

Linux系統管理基礎

授課講師

謝玉騰

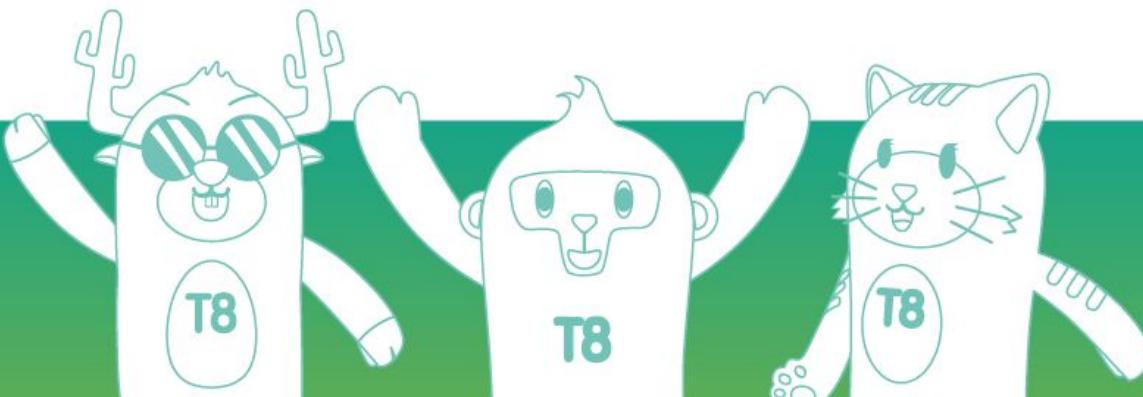
教材編寫

謝玉騰

緯
育 *TibaMe*

即學 · 即戰 · 即就業

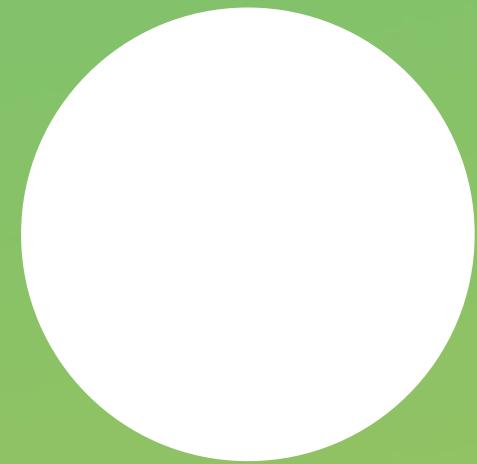
<https://www.tibame.com/>



課程大綱

- ◆ **Module 1. Linux 介紹**
- ◆ **Module 2. 發行版本**
- ◆ **Module 3. 系統安裝**
- ◆ **Module 4. 虛擬化工具**
- ◆ **Module 5. shell**
- ◆ **Module 6. 檔案系統**
- ◆ **Module 7. 基本指令介紹**
- ◆ **Module 8. 查指令說明**
- ◆ **Module 9. 輔助工具**
- ◆ **Module 10. 基本檔案管理**
- ◆ **Module 11. 文字檔處理工具**
- ◆ **Module 12. I/O重導與管線**
- ◆ **Module 13. 資料串流指令**
- ◆ **Module 14. 檔案搜尋**
- ◆ **Module 15. 檔案內容搜尋**
- ◆ **Module 16. 延伸正規表達式**
- ◆ **Module 17. 檔案目錄權限架構**
- ◆ **Module 18. 檔案目錄權限管理**
- ◆ **Module 19. 檔案打包壓縮**
- ◆ **Module 20. 文書處理器**
- ◆ **Module 21. 文書處理器vi/vim**
- ◆ **Module 22. 帳號管理**
- ◆ **Module 23. 群組管理**
- ◆ **Module 24. 軟體管理**
- ◆ **Module 25. 軟體編譯與建置**
- ◆ **Module 26. 程序管理**
- ◆ **Module 27. 背景程式**
- ◆ **Module 28. 工作排程**
- ◆ **Module 29. 系統服務**
- ◆ **Module 30. 網路設定**
- ◆ **Module 31. 網路架構與連接埠**
- ◆ **Module 32. 遠端連線**
- ◆ **Module 33. 檔案傳輸**
- ◆ **Module 34. bash 變數**
- ◆ **Module 35. 別名、系統設定檔**
- ◆ **Module 36. shell script**

授課講師介紹



謝玉騰

專長

虛擬化技術、網站後台建置、資料串流處理

簡歷

現任於中央氣象局資訊相關部門，負責網站後台 api 開發、觀測資料對接處理以及私有雲管理

老師的話

本門課程之設計是針對沒有相關作業系統背景知識或經驗的學習者為主要對象，課程中會以深入淺出的教學方式逐步導引您進入 Linux 世界

聯絡方式

fayteng@mfc.cwb.gov.tw

學習本課程須知

先備知識

熟悉電腦基本概念及操作

學習目標 (1)

- A. 認識Linux作業系統
- B. 認識虛擬化技術
- C. 熟悉CentOS的安裝
- D. 瞭解Linux檔案系統及重要目錄
- E. 學習使用Linux基本指令
- F. 利用管線及重導活化指令用法
- G. 學習文書處理相關操作
- H. 學習檔案搜尋相關操作
- I. 學習正規表達式相關應用
- J. 瞭解檔案與目錄權限

學習目標 (2)

- K. 檔案打包與壓縮/解壓縮
- L. 使用文書編輯器
- M. 帳號與群組管理
- N. 軟體套件建置管理
- O. 前景背景行程管理
- P. 設定工作排程與系統服務
- Q. 網路設定與應用工具
- R. Shell 指令稿撰寫技巧

