are a weak link. Activated by:
  - *Execute by hand when required.*
  - *Schedule tasks with a cron-like feature.*
  - *Execute an action (run a task on a set of hosts, copy a package out) in response to an event.*
    - Tivoli's Enterprise Console includes a language Prolog for attaching actions to events.
- Client-server reliance could also be a problem if network communications are down



# Automation of Configuration



- HP BTO (OpenView) is a commercial product based on SNMP network control protocols.
  - *Openview aims to provide a common configuration management system for printers, network devices, hosts*
  - *The advantage of Openview is a consistent approach to the management of network services;*
  - *The principal disadvantage, is that the use of network communication opens the system to possible attack from hacker activity.*
  - *Alerts Administrator only, no auto repair*
- Sun's Solstice system is a series of shell scripts with a graphical user interface which assists the administrator of a centralised LAN, consisting of Solaris machines, to initially configure the sharing of printers, disks and other network resources.



# Automation of Configuration

- GNU/Linux community has been engaged in an effort to make GNU/Linux more user-friendly by developing any number of graphical user interfaces for the system administrator
  - *These tools offer no particular innovation other than the novelty of a more attractive work environment.*
  - *Most of the tools are aimed at configuring a single stand-alone host, perhaps attached to a network.*
  - *Recently, several projects have been initiated to tackle clusters of Linux workstations*



# Preventative Maintenance

- Damage and loss can come in many forms:
  - *Hardware failure, resource exhaustion (full disks, excessive load), security breaches and by accidental error.*
- General provisions for prevention
  - *Do not rely exclusively on service or support contracts with vendors*
  - *Educate users by posting information in a clear and friendly way*
  - *Make rules and structure as simple as possible, but no simpler.*
  - *Keep valuable information about configuration securely, but readily, available*
  - *Document all changes and make sure that co-workers know about them*
  - *Do not make changes just before going away on holiday*
  - *Be aware of system limitations, hardware and software capacity.*
  - *'If it ain't broke, don't fix it'*
    - *but still aim for continuous but cautious improvement.*
  - *Duplication of service and data gives us a fallback which can be brought to bear in a crisis.*



# Faults

- IEEE classification of software anomalies
  - *Operating system crash*
  - *Program hang-up*
  - *Program crash*
  - *Input problem*
  - *Output problem*
  - *Failed required performance*
  - *Perceived total failure*
  - *System error message*
  - *Service degraded*
  - *Wrong output*
  - *No output*





# Human Edge Faults



- Management error
- Miscommunication
- Forgetfulness
- Misunderstanding/miscommunication
- Misidentification
- Confusion/stress
- Ignorance
- Carelessness
- Slowness of response
- Random procedural errors
- Systematic procedural errors
- Inability to deal with complexity
- Inability to cooperate with others.



# Fault Analysis



- Always eliminate the obvious first
  - *Physical layer first*
    - Amazing how often a cable has fallen out
    - Check power
  - *Work systematically*
- Establishing cause and effect
  - *Gather evidence from users and from other tests*
  - *Make an informed guess as to the probable cause*
  - *Try to reproduce (or perhaps just fix) the error.*



# Monitoring



- Set Benchmarks
  - *Otherwise there is nothing to compare current performance to!!*
- Two issues:
  - *user perception of performance (interactive response time)*
  - *system throughput*
  - *Need to choose the criterion to meet*



# Monitoring



- ▶ Look at what resources are being tested
  - ▶ *What processes are running*
  - ▶ *How much available memory the system has*
  - ▶ *Whether disks are being used excessively*
  - ▶ *Whether the network is being used heavily*
  - ▶ *What software dependencies the system has (e.g. DNS, NFS).*
    - ▶ If one host dependent on another then the dependent host will always be limited by the host on which it depends. This is particularly true of file-servers (e.g. NFS, DFS, Networkware distributed filesystems) and of the DNS service.



# Monitoring



- Windows
  - *Process manager and performance monitor*
- Unix-like systems
  - *BSD compatible – ps aux*
  - *System V ps -efl*
  - *Top is handy as well*
  - *vmstat for virtual memory*
- Many other tools available