嵌入式系統
Lab02: RPi 基礎操作與管理

銘傳大學 電腦與通訊工程學系 羅嘉寧

Bash shell

Linux 基礎運作—BASH shell

分享指引

3

- Bash Shell
- 查閱檔案內容指令
- vi 與 vim 程式編輯器
- 資料流重導向
- 管線命令
- 檔案/指令搜尋
- 正規表示法
- 工作管理(job control)
- 精選範例

什麼是Shell

Shell 的角色

5

- 使用者可以透過 shell 直接控制核心，以達成各項任務

什麼是Shell

Linux shell

6

- Linux 合法的 shell
  - /etc/shells
    - /bin/sh
    - /bin/bash
    - /sbin/nologin
    - /bin/ksh
    - /usr/bin/ksh
    - /bin/tcsh
    - /bin/csh
    - /bin/zsh
- 預設的 shell
  - /bin/bash
- 使用者修改預設 shell
  - chsh –s 'shellname'

Bash

bash 的功能 7

- 命令編修能力 (類似 DOS 的 doskey 功能)
- 命令與檔案補全功能：
- 命令別名(alias)設定功能
- 工作控制(jobs)、前景背景控制：
- Shell scripts 的強大功能
- 萬用字元！

Bash

bash 的慣用按鍵 8

- [Tab] 按鍵
  - [Tab] 接在一串指令的第一個字的後面，則為命令補全；
  - [Tab] 接在一串指令的第二個字以後時，則為『檔案補齊』！
- [Ctrl]-c 組合鍵
  - 可以中斷目前正在執行中的程式
- [Ctrl]-d 組合鍵
  - 結束某些程式所需的輸入資訊
- [Shift]-[Pageup]/[Shift]-[Pagedown]
  - 在終端機模式下，向前/向後翻頁

Bash

bash 的變數 9

- 變數的設定方式：
  - 變數名稱=“變數內容”
- 變數設定規則
  - 變數與變數內容以等號『=』來連結，且等號兩邊不能直接接空白字元
  - 變數名稱只能是英文字母與數字，且數字不能是開頭字元；
  - 可以使用雙引號『 “ 』或單引號『 ‘ 』來將變數內容結合起來
    - 雙引號內的特殊字元可以保有變數特性，
    - 單引號內的特殊字元則僅為一般字元；
  - 跳脫字元『 \ 』來可特殊符號 變成一般字元；
  - 指令內的指令可用『 `command` 』或『$(command)』
  - 可以 export 來使變數變成環境變數， 如『export PATH』；
  - 取消變數的方法為：『unset 變數名稱』。

Bash

變數的呼叫/使用 10

- 變數的呼叫：
  - echo $var
  - echo ${var}
- 變數的使用：
  - mkdir ‘~dmtsai’ →建立名為 ~dmtsai 的目錄
  - echo “$PATH” →叫出PATH變數的內容
  - kversion=$(uname –r) →設定kversion為核心版本
  - echo “\$PATH” →顯示 $PATH 在螢幕上
  - set →顯示目前所有的變數

Bash

影響bash操作環境的變數 11

- 幾個較重要的，會影響環境的變數
  - HOME 家目錄，即 ~ 所代表的目錄
  - MAIL 輸入 mail 即可收信(信箱)
  - HISTSIZE history 的紀錄比數
  - LANG 語系資料 (可用 locale –a)
  - PATH 指令執行查詢目錄順序
  - PS1 命令提示字元(man bash)
  - $ 此 shell 的 PID 號碼
  - ? 上個指令執行回傳值 (0為正確)

Bash

變數的有效範圍 12

- 環境變數
  - 當啟動一個 shell ，作業系統分配一記憶區塊給 shell 使用，此區域之變數可以讓子程序存取；
  - 利用 export 功能，可以讓變數的內容寫到上述的記憶區塊當中(環境變數)；
  - 當載入另一個 shell 時 (亦即啟動子程序，而離開原本的父程序了)，子 shell 可以將父 shell 的環境變數所在的記憶區塊導入自己的環境變數區塊當中。

Bash

bash 的內建指令

13

- bash 本身有提供一些內建指令
  - cd, pwd
  - echo
  - jobs, fg, bg
  - history
  - type
  - …
  - man cd (可以看到很多喔！)

Bash

歷史命令

14

- 呼叫過去下達過的指令
  - history [-n]
- history 的紀錄檔
  - ~/.bash_history
- 指令的應用
  - !!
  - !vi
  - !50
  - !-5

Bash

命令別名

15

- 建立簡單好用的新指令
  - alias ll= 'ls –al'
  - alias h=history
- 取消的方式
  - unalias h

Bash

指令執行的順序

16

- 若於多個地方擁有相同指令，如 ls, echo
  - 絕對路徑/相對路徑直接執行某程式
  - 命令別名所載 (alias)
  - bash 內建指令
  - 由 PATH 所找到的指令
  - 可用 type –a 指令 來檢查！

Bash

環境參數設定檔

17

- login-shell：登入時會讀取的設定檔
  - /etc/profile
  - ~/.bash_profile, ~/.bash_login, ~/.profile
- non-login-shell：非登入時所取得 bash 的環境
  - 例如 X 畫面下的終端機
  - 在 bash 中執行 bash
  - 執行 script 時
  - ~/.bashrc
- 不登出立刻讓設定檔生效的方法
  - . ~/.bashrc
  - source ~/.bashrc

Bash

萬用字元

18

- 常見的bash環境萬用字元
  - * 0到無窮多個任意字元
  - ? 一個任意字元
  - [a-c] 一個在中刮號中的字元存在
  - [^a-c] 一個不在中刮號中的字元存在
- 一些範例
  - 具有3個字母的檔案： /etc/???
  - 具有數字的檔名： /etc/*[0-9]*
  - 大寫字元的檔案： /etc/*[[:upper:]]*

Bash

身份切換

19

- bash 環境的操作
  - 盡量不要使用 root 身份，以免不小心影響系統
  - 一般使用者想要切換身份，可用 su –
    - 轉身份成為root： su – (然後輸入root密碼)
  - 離開 su - 的環境，使用exit來回到原本的身份
  - 切換成為其他使用者時
    - su – username
    - 需要輸入該使用者的密碼才行
    - root 變身成為他人，不需要輸入密碼

查閱檔案內容

cat 與 nl

22

```
[root@linux ~]# cat [-AEnTv]
參數：
-A  ：相當於 -vET 的整合參數，可列出一些特殊字符～
-E  ：將結尾的斷行字元 $ 顯示出來；
-n  ：列印出行號；
-T  ：將 [tab] 按鍵以 ^I 顯示出來；
-v  ：列出一些看不出來的特殊字符
```

```
[root@linux ~]# nl [-bnw] 檔案
參數：
-b  ：指定行號指定的方式，主要有兩種：
      -b a ：表示不論是否為空行，也同樣列出行號；
      -b t ：如果有空行，空的那一行不要列出行號；
-n  ：列出行號表示的方法，主要有三種：
      -n ln ：行號在螢幕的最左方顯示；
      -n rn ：行號在自己欄位的最右方顯示，且不加 0 ；
      -n rz ：行號在自己欄位的最右方顯示，且加 0 ；
-w  ：行號欄位的佔用的位元數。
```

查閱檔案內容指令

查閱檔案內容

head 與 tail

23

```
[root@linux ~]# head [-n number] 檔案
參數：
-n  ：後面接數字，代表顯示幾行的意思
範例：
[root@linux ~]# head /etc/man.config
# 預設的情況中，顯示前面十行！若要顯示前 20 行，就得要這樣：

[root@linux ~]# head -n 20 /etc/man.config
```

```
[root@linux ~]# tail [-n number] 檔案
參數：
-n  ：後面接數字，代表顯示幾行的意思
範例：
[root@linux ~]# tail /etc/man.config
# 預設的情況中，顯示最後的十行！若要顯示最後的 20 行，就得要這樣：

[root@linux ~]# tail -n 20 /etc/man.config
```

tail +5 /etc/man.config → 第五行以後的資料通通印出來
tail –f /var/log/messages →持續追蹤該檔案的內容

查閱檔案內容

常用來查詢檔案內容的指令

21

- cat 由第一行開始顯示檔案內容
- tac 從最後一行開始顯示
- nl 顯示的時候，順道輸出行號！
- more 一頁一頁的顯示檔案內容
- less 與 more 類似，且可以往前翻頁！
- head 只看頭幾行
- tail 只看尾巴幾行
- od 以二進位的方式讀取檔案內容

查閱檔案內容

查詢檔案屬性

24

```
[root@linux ~]# ls [-aAdfFhilRS] 目錄名稱
[root@linux ~]# ls [--color={none,auto,always}] 目錄名稱
[root@linux ~]# ls [--full-time] 目錄名稱
參數：
-a  ：全部的檔案，連同隱藏檔( 開頭為 . 的檔案) 一起列出來～
-A  ：全部的檔案，連同隱藏檔，但不包括 . 與 .. 這兩個目錄，一起列出來～
-d  ：僅列出目錄本身，而不是列出目錄內的檔案資料
-f  ：直接列出結果，而不進行排序 (ls 預設會以檔名排序！)
-F  ：根據檔案、目錄等資訊，給予附加資料結構，例如：
      *：代表可執行檔； /：代表目錄； =：代表 socket 檔案； |：代表 FIFO 檔案；
-h  ：將檔案容量以人類較易讀的方式(例如 GB, KB 等等)列出來；
-i  ：列出 inode 位置，而非列出檔案屬性；
-l  ：長資料串列出，包含檔案的屬性等等資料；
-n  ：列出 UID 與 GID 而非使用者與群組的名稱 (UID與GID會在帳號管理提到！)
-r  ：將排序結果反向輸出，例如：原本檔名由小到大，反向則為由大到小；
-R  ：連同子目錄內容一起列出來；
-S  ：以檔案容量大小排序！
-t  ：依時間排序
```

vi 與 vim 程式編輯器

vim的環境設定

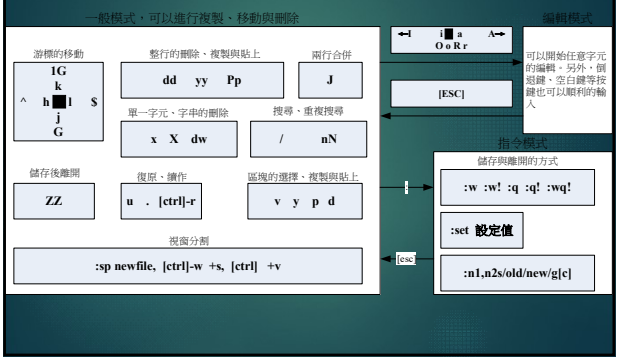
- vim 尚有非常多的設定資訊，包括有：
  - :set nu (行號)
  - :set autoindent(縮排)
  - :set textwidth=80(行寬)
  - :set hlsearch(高亮度反白)
  - :syntax {on|off}(語法的正確性與否檢驗)
- 各項目可寫入設定檔，亦即：~/.vimrc

vi 是什麼

- vi 與 vim
  - vi 是一個文書編輯器，在各主要 Unix like 系統中均存在
  - vi 會被其他軟體所呼叫，例如 crontab
  - vim 是加強版的 vi，可以具有顏色顯示、語法校驗等功能
  - vim 應該可被稱為程式編輯器

資料流重導向

vi 的慣用按鈕

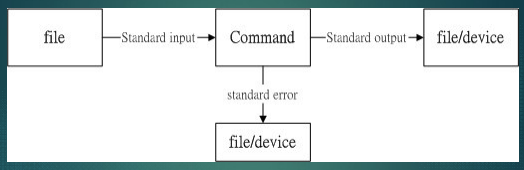


指令的訊息

- 每個指令的執行結果可能都會有輸出的資料
  - 正確的資料：Standard Output (STDOUT)
  - 錯誤的資料：Standard Error Output(STDERR)
- 指令在運作時，可能需要讀入資料
  - 輸入的資料：Standard Input
  - 可能由檔案，或者是鍵盤輸入而來。

資料流重導向

訊息的顯示方式 31



- STDOUT 與 STDERR 預設都輸出至螢幕上
  - >, >> 可將STDOUT轉傳到其他檔案/裝置
  - 2>, 2>> 可將STDERR轉傳到其他檔案/裝置
  - < 可代表讀入的資料。

資料流重導向

結束輸入關鍵字 34

```
[root@linux ~]# cat > catfile <<eof
> This is a test testing
> OK now stop
> eof  <==輸入這個玩意兒，嘿！立刻就結束了！
```

- 透過 <<keyword 來結束鍵盤的輸入

資料流重導向

一個範例 32

```
[dmtsai@linux ~]$ find /home -name testing
find: /home/test1: Permission denied   <== Starndard error
find: /home/root: Permission denied    <== Starndard error
find: /home/masda: Permission denied   <== Starndard error
/home/dmtsai/testing                   <== Starndard output
```

```
[dmtsai@linux ~]$ find /home -name testing > list_right 2> list_error
```

- 透過 > 與 2> 將原本由螢幕輸出的資料分別轉送到 list_right 與 list_error 當中。
- 螢幕不會有任何訊息的產生

資料流重導向

資料流重導向的使用時機 35

- 當螢幕輸出的資訊很重要，而且我們需要將他存下來的時候；
- 背景執行中的程式，不希望他干擾螢幕正常的輸出結果時；
- 一些系統的例行命令（例如寫在 /etc/crontab 中的檔案）的執行結果，希望他可以存下來時；
- 一些執行命令，我們已經知道他可能的錯誤訊息，所以想以『 2> /dev/null 』將他丟掉時；
- 錯誤訊息與正確訊息需要分別輸出時。

資料流重導向

特殊寫法 33

```
[dmtsai@linux ~]$ find /home -name testing > list_right 2> /dev/null
```

```
[dmtsai@linux ~]$ find /home -name testing > list 2> list  <==錯誤寫法
[dmtsai@linux ~]$ find /home -name testing > list 2>&1     <==正確寫法
```

- 可用垃圾桶 (/dev/null) 去除不要的資訊
- 可用 2>&1 將所有訊息導向同一個檔案

資料流重導向

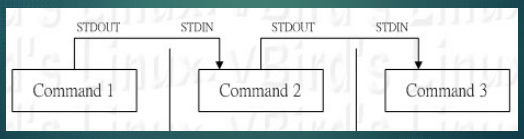
連續指令的下達 36

- 逐次執行指令
  - cmd1 ; cmd2 ; cmd3
- 前一個指令回傳值為0後面才執行
  - cmd1 && cmd2
- 前一個指令回傳值非為0後面就執行
  - cmd1 || cmd2
- 綜合處理
  - cmd1 && cmd2 || cmd3

管線命令

管線命令 38

- 管線命令的意義
  - 可以處理來自前一個指令的STDOUT
  - 不處理STDERR的資訊
  - cat, more, less都是管線命令
  - ls, cp...並非管線命令



cut 39

```
[root@linux ~]# cut -d'分隔字元' -f fields
[root@linux ~]# cut -c 字元區間
參數：
-d  ：後面接分隔字元。與 -f 一起使用；
-f  ：依據 -d 的分隔字元將一段訊息分割成為數段，用 -f 取出第幾段的意思；
-c  ：以字元 (characters) 的單位取出固定字元區間；
範例：

範例一：將 PATH 變數取出，我要找出第五個路徑。
[root@linux ~]# echo $PATH
/bin:/usr/bin:/sbin:/usr/sbin:/usr/local/bin:/usr/X11R6/bin:/usr/games:
[root@linux ~]# echo $PATH | cut -d ':' -f 5
# 嘿嘿！如此一來，就會出現 /usr/local/bin 這個目錄名稱！
# 因為我們是以 : 作為分隔符號，第五個就是 /usr/local/bin 啊！
# 那麼如果想要列出第 3 與第 5 呢？，就是這樣：
[root@linux ~]# echo $PATH | cut -d ':' -f 3,5
```


擷取字元 grep 40

```
[root@linux ~]# grep [-acinv] '搜尋字串' filename
參數：
-a  ：將 binary 檔案以 text 檔案的方式搜尋資料
-c  ：計算找到 '搜尋字串' 的次數
-i  ：忽略大小寫的不同，所以大小寫視為相同
-n  ：順便輸出行號
-v  ：反向選擇，亦即顯示出沒有 '搜尋字串' 內容的那一行！
範例：

範例一：將 last 當中，有出現 root 的那一行就取出來；
[root@linux ~]# last | grep 'root'

範例二：與範例一相反，只要沒有 root 的就取出！
[root@linux ~]# last | grep -v 'root'

範例三：在 last 的輸出訊息中，只要有 root 就取出，並且僅取第一欄
[root@linux ~]# last | grep 'root' |cut -d ' ' -f1
# 在取出 root 之後，利用上個指令 cut 的處理，就能夠僅取得第一欄囉！
```


排序 sort 41

```
[root@linux ~]# sort [-fbMnrtuk] [file or stdin]
參數：
-f  ：忽略大小寫的差異，例如 A 與 a 視為編碼相同；
-b  ：忽略最前面的空白字元部分；
-M  ：以月份的名字來排序，例如 JAN, DEC 等等的排序方法；
-n  ：使用『純數字』進行排序(預設是以文字型態來排序的)；
-r  ：反向排序；
-u  ：就是 uniq ，相同的資料中，僅出現一行代表；
-t  ：分隔符號，預設是 tab 鍵；
-k  ：以那個區間 (field) 來進行排序的意思，
範例：

範例一：個人帳號都記錄在 /etc/passwd 下，請將帳號進行排序。
[root@linux ~]# cat /etc/passwd | sort
adm:x:3:4:adm:/var/adm:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
# 我省略很多的輸出～由上面的資料看起來， sort 是預設『以第一個』資料來排序，
# 而且預設是以『文字』型態來排序的喔！所以由 a 開始排到最後囉！
```


單一輸出uniq與字元計算wc 42

```
[root@linux ~]# uniq [-ic]
參數：
-i  ：忽略大小寫字元的不同；
-c  ：進行計數
範例：

範例一：使用 last 將帳號列出，僅取出帳號欄，進行排序後僅取出一位；
[root@linux ~]# last | cut -d ' ' -f1 | sort | uniq

範例二：承上題，如果我還想要知道每個人的登入總次數呢？
[root@linux ~]# last | cut -d ' ' -f1 | sort | uniq -c
```

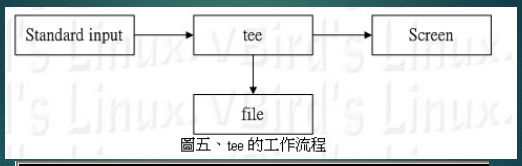
```
[root@linux ~]# wc [-lwm]
參數：
-l  ：僅列出行；
-w  ：僅列出多少字(英文單字)；
-m  ：多少字元；
範例：

範例一：那個 /etc/man.config 裡面到底有多少相關字、行、字元數？
[root@linux ~]# cat /etc/man.config | wc
     138     709    4506
# 輸出的三個數字中，分別代表： 『行、字數、字元數』
```

管線命令

雙重導向 tee

43



圖五、tee 的工作流程

```
[root@linux ~]# tee [-a] file
參數：
-a  ：以累加 (append) 的方式，將資料加入 file 當中！
範例：
[root@linux ~]# last | tee last.list | cut -d " " -f1
# 這個範例可以讓我們將 last 的輸出存一份到 last.list 檔案中；
[root@linux ~]# ls -l /home | tee ~/homefile | more
# 這個範例則是將 ls 的資料存一份到 ~/homefile ，同時螢幕也有輸出訊息！
[root@linux ~]# ls -l / | tee -a ~/homefile | more
```

管線命令

參數代換 xargs

44

```
[root@linux ~]# xargs [-0epn] command
參數：
-0  ：如果輸入的 stdin 含有特殊字元，例如 `, \, 空白鍵等等字元時，這個 -0 參數
      可以將他還原成一般字元。這個參數可以用於特殊狀態喔！
-e  ：這個是 EOF (end of file) 的意思。後面可以接一個字串，當 xargs 分析到
      這個字串時，就會停止繼續工作！
-p  ：在執行每個指令的 argument 時，都會詢問使用者的意思；
-n  ：後面接次數，每次 command 指令執行時，要使用幾個參數的意思。看範例三。
當 xargs 後面沒有接任何的指令時，預設是以 echo 來進行輸出喔！
範例：

範例一：將 /etc/passwd 內的第一欄取出，僅取三行，使用 finger 這個指令將每個
        帳號內容秀出來
[root@linux ~]# cut -d':' -f1 < /etc/passwd |head -n 3| xargs finger
```

- 讓無法支援管線命令的指令可以讀取STDOUT成為其參數(argument)

檔案/指令搜尋

指令的搜尋

46

- 判斷指令從何而來，包括內建指令的顯示：
  - type command
    - ex> type –a echo
- 從 PATH 當中搜尋實際指令檔案
  - which command

檔案/指令搜尋

由資料庫中搜尋檔案

47

- 檔名資料庫的建置
  - updatedb
  - 資料庫在：/var/lib/slocate
- 檔名關鍵字的搜尋
  - locate keyword
  - locate –r {正規表示法}
- man page 的搜尋
  - makewhatis →建立資料庫
  - whatis keyword →搜尋keyword是否有man page

檔案/指令搜尋

檔案/指令搜尋

直接找硬碟：find

48

- 指令語法：
  - find [目錄] [類型] [動作]
  - 範例：
    - 將過去系統上面 24 小時內有更動過內容 (mtime) 的檔案列出
      - [root@linux ~]# find / -mtime 0
    - 尋找 /etc 底下的檔案，如果檔案日期比 /etc/passwd 新就列出
      - [root@linux ~]# find /etc -newer /etc/passwd
    - 搜尋 /home 底下屬於 dmtsai 的檔案
      - [root@linux ~]# find /home -user dmtsai
    - 找出檔名為 passwd 這個檔案
      - [root@linux ~]# find / -name passwd
    - 找出系統中，大於 1MB 的檔案
      - [root@linux ~]# find / -size +1000k

正規表示法

工作管理(job control)

正規表示法 50

- 正規表示法的功能：
  - 透過一種符號表示的方法，來擷取所需要的資訊
  - 通常為『一行一行』處理
  - 常用的指令如：
    - grep
    - sed
    - awk

Job control 53

- 單一終端機工作介面
  - command &　　→在背景中『執行』
  - jobs　　→查看背景中的工作情況
  - fg %n　　→取出第n個工作到前景
  - bg %n　　→讓第n個工作在背景執行
  - kill %n　　→刪除第 n 個工作
  - [ctrl]-z　　→將前景的工作丟到背景暫停

正規表示法的字符 51

RE 字符	意義與範例
^word	待搜尋的字串(word)在行首！ 範例：grep -n '^#' regular_express.txt 搜尋行首為 # 開始的那一行！
word$	待搜尋的字串(word)在行尾！ 範例：grep -n '!$' regular_express.txt 將行尾為 ! 的那一行列印出來！
.	代表『任意一個』字符，一定是一個任意字符！ 範例：grep -n 'e.e' regular_express.txt 搜尋的字串可以是 (eve) (ese) (eee) (e e)，但不能僅有 (ee)！亦即 e 與 e 中間『一定』僅有一個字元，而空白字元也是字元！
\	跳脫字符，將特殊符號的特殊意義去除！ 範例：grep -n \' regular_express.txt 搜尋含有單引號 ' 的那一行！
*	重複零個或多個的前一個 RE 字符 範例：grep -n 'ess*' regular_express.txt 找出含有 (es) (ess) (esss) 等等的字串，注意，因為 * 可以是 0 個，所以 es 也是符合帶搜尋字串。另外，因為 * 為重複『前一個 RE 字符』的符號，因此，在 * 之前必須要緊接著一個 RE 字符喔！例如任意字元則為『.*』！

精選範例

55
- 請問在bash中，||和&&各代表什麼意思，例如 ls foo||cd /tmp; ls&& cd /tmp？ 複選 AB
  - (A)前面一個命令若執行成功則後面的命令就不執行；前面一個命令如果執行成功則後面的命令就執行
  - (B)前面一個命令若執行失敗則後面的命令就須執行；前面一個命令如果執行失敗則後面的命令就不執行
  - (C)前面一個命令若執行成功則後面的命令就須執行；前面一個命令如果執行成功則後面的命令就不執行
  - (D)前面一個命令若執行失敗則後面的命令就不執行；前面一個命令如果執行成功則後面的命令就不執行

58
- 以下關於『program1 | tee /tmp/logfile』的敘述，何者正確？A
  - (A) 所有程式執行結果都會被寫入 /tmp/logfile
  - (B) 所有程式執行結果將無法顯示於螢幕上
  - (C) 如果程式發生錯誤，所有錯誤訊息也會一併寫入 /tmp/logfile
  - (D) 本指令之作用，等同於 program | tee > /tmp/logfile
- 以下哪些指令可以列出/etc目錄下的所有檔案名稱？AB
  - (A) ls –lR /etc
  - (B) find /etc -print
  - (C) dump –R /etc
  - (D) search –l /etc

56
- Linux的 shell 操作環境中，支援下列哪些功能？複選ABD
  - (A) 提供 alias 別名設定功能
  - (B) 按<tab>鍵可以指令補齊
  - (C) 可用 ？號查詢指令用法，如 ls /?
  - (D) 提供 history 歷史命令功能，方便使用者操作
- 命令串『cat –n < test1 > test2 』是何意思？單選 D
  - (A) 將 test1 合併到 test2
  - (B) 將 test1 重導到 test2
  - (C) 將 test2 合併到 test1
  - (D) 將 test1 加入列號重導到 test2 去

59
- 當使用者 david 執行『 cd ‘~david’ 』會產生何種動作？C
  - (A) 切換目錄到 david 的 $HOME 目錄
  - (B) 切換目錄至名稱為 ‘~david’ 的目錄
  - (C) 切換目錄至名稱為 ~david 的目錄
  - (D) 切換目錄至名稱為 david 的目錄
- 下列哪依個命令行可以找出當前目錄內(不含子目錄)的符號連結檔？ A
  - (A) ls –l | grep ‘^l’ | awk ‘{print $NF}’
  - (B) ls –type l
  - (C) find –type l
  - (D) ls -ll

57
- 關於指令『 cat /etc/passwd | grep ‘vincent’ 』何者正確？B
  - (A) 列出 /etc/passwd 所有內容
  - (B) 只顯示 /etc/passwd 中有 vincent 字串的整列
  - (C) 列出 /etc/passwd 中 vincent 的字串有幾個
  - (D) 指 vincent 字串清掉，並顯示其部分
- 以下關於 su 指令的敘述，何者正確？複選 AB
  - (A) 用於切換使用者身份
  - (B) 如果沒有指定欲切換之使用者，預設切換成為 root
  - (C) 一旦切換成為其他使用者，若欲切換回原本的使用者，必須再次執行 su 指令進行切換
  - (D) 無論任何使用者，若欲切換至其他使用者，必須輸入欲切換者之密碼，方可順利切換

60
- 假設當前工作目錄有一個檔案，檔名為1，且其內容為2，請問下列那個命令會得到 2 這樣的輸出結果？ B
  - (A) /bin/false || echo $? | xargs ls
  - (B) /bin/false || echo $? | xargs cat
  - (C) /bin/false || xargs echo
  - (D) /bin/false | xargs cat
- 變數有兩種，一種是全域變數(Global)一種是自訂變數(Local)，請問下列何者正確？ ACD
  - (A) Global變數可被繼承於子程序中，而Local不行
  - (B) 想要顯示目前的 local 變數，可用 env
  - (C) 如果要讓 local 變數成為 global ，可用 export 指令
  - (D) 命令 set 可同時顯示 local 與 global 變數

61

- 關於『find / -name “java*” > msg1 2>msg2』的描述，下列何者正確？C
  - (A) 該指令會搜尋根目錄底下檔名以 java 為開頭的檔案，並將結果顯示於螢幕上
  - (B) 該指令的執行結果如了會顯示在螢幕上外，也會儲存在兩個檔案中
  - (C) 該指令正確執行結果會儲存在檔案 msg1 中，錯誤執行結果會儲存在檔案 msg2 中
  - (D) 在檔案 msg2 的內容可能會有 /home/peter/ibm_java字串

分享指引

64

- Linux 的帳號與群組
- 帳號管理與身份切換
- 程序與程序管理
- 程序的nice值功能
- 精選範例

62

- 在 bash 中組合按鍵 [ctrl]-z 代表什麼功能？A
  - (A) 暫停目前的命令
  - (B) 終止目前的命令
  - (C) 暫停螢幕的輸出
  - (D) 恢復螢幕的輸出

Linux的帳號與群組

經濟部資訊專業人員鑑定—開放式系統類

Linux 基礎運作—帳號與程序管理

崑山科技大學資訊傳播系
蔡德明
(鳥哥, VBIRD)

帳號功能

66

- 登入識別碼
  - 人類容易記帳號，但作業系統記得則是識別碼(ID)
  - 將使用者ID(UID)與群組ID(GID)達成對應
    - /etc/passwd → 記錄 UID
    - /etc/group → 記錄 GID

UID範圍

67

- 使用者ID的意義
  - 0 →系統管理員使用的UID
  - 非為零 →一般帳號
    - 1-499 →保留給系統用的
      - 1-99 →給系統本身用的帳號
      - 100-499 →給一些網路服務用的帳號
    - 500- →讓可登入者使用的ID

群組功能

70

- 群組設定檔 /etc/group，共分四個欄位
  - 群組名稱
  - 群組密碼(移動至 /etc/gshadow)
  - GID
  - 加入此群組的使用者帳號
- 有效群組與初始群組(initial group)
  - 有效群組：新建檔案時，該檔案的所屬群組
  - 初始群組：寫在/etc/passwd內的GID
- 有效群組的切換
  - 使用 newgrp [group_name]
  - 離開 newgrp 用 exit 即可。

帳號記錄檔

68

- 帳號的參數檔案分別為：
  - /etc/passwd →包括家目錄、UID、shell等
  - /etc/shadow →密碼相關資訊
- /etc/passwd 共有七的欄位，以『:』分開
  - 帳號名稱
  - 密碼(已經挪至/etc/shadow)
  - UID
  - GID
  - 全名(這個帳號的說明)
  - 家目錄
  - shell

帳號管理與身份切換

密碼/etc/shadow格式

69

- /etc/shadow共分為九個欄位，第一欄位需與/etc/passwd相對應
  - 帳號名稱，需與/etc/passwd 對應
  - 密碼，加密過的資料，一般使用md5加密機制
  - 最近密碼更動過的日期，使用1970/1/1累積的日數
  - 密碼不可被更動的日期：從密碼最近被更動的日期到下次能夠變更的日期
  - 密碼需重新變更的日期：多久需要重新設定一次密碼
  - 密碼變更前的警告日期：密碼需變更前的幾日發出警告
  - 密碼過期寬恕時間：過期後還能使用的日期
  - 帳號失效日：無論如何，密碼到這個設定值就會失效
  - 保留

新增使用者

72

```
[root@linux ~]# useradd [-u UID] [-g initial_group] [-G other_group] \
>  -[Mm] [-c 說明欄] [-d home] [-s shell] username
參數：
-u  ：後面接的是 UID ，是一組數字。直接指定一個特定的 UID 給這個帳號；
-g  ：後面接的那個群組名稱就是我們上面提到的 initial group 啦～
      該 group ID (GID) 會被放置到 /etc/passwd 的第四個欄位內。
-G  ：後面接的群組名稱則是這個帳號還可以支援的群組。
      這個參數會修改 /etc/group 內的相關資料喔！
-M  ：強制！不要建立使用者家目錄
-m  ：強制！要建立使用者家目錄！
-c  ：這個就是 /etc/passwd 的第五欄的說明內容啦～可以隨便我們設定的啦～
-d  ：指定某個目錄成為家目錄，而不要使用預設值；
-r  ：建立一個系統的帳號，這個帳號的 UID 會有限制 (/etc/login.defs)
-s  ：後面接一個 shell ，預設是 /bin/bash 的啦～
```

- 建立系統帳號 → useradd –r account
- 建立不可登入的帳號 → useradd –s /sbin/nologin account

帳號管理與身份切換

新增使用者的參考資訊 73

- /etc/default/useradd
  - 指定shell, 主要家目錄, 初始群組及家目錄參考來源目錄之資料
- /etc/skel/*
  - 預設的家目錄參考來源目錄，可由 /etc/default/useradd 修改
- /etc/login.defs
  - 最大及最小的UID/GID設定
  - 密碼相關資訊的指定

帳號管理與身份切換

密碼設定 74

- 密碼的設定與修改
  - 密碼設定：以root身份『passwd account』
  - 密碼修改：使用者自己下達『passwd』
- 密碼參數的查閱
  - chage –l
- 強制使用者初次登入需修改密碼的方法
  - chage –d 0 account

帳號管理與身份切換

使用者參數修改/刪除 75

- 需以root身份處理
  - 刪除使用者用 userdel [-r] username

```
[root@linux ~]# usermod [-cdegGlsuLU] username
參數：
-c  ：後面接帳號的說明，即 /etc/passwd 第五欄的說明欄，可以加入一些帳號的說明。
-d  ：後面接帳號的家目錄，即修改 /etc/passwd 的第六欄；
-e  ：後面接日期，格式是 YYYY-MM-DD 也就是在 /etc/shadow 內的第八個欄位資料啦！
-g  ：後面接 group name，修改 /etc/passwd 的第四個欄位，亦即是 GID 的欄位！
-G  ：後面接 group name，修改這個使用者能夠支援的群組，修改的是 /etc/group 喔~
-l  ：後面接帳號名稱。亦即是修改帳號名稱， /etc/passwd 的第一欄！
-s  ：後面接 Shell 的實際檔案，例如 /bin/bash 或 /bin/csh 等等。
-u  ：後面接 UID 數字啦！即 /etc/passwd 第三欄的資料；
-L  ：暫時將使用者的密碼凍結，讓他無法登入。其實僅改 /etc/shadow 的密碼欄。
-U  ：將 /etc/shadow 密碼欄的 ！ 拿掉，解凍啦！
```

帳號管理與身份切換

群組處理 76

- 群組的建立
  - groupadd [-g gid] group_name
- 群組的修訂
  - groupmod [-g gid] [-n name] group_name
- 群組的刪除
  - groupdel group_name
  - 此群組不可是某帳號的初始群組，才能被刪除

帳號管理與身份切換

使用者身份參數自我修改 77

- 常見可修改/觀察的指令
  - 修改 shell 的方法：
    - chsh –s shell
  - 查閱/修改使用者的註解說明
    - finger username
    - chfn
  - 查閱使用者的UID/GID
    - id [username]
  - 查閱使用者支援的群組
    - groups

帳號管理與身份切換

用 su 切換身份 78

- 使用 su 切換身份須知
  - 需要知道被切換者的密碼
  - root可切換成為任何人，不需原使用者密碼
  - 最好加上 – 才能夠讀取使用者的環境設定檔
    - su – [username]
  - 在多人共管的環境中，可能有很多人會知道root的密碼，理論上是比較不安全些。

帳號管理與身份切換

使用sudo操作系統指令 79

- 可用sudo來操作系統指令，語法如下：
  - sudo [-u username] command
- 可使用 sudo 的使用者需規範於 /etc/sudoers
  - /etc/sudoers 不可直接編輯
  - 使用visudo可編輯/etc/sudoers

程序與程序管理

程序與程序管理

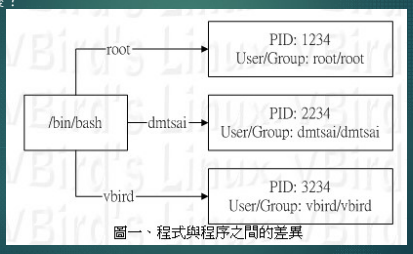
程式與程序 81

- 程式 (program)
  - 通常為binary program，是作業系統可以執行的檔案資料，如/bin/ls
  - 程式通常放置於硬碟/光碟/軟碟等儲存設備
- 程序 (process)
  - program被系統觸發後，其程式碼會被載入記憶體中，同時會載入該程式所需的資料，及執行者的相關屬性/權限資訊
  - 系統會給予該段記憶區段一個識別碼，稱為PID
  - 一個program可被同時觸發多次，彼此間不會互相干擾，因為PID並不相同。

程序與程序管理

程序執行的範例 82

- 問：為何大家登入都是使用/bin/bash，卻不會互相干擾？



圖一、程式與程序之間的差異

程序與程序管理

程序的相關性 83

- 系統執行的第一支程序：init (/sbin/init)
  - 此程序的PID為 1
  - 系統所有的其他程序都由此 PID 所衍生
- 子程序與父程序
  - 被衍生的程序成為子程序
  - 子程序會再含有一個 PPID，代表父程序的PID
  - 當父程序結束時，所有的子程序亦將被結束

程序與程序管理

程序的管理 84

- 使用 ps 查閱
  - ps –l
    - 只看bash自己的程序
  - ps aux
    - 看所有的程序，包含背景中的各項程序資料
- USER：屬於那個使用者帳號
- PID ：程序的號碼。
- %CPU：使用掉CPU資源百分比
- %MEM：所佔用記憶體百分比
- TTY ：是在那個終端機上面運作
- STAT：該程序目前的狀態，主要的狀態
  - R ：目前正在運作
  - S ：目前正在睡眠當中
  - T：目前正在偵測或是停止
  - Z ：殭屍程序
- START：被觸發啟動的時間
- TIME：實際使用CPU運作時間
- COMMAND：實際指令

程序與程序管理

動態觀察程序

85

- 使用 top 嘿

```
[root@linux ~]# top [-d] | top [-bnp]
參數：
-d  ：後面可以接秒數，就是整個程序畫面更新的秒數。預設是 5 秒；
-b  ：以批次的方式執行 top ，還有更多的參數可以使用喔！
      通常會搭配資料流重導向來將批次的結果輸出成為檔案。
-n  ：與 -b 搭配，意義是，需要進行幾次 top 的輸出結果。
-p  ：指定某些個 PID 來進行觀察監測而已。
在 top 執行過程當中可以使用的按鍵指令：
      ? ：顯示在 top 當中可以輸入的按鍵指令；
      P ：以 CPU 的使用資源排序顯示；
      M ：以 Memory 的使用資源排序顯示；
      N ：以 PID 來排序喔！
      T ：由該 Process 使用的 CPU 時間累積 (TIME+) 排序。
      k ：給予某個 PID 一個訊號 (signal)
      r ：給予某個 PID 重新制訂一個 nice 值。
```

程序與程序管理

程序樹相關性與程序的刪除

86

- pstree [-pu]
- 程序的刪除
  - 給予程序一個訊號(signal)，常見的訊號
    - kill –l
    - man 7 signal
      - 1) SIGHUP 重新讀取設定檔(reload)
      - 9) SIGKILL 強制將某程序從記憶體中移除
      - 15) SIGTERM 嘗試以正常流程將程序關閉
    - kill -9 12345
  - 給予某指令一個訊號
    - killall -9 command

程序的nice值功能

程式的執行順序

88

- ps –l

```
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S   500  4663  4662  0  76   0 -  1351 wait   pts/0    00:00:00 bash
0 R   500  4788  4663  0  78   0 -  1244 -      pts/0    00:00:00 ps
```

- PRI (Priority, 優先執行順序)
  - 越小越早被CPU所執行
  - 為核心動態調整，不會一直是固定的
- NI (Nice, 可修改 PRI)
  - PRI(new) = RPI(old) + Nice
  - Nice越小可讓PRI變小，所以越快被CPU所執行
  - Nice 範圍為 -20 ~ 19
  - nice 只有 root 可以設定為負值
  - 使用者只能將nice越調越高，且只能調整自己的PID

程序的nice值功能

nice值的使用

89

- 新執行的指令，使用 nice
  - nice –n NI command
    - ex> nice –n 10 bash
- 從已經存在的PID修改其nice值
  - renice NI PID
    - ex> renice -5 12345
  - top
    - 按下 r 即可選擇 PID 與 Nice 值了

程序的nice值功能

精選範例

91

- 想產生一個不用登入密碼的帳號 guset ，何者正確？ BC
  - (A) 輸入 useradd guest 即可免密碼登入
  - (B) 輸入 useradd –p ‘’ guest
  - (C) 輸入 useradd guest; passwd –d guest
  - (D) 輸入 useradd guest; passwd –p ‘’ guest
- 下列何者可使使用者 foo 無法登入系統？ ABC
  - (A) useradd –s /sbin/nologin foo
  - (B) usermod –s /bin/true foo
  - (C) usermod –s /bin/false foo
  - (D) useradd –s /sbin/tcsh foo

94

- 有關 Linux 的程序觀念，下列哪些正確？ ABC
  - (A) 所有程序都是由 init 這一個程序分出來的，或再分出來的
  - (B) 子程序都是由父程序分出來的
  - (C) 程序就是只執行中的程式
  - (D) 程序和程序之間完全獨立，無法互相溝通

92

- 若要把某個檔案的權限改為 Owner 可讀寫，其餘可讀，應如何處理？ B
  - (A) chmod 755
  - (B) chmod 644
  - (C) chmod 722
  - (D) chmod 622
- 關於記錄使用者帳號的 /etc/passwd 下列敘述何者有誤？C
  - (A) 該檔案裡面一行代表一個使用者帳號
  - (B) 裡面記載了某一帳號登入後要使用哪一個shell
  - (C) 該檔案也記載了帳號的失效日期
  - (D) 該檔案中不會有兩個相同的使用者別碼 (UID)

經濟部資訊專業人員鑑定—開放式系統類

Linux進階系統管理
帳號與權限管理、程序管理

昆山科技大學資訊傳播系
蔡德明
(鳥哥, VBIRD)

93

- 關於程序的描述下列何者有誤？A
  - (A) 如果A程序的PID與B程序的PPID相同，則A程式是B程序的子程序
  - (B) 使用pstree可以察看每個程序之間的相關性
  - (C) 系統中執行程序所擁有的權限與該程序的執行者權限有關
  - (D) 每一個系統中執行的程序都會有一個PID，而且這個PID不會重複
- 關於SUID(SetUID)的描述下列何者有誤？B
  - (A) SUID的權限設定只對二進位檔案有效
  - (B) 一般使用者執行passwd指令來修改密碼時，passwd程序的權限就是執行者的權限
  - (C) 使用者執行具有SUID的檔案時，該檔案執行中的權限與該檔案的擁有者的權限相同
  - (D) 使用find /home –perm -4000 可查詢具有 SUID 權限的檔案

96

分享指引

- 帳號管理
- 權限管理
- BASH的工作管理
- 程序管理
- 精選範例

帳號管理

帳號/群組相關設定檔

- 帳號相關設定檔
  - /etc/passwd　使用者帳號參數
  - /etc/shadow　使用者密碼相關參數
  - /etc/login.defs　密碼、UID、GID等限制參數
  - /etc/default/useradd 新建使用者參考檔案
  - /etc/skel/*　預設家目錄參考檔
- 群組相關設定檔
  - /etc/group　群組與GID還有支援的用戶
  - /etc/gshadow　群組的密碼(少用)
- 登入相關設定檔
  - /etc/shells　使用者可使用的合法 shell
  - /etc/nologin　維護時所使用，對root無效。

帳號/群組相關工具指令

- 與使用者新增/修改/刪除有關之指令
  - useradd　新增用戶
  - usermod　修改用戶的參數
  - userdel　刪除用戶
  - chsh　修改使用者的 shell
  - chfn　修改finger的內容
  - passwd　修改密碼
- 觀察使用者的相關指令
  - finger　顯示使用者的家目錄/shell/計畫檔等資料
  - id　顯示使用者的UID、GID等資料
  - groups　顯示使用者加入的群組
  - w, who　顯示目前在線上的使用者
  - last　顯示過去有登入系統的帳號資訊
  - chage　顯示/修改使用者密碼參數

帳號/群組切換的方法

- 帳號變換的指令
  - su –
    - 一般用戶需要知道root/user的密碼才能變更
    - root不需要密碼可隨意變更
  - sudo command
    - 需透過/etc/sudoers的設定決定用戶是否能使用sudo
    - 需使用visudo去修改/etc/sudoers
    - 只需要輸入用戶自己的密碼即可。
- 有效群組的切換
  - newgrp 新群組
    - 有效群組在新建檔案/目錄時有作用

bash與使用者有關的變數

- 在bash的環境變數中：
  - HOME　代表使用者家目錄，即 ~
  - HISTSIZE　記錄歷史命令的筆數
  - LANG　語系資料，可觀察locale –a
  - PATH　與指令的搜尋順序有關
  - MAIL　使用者的郵件信箱
  - PS1　使用者的命令提示字元
- 與bash環境有關的設定檔(主要的)
  - /etc/profile 整體bash所必須讀取的login-shell
  - ~/.bashrc　自己可以設定的non-login-shell

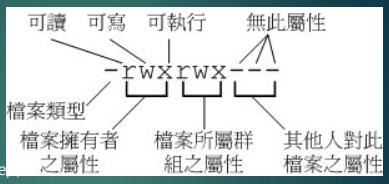
使用者工作排程相關

- 使用者可使用的工作排程
  - at time
    - 單一次的工作，可使用atq, atrm管理
  - crontab –e
    - 循環型的，五個參數『分 時 日 月 週』
  - batch time
    - 與 at 類似，該工作會在 CPU loading 小於 0.8 時才會運作
  - nohup command
    - 讓 command 這個指令在使用者登出後可以持續運作，與工作管理並不相同喔！

權限管理

檔案的權限

- 檔案類型
  - [d]是目錄
  - [-]是檔案
  - [l]為連結檔(link file
  - [b]為裝置檔裡面的可供儲存的周邊設備；
  - [c]則表示為裝置檔裡面的序列埠設備



檔案/目錄權限的意義

- 檔案與目錄
  - 檔案可記錄實際的資料
  - 目錄記錄的是『該目錄下的檔名』
- 一般檔案
  - r　可讀取該檔案的內容
  - w　可修改/編輯檔案的內容
  - x　可執行該檔案
- 目錄檔案
  - r　可讀取目錄內記錄的檔名
  - w　可在該目錄內建立/刪除檔名
  - x　可進入該目錄

權限的應用

- 進入某目錄成為『可工作目錄』的基本權限
  - 使用者可進入該目錄需具備 x 權限
  - 使用者可在該目錄查閱檔名，需具備 r 權限
- 讀取一個檔案的基本權限
  - 使用者在該檔案所在的目錄至少要有 x 權限
  - 使用者對該檔案至少要有 r 權限
- 修改一個檔案的基本權限
  - 使用者在該檔案所在的目錄至少要有 x 權限
  - 使用者對該檔案至少要有 r, w 權限

權限的應用(續)

- 建立一個檔案的基本權限
  - 使用者在該目錄要具有 w,x 的權限
- 進入某目錄並執行該目錄下的某個指令
  - 使用者在該目錄至少要有 x 的權限
  - 使用者在該檔案至少需要有 x 的權限
- 思考
  - 對於 cp 來說，來源/目標的權限各是如何？

特殊屬性與旗標

- 特殊屬性的建立/顯示方法(ext2/ext3支援)
  - chattr [+-=][ia] file
  - lsattr file
- 特殊旗標的意義
  - rwsrwsrwt
    - 第一個s　SUID，二進位程式，run-time，owner
    - 第二個s　SGID，目錄，有效群組
    - 第三個t　SBIT，只有自己與root可刪除自己的file
- 預設權限
  - umask

BASH的工作管理

程序管理

觀察程序的重要指令

112

- 靜態觀察
  - ps –l　　僅查詢個人有關的bash相關程序
  - ps aux　　查詢整個系統的程序資訊
  - pstree [-pu]以樹狀關係圖查詢程序相關性
- 動態觀察
  - 使用X Window底下的圖形介面，ex>gnome-system-monitor
  - top [-d sec] 預設五秒鐘更新一次
- 系統資源觀察
  - free
  - uptime
  - sar, vmstat　需安裝額外的套件才能有sar指令

BASH的工作管理

job control

110

- 單一終端機工作介面
  - command &　　→在背景中『執行』
  - jobs　　→查看背景中的工作情況
  - fg %n　　→取出第n個工作到前景
  - bg %n　　→讓第n個工作在背景執行
  - kill %n　　→刪除第 n 個工作
  - [ctrl]-z　　→將前景的工作丟到背景暫停

程序管理

程序的優先執行緒(Priority)

113

- 關於nice值
  - 可更改PRI數值
  - 範圍界於 -20~19之間
  - 只有root可以調整成為負值
  - 一般使用者只能調整自己的PID且僅能越調越高
- 常用指令
  - nice –n [nice] command
  - renice [nice] PID

程序管理

程序管理

程序的控制

114

- 程序之間的管理
  - 可以透過給予PID一個signal來管理
  - 常見的幾個signal為
    - 1　SIGHUP
    - 9　SIGKILL
    - 15　　SIGTERM (default)
- 常用的指令
  - kill –N PID
  - killall –N command

精選範例

115

- 在Linux檔案系統中，刪除一個檔案最少具備什麼權限？A
  - (A) 只需對檔案所在目錄具有 wx
  - (B) 只需對檔案所在目錄具有 wx，及對檔案具有 w
  - (C) 只需對檔案所在目錄具有 wx，及對檔案具有 rw
  - (D) 只需對檔案所在目錄具有 w ，及對檔案具有 w
- 在/etc/shadow中，如『adm:*:11979:0:99999:7:::』第六欄代表？D
  - (A) 密碼需變更的天數
  - (B) 帳號失效日期
  - (C) 帳號取消日期
  - (D) 密碼變更期限前之警告期限

116

- 請問如何把程式foo放到背景執行？（複選）AD
  - (A) foo &
  - (B) foo %
  - (C) foo !
  - (D) foo然後[Ctrl]+z然後bg
- 我的系統現在是處於X Window環境中，這表示我現在正執行runlevel 5嗎？D
  - (A) 是的！因為只有 runlevel 5才會啟動X Window
  - (B) 不一定，得查察看/etc/inittab檔案中的『initdefault』那一行
  - (C) 是的，因為我執行runlevel指令時，指令輸出為 5 3
  - (D) 不是，因為我執行runlevel指令時，指令輸出為 N 3

117

- 請問下列哪一個bash SHELL VARIABLE代表所需記憶指令記錄(command history)的數目？C
  - (A) HISTFILE
  - (B) HISTFILESIZE
  - (C) HISTSIZE
  - (D) HISTCONTROL
- Linux在何種狀態下必須要重新開機使新的設定啟動？A
  - (A) 更新kernel
  - (B) 更改 IP 位址
  - (C) 啟動使用者或群組的 quota
  - (D) 安裝新的應用程式

118

- 使用者foo屬於群組foo，有一個目錄名稱為foo，使於使用者root及群組root，並且目錄foo的存取權限為『drwxr-x--x』請問下列哪些正確？AD
  - (A) 使用者foo可以進入目錄foo
  - (B) 使用者foo不能進入目錄foo
  - (C) 使用者foo可以由ls看到目錄foo下有哪些檔案
  - (D) 如果目錄foo中有一檔案bar其存取權限為777，使用者foo可以讀取bar
- 當一個使用者bill執行下面這個指令時，所謂的effective UID為何『r-sr-x--x joe sales /usr/local/bin/program』B
  - (A) bill
  - (B) joe
  - (C) sales
  - (D) root

120

- 請問『chmod 3757 foo』，foo 為一個目錄，其權限為？B
  - (A) drwtr-srws
  - (B) drwxr-srwt
  - (C) drwsr-Srwt
  - (D) drwsr-srwx
- 使用jobs命令顯示背景執行工作，並列出執行的PID，下列何者正確？B
  - (A) jobs -t
  - (B) jobs -l
  - (C) jobs -i
  - (D) jobs -s

121

- 使用 nohup 命令敘述，何者最正確？B
  - (A) 讓程序執行於背景中
  - (B) 使用者登出系統後，繼續於背景中執行該程序
  - (C) 使用者登出系統後，結束執行該程序
  - (D) 使用者登出系統後，該執行程序會暫停
- 藉由以下哪一種訊號(signal)可以促使某一程序重新啟動？D
  - (A) SIGKILL
  - (B) SIGRESTART
  - (C) SIGTERM
  - (D) SIGHUP

122

- 以下關於kill指令的敘述何者不正確？B
  - (A) kill -9 11251 代表傳送signal value 9 給PID 11251
  - (B) 如果沒有指定特別的signal，當執行kill 11251時，系統會預設傳送signal 1給 PID 11251
  - (C) 所有signal的定義皆可透過 man signal查詢
  - (D) kill 指令的真正含意為傳送 signal給某一正在執行的程序
- 以下關於程序執行的優先權(priority and nice)的敘述，何者不正確？D
  - (A) 一個程序的nice值越高，代表該程序的優先權越低
  - (B) 一個程序的priority越高，代表該程序的優先權越低
  - (C) 程序的擁有者可以自行提升nice值
  - (D) 程序的擁有者可以自行提升priority值

123

- 關於系統啟動init程式的敘述，何者正確？複選 ABCD
  - (A) init是系統啟動的第一支程序
  - (B) init 程式的PID絕對固定為 1
  - (C) 系統所有的其他程序都是由 init fork 出來的
  - (D) 控制 init 程式行為的設定檔為 /etc/inittab
- 關於程序優先權的說明，如下敘述哪些是正確的？ABD
  - (A) NI值越高優先權越低，反之則越高
  - (B) PRI值由kernel動態調整，但NI值需要使用者或管理員調整
  - (C) NI的取值範圍 -19 到 20 ，若不指定，欲設為 0
  - (D) 管理員可以用一個單一的命令 (renice) 將某一使用者的所有程序之NI值作重新的安排

124

- 某一使用者在早上時間安排了一個 at 排程工作『at 2pm』其內容為『echo meeting』但是時間到了卻沒有在螢幕上看到這個提示，請問原因為何？B
  - (A) 因為結果存到其家目錄的 at.out，而非傳輸至螢幕
  - (B) 因為結果會送到其信箱而非輸出至螢幕
  - (C) 因為 at 的結果只會輸出至管理員的螢幕
  - (D) 上面那行 at 命令格式是不正確的。
- 使用者 foo 想於每個禮拜天的上午一點到三點，和下午九點到十一點期間，每20分鐘執行程式bar，請問下列哪些為正確的cron設定？AC
  - (A) 0,20,40 1-3,21-23 * * 7 bar
  - (B) 0,20,40 1-3;21-23 * * 7 bar
  - (C) */20 1-3,21-23 * * 0 bar
  - (D) 0,20,40 1-3:21-23 * * 7 bar

125

- 使用者foo想將程序編號168由優先權0改為-10，請問下列何者正確？C
  - (A) nice –p 168 –n -10
  - (B) renice -10 168
  - (C) 使用者 foo 不能完成該變更
  - (D) renice –n -10 168
- 例行檢查主記憶體使用狀態可使用哪種方式？複選 BCD
  - (A) df -h
  - (B) free
  - (C) top
  - (D) cat /proc/meminfo

126

- 使用 dmesg 指令可以查閱觀看啥訊息？C
  - (A) 郵件伺服器的錯誤訊息
  - (B) 系統產生的錯誤訊息
  - (C) 開機時的訊息
  - (D) 使用者遠端登入的訊息
- 下列哪幾個工具程式可以用來調整程序的優先權(Priority)？AD
  - (A) top
  - (B) ps
  - (C) pstree
  - (D) renice

127

- 請問 fg – 的結果為何？
  - (A) 將最近第一個放入背景的程序放到前景來
  - (B) 將負載最小的背景程序放到前景
  - (C) 將名為『 - 』的背景執行程序放到前景
  - (D) 將最近第二個放入背景的程序放到前景來

- (A)
- (B)
- (C)
- (D)

BASH複習

經濟部資訊專業人員鑑定—開放式系統類

Linux進階系統管理
Shell scripts

昆山科技大學資訊傳播系
蔡德明
(鳥哥, VBIRD)

131

萬用字元的支援

- bash 的萬用字元功能：
  - * 0~無窮多個任意字元
  - ? 一定有一個任意字元
  - [0-9] 有一個在中刮號內的字元存在
  - [^abc] 有一個不在中刮號內的字元存在
  - 可 man grep 搜尋 \[: 找出特殊字浮功能
    - [:upper:], [:lower:]...
    - 可略過語系的問題

分享指引

129

- BASH複習
- shell script
- 精選範例

132

Bash的變數

- 變數的設定方式：
  - 變數名稱= “變數內容”
- 變數設定規則
  - 變數與變數內容以等號『=』連結，且等號兩邊不能直接接空白字元
  - 變數名稱只能是英文字母與數字，且**數字不能是開頭字元**；
  - 可以使用雙引號『 “ 』或單引號『 ‘ 』來將變數內容結合起來
    - 雙引號內的特殊字元可以保有變數特性，
    - 單引號內的特殊字元則僅為一般字元；
  - 跳脫字元『 \ 』來可特殊符號 變成一般字元；
  - 指令內的指令可用『 `command`』或『$(command)』
  - 可以 export 來使變數變成環境變數， 如『export PATH』；
  - 取消變數的方法為：『unset 變數名稱』。

BASH複習

變數的呼叫/使用

133

- 變數的呼叫：
  - echo $var
  - echo ${var}
- 變數的使用：
  - mkdir '~dmtsai' →建立名為 ~dmtsai 的目錄
  - echo "$PATH" →叫出PATH變數的內容
  - kversion=$(uname –r) →設定kversion為核心版本
  - echo "\$PATH" →顯示 $PATH 在螢幕上
  - set →顯示目前所有的變數

BASH複習

指令執行的順序

136

- 指令執行的順序：
  - 絕對/相對路徑直接執行指令
  - alias 所建立的別名
  - bash 內建的指令
  - PATH 所找到的指令
- 控制指令執行的細項：
  - cmd1 ; cmd2 兩者間沒有關係，依序執行
  - cmd1 && cmd2 cmd1成功，cmd2才會執行
  - cmd1 || cmd2 cmd1失敗，cmd2才會執行

BASH複習

亂數產生器

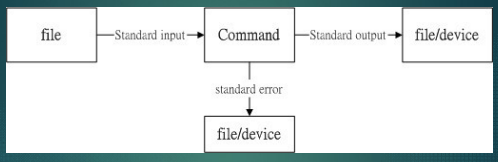
134

- BASH當中的特殊變數
  - RANDOM
    - 值介於0 ~ 32767
  - 隨機產生 0-100的亂數
    - echo $(($RANDOM*100/32727))
- 亂數產生器
  - /dev/random
  - /dev/urandom

BASH複習

資料流重導向

137



STDOUT： >(覆蓋) >>(累加)
STDERR： 2>(覆蓋) 2>>(累加)
全部的輸出： &> command > something 2>&1

BASH複習

命令別名與歷史命令

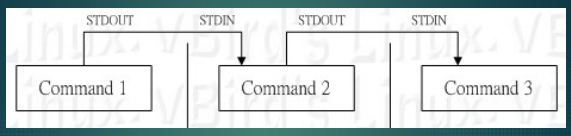
135

- 命令別名
  - 透過 alias newcmd= 'oldcmd options'
  - 取消使用 unalias newcmd
  - 可載入 ~/.bashrc 讓每次都生效
- 歷史命令
  - history
  - 透過 !n, !!, !cmd 等方式重複執行某指令
  - HISTSIZE可控制筆數
  - ~/.bash_history為記錄的檔案

BASH複習

管線命令

138



- 前一個指令的 STDOUT 作為後一個指令的 STDIN 之意。(STDERR沒有作用)

BASH複習

正規表示法

139

- 常用的正規表示法(基礎正規表示法)：利用一些字元符號(字符)來作為關鍵字的一種表示方式
  - ^ ：一行的行首
  - $ ：一行的行尾
  - [a-b]：一定有一個，視為集合
  - [^a-b] ：反向集合
  - * ：重複前一個 0 ~ 無窮多次
  - . ：一定有一個任意字元
  - .* ：0~無窮多個任意字元。

Shell scripts

BASH複習

利用test判斷

140

- 關於檔名的『類型』偵測(存在與否)，如 test -e filename
  - -e 該『檔名』是否存在？
  - -f 該『檔名』是否為檔案(file)？
  - -d 該『檔名』是否為目錄(directory)？
- 關於檔案的權限偵測，如 test -r filename
  - -r 偵測該檔名是否具有『可讀』的屬性？
  - -w 偵測該檔名是否具有『可寫』的屬性？
  - -x 偵測該檔名是否具有『可執行』的屬性？
- 兩個檔案之間的比較，如： test file1 -nt file2
  - -nt (newer than)判斷 file1 是否比 file2 新
  - -ot (older than)判斷 file1 是否比 file2 舊

Shell script

Shell script

143

- 什麼是 shell script
  - 將許多指令寫入一個批次檔，藉此快速執行
  - 透過 shell 的語法，可以設定程式 (if, then...)
  - 就是純文字檔囉
  - 需具有 r 或者是 r+x 才能夠被執行
- 為何需要學習 shell script
  - 很多的指令可以整合在一起，然後讓系統自動執行
  - 可以撰寫讓系統自動管理及自動進行錯誤偵測(troubleshooting)的任務
  - 可以用之以建立簡單的應用程式(如鳥哥自己的 logfile.sh 程式)

BASH複習

利用test判斷(續)

141

- 關於兩個整數之間的判定，例如 test n1 -eq n2
  - -eq 兩數值相等 (equal)
  - -ne 兩數值不等 (not equal)
  - -gt n1 大於 n2 (greater than)
  - -lt n1 小於 n2 (less than)
  - -ge n1 大於等於 n2 (greater than or equal)
  - -le n1 小於等於 n2 (less than or equal)
- 判定字串的資料
  - test -z string 字串是否為空, 若 string 為空,則 true
  - test -n string 字串是否非為空,若 string 為空,則 false。