須完成 哪些作業 或考試

於課程最後一天前完成課程評量

填寫網址

<https://forms.gle/4g3jmFQ8UQMcG31Q9>

補充教材

<https://sites.google.com/site/iiimyclass/Linux>

Module 1. Linux 介紹

- 1-1: Linux的由來
- 1-2: GNU 與 GPL
- 1-3: Linux Kernel

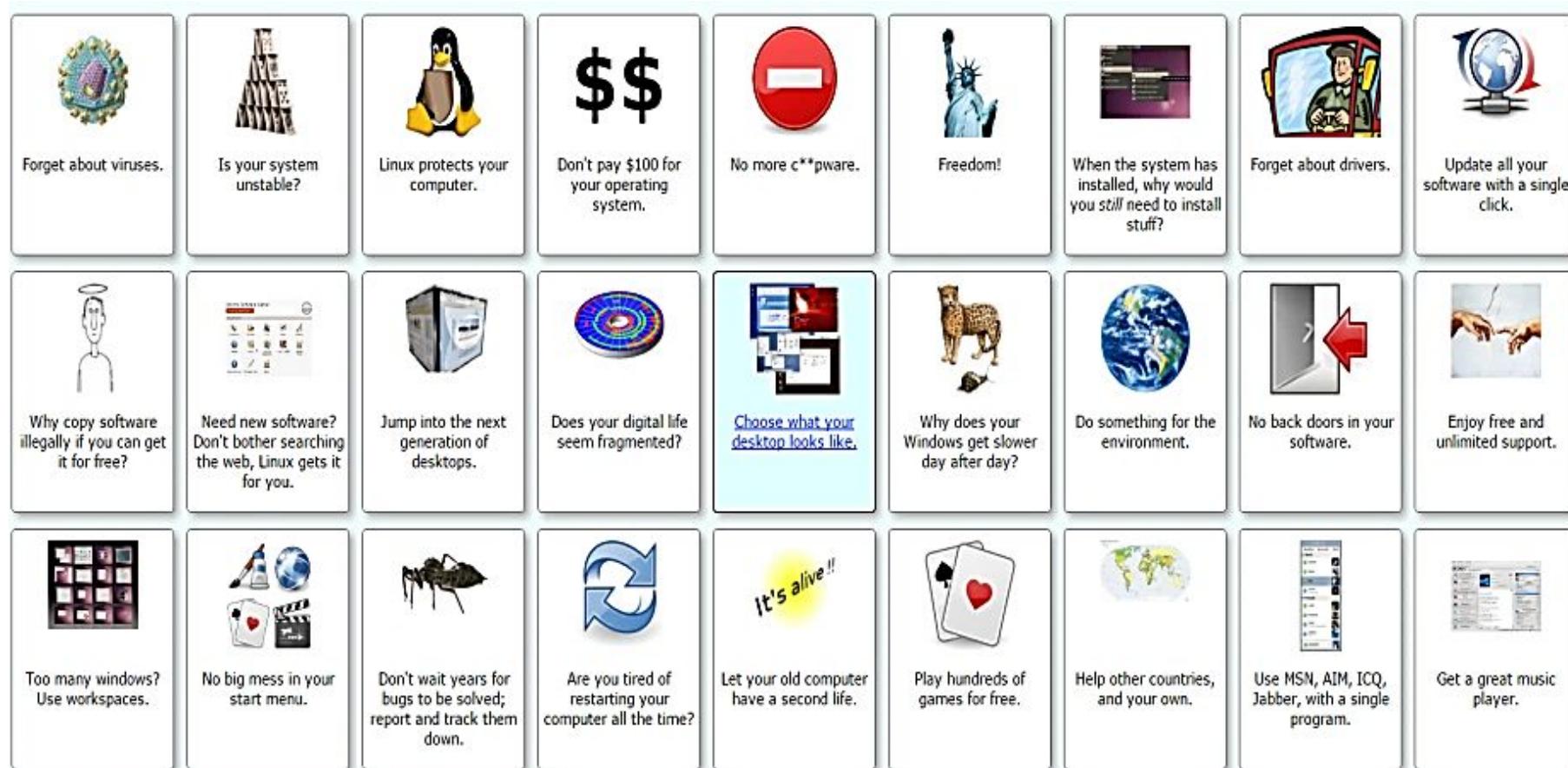
Linux作業系統

- 自由(Free)及開放原始碼(Open Source)
- 類Unix作業系統 (Unix-like)
- 可移植至多種硬體平台上
 - 例如個人電腦、伺服器、嵌入式系統等



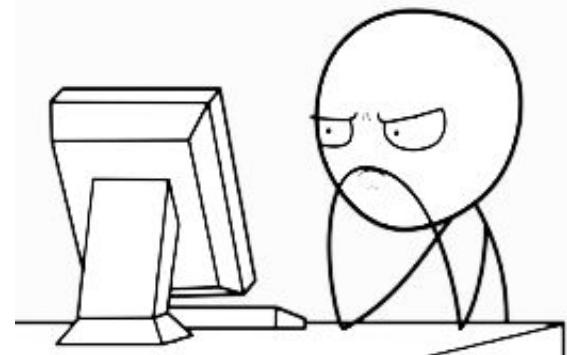
Linux作業系統的優點

- 開放原始碼
- 免費或少許費用
- 網路服務功能強大
- 系統穩定可靠
- 安全性高
- 支援多人多工
- 