Shell script

撰寫習慣與執行

144

- 撰寫習慣之建立：
  - 第一行請宣告 shell ，例如 #!/bin/bash
  - 第二行說明程式功能、授權、作者、此程式的版本等等
  - 建議常使用變數來處理複雜的資料
  - 常用『 #註解內容』來說明比較難懂得程式碼部分。
- 如何執行？
  - /root/script1.sh 需具有 r 與 x
  - cd /root; ./script1.sh 需具有 r 與 x
  - sh /root/script1.sh 具有 r 即可
  - 將 script1.sh 放入 $PATH 中 需具有 r 與 x

Shell script

145

第一支script

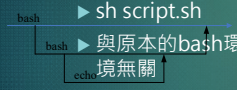
- 第一支shell script
  - #!/bin/bash
  - # This script displays some information about your system
  - # write by VBird 2007/11/17
  - echo “Hello world!”
  - echo “The day is $(date)”
  - echo “Your working directory is: $(pwd)”
  - exit 0

Shell script

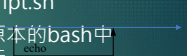
146

執行方式與結果

- 新增bash程序來處理
  - ./script.sh
  - sh script.sh
  - 與原本的bash環境無關



- 原bash環境下處理
  - source script.sh
  - . script.sh
  - 在原本的bash中執行



Shell script

147

鍵盤直接輸入變數 read

- 作中學一：利用 read 這個指令，讓使用者連續輸入first name與last name，最終組合起來顯示出來
  - #!/bin/bash
  - **read** -p "Please input your first name: " firstname
  - read -p "Please input your last name: " lastname
  - echo -e "\nYour full name is: $firstname $lastname"

Shell script

148

外加指令$(cmd)與`cmd`的功能

- 作中學二：透過 date 來產生與日期有關的檔案。
  - #!/bin/bash
  - echo -e "I will use 'touch' command to create 3 files."
  - read -p "Please input the filename what you want: " fileuser
  - date1=`date --date='2 days ago' +%Y%m%d`
  - date2=`date --date='1 days ago' +%Y%m%d`
  - **date3=`date +%Y%m%d`**
  - file1="$filename""$date1"
  - file2="$filename""$date2"
  - file3="$filename""$date3"
  - touch {$file1,$file2,$file3}

Shell script

149

數學運算功能

- 作中學三：讓使用者自己輸入數字，好進行乘法！
  - #!/bin/bash
  - echo -e "You SHOULD input 2 number, I will cross them! \n"
  - read -p "first number: " firstnu
  - read -p "second number: " secnu
  - total=**$(($firstnu*$secnu))**
  - echo -e "\nThe number $firstnu x $secnu is ==> $total"

Shell script

150

條件判斷 if...then...fi

- 作中學四：利用if...then的條件判斷式來處理重要任務。例如判斷使用者輸入的是 yes 或 no：
  - #!/bin/bash
  - read -p "Please input (Y/N): " yn
  - **if** ["$yn" == "Y" -o "$yn" == "y"]; **then**
  - echo "OK, continue"
  - exit 0
  - **elif** ["$yn" == "N" -o "$yn" == "n"]; **then**
  - echo "Oh, interrupt!"
  - exit 0
  - **else**
  - echo "I don't know what is your choice" && exit 0
  - **fi**

Shell script

151

script的內建變數

- Shell script的內建變數(數字表示)：
  - /path/to/scriptname opt1 opt2 opt3 opt4 . .
  - $0 $1 $2 $3 $4 . .
    - $* ：代表所有的後續參數，亦即『$1 $2 $3 $4...』
    - $@ ：代表所有參數總和，亦即『 "$1 $2 $3 $4.." 』
    - $# ：代表總共有幾個參數，例如 3 個或 4 個等等。
    - 可被 shift 這個指令挪動 opt 的號碼
      - ex> shift 2 → $1=opt3 (原本 $1=opt1)

Shell script

152

script後接參數的運用

- 作中學五：利用 script 後接的參數來處理：
  - #!/bin/bash
  - echo "The script name is ==> $0"
  - echo "There are total $# arguments used in this script"
  - [-n "$1"] && echo "The 1st paramter is ==> $1" || exit 0
  - [-n "$2"] && echo "The 2nd paramter is ==> $2" || exit 0
  - [-n "$3"] && echo "The 3th paramter is ==> $3" || exit 0
    - 執行：ex> ./script.sh one two three

Shell script

153

利用特定目標 case ... esac

- 作中學六：利用『case ....esac』來進行特定參數的定義：
  - #!/bin/bash
  - case $1 in
  - "hello")
  - echo "Hello, how are you ?"
  - ;;
  - "")
  - echo "You MUST input parameters, ex> $0 someword"
  - ;;
  - *)
  - echo "Usage $0 {hello}"
  - ;;
  - esac

Shell script

154

固定數量迴圈for do done

- 作中學七：使用迴圈 part I：利用 for do done處理固定次數的迴圈動作
  - #!/bin/bash
  - for animal in dog cat elephant
  - do
  - echo "There are ""$animal""s.... "
  - done

Shell script

155

不固定數量迴圈while, until

- 作中學八：使用迴圈 part II：利用條件判斷來處理不特定的迴圈次數
  - #!/bin/bash
  - while ["$yn" != "yes"] && ["$yn" != "YES"]
  - do
  - read -p "Please input yes/YES to stop this program: " yn
  - done

精選範例

157

- script中的 $RANDOM有何作用？ C
  - (A) 讀取亂數
  - (B) 傳遞亂數
  - (C) 產生亂數
  - (D) 無任何作用
- 請問如何使你的Linux可有路由的功能？C
  - (A) echo “1” > /proc/net/sys/ipv4/ip_forward
  - (B) echo “0” > /proc/net/sys/ipv4/ip_forward
  - (C) echo “1” > /proc/sys/net/ipv4/ip_forward
  - (D) echo “0” > /proc/sys/net/ipv4/ip_forward

160

- 若有程式 ev.sh 如右所示，
  執行 ./ev.sh a b c 的結果為？B

```
#!/bin/bash
echo –n \$$#
echo –n “ is “
eval echo \$$#
```

  - (A) c is $3
  - (B) $3 is c
  - (C) \$ is \$
  - (D) \$$# is \$$#
- 如下關於變數的定義，哪些是不正確的？ ACD
  - (A) a1=b c
  - (B) a1= “b c”
  - (C) a1 = “b c”
  - (D) 1a= “b c”

158

- 執行 shell 時，用下列那個方法可以顯示每個執行步驟的回傳值？ AB
  - (A) sh –v test.sh
  - (B) sh –x test.sh
  - (C) sh –c test.sh
  - (D) sh –f test.hs
- 下列應用中，變數設定其傳回值何者正確？『now=`date`』 A
  - (A) 四 6 月 3 17:35:07 CTS 2004
  - (B) date
  - (C) $now
  - (D) 1

161

- 假設使用者寫了一支script名為 foobar.sh ，其內容為『#!/bin/bash; cd /tmp』並放在 ~/bin/ 內，然後使用者在家目錄執行該script，當script結束時，使用者的工作目錄會在？A
  - (A) ~/
  - (B) ~/bin
  - (C) /tmp
  - (D) /
- 假設我在目前目錄撰寫 test.sh ，但是我並沒有建立 PATH，則該如何下達指令執行？ BCD
  - (A) test.sh
  - (B) ./test.sh
  - (C) sh test.sh
  - (D) exec test.sh

159

- 要寫作一支 script 程式時，程式的第一行應該要如何撰寫？ B
  - (A) #/bin/bash
  - (B) #!/bin/bash
  - (C) #$/bin/bash
  - (D) #%/bin/bash
- 若有程式 s.sh 的內容如右所示
  當執行 ./s.sh a b c d e f g，輸出結果為？C

```
#!/bin/bash
shift
echo $@
shift 4
echo $@
```

  - (A) a b c d e f g; f g
  - (B) a b c d e f g; e f g
  - (C) b c d e f g; f g
  - (D) b c d e f g; e f g

162

- 有關 shell script的 case 用法，何者有誤？AB
  - (A) 以『end case』作為指令敘述的結尾
  - (B) 以『break』作為條件區塊的結束
  - (C) 『*』加上右刮號『*)』代表所有條件都不符合時，則執行其後的敘述
  - (D) 可使用參數『 $1』 接在執行檔案的後面執行之
- 一個shell script叫做 foo，foo能被執行的先決條件有哪些？ BC
  - (A) 為了安全考量foo可以有執行的權限開放，但不需要有讀的權限
  - (B) 如果要使命令 ./foo 能夠被所指定的shell所執行的話，foo中必須要有『#!』的符號指示
  - (C) foo必須要有可讀與執行的權限開放給欲執行foo的使用者
  - (D) shell script一定要用bash來執行

163

- 請問下列何者為bash的判別是用法？ BD
  - (A) if ... elseif ... fi
  - (B) if ... then ... elif ... then ... else ... fi
  - (C) if ... then ... elseif ... fi
  - (D) if ... then ... else ... fi
- 請問下列何者為 bash 使用迴圈的命令 ABD
  - (A) while
  - (B) until
  - (C) foreach
  - (D) for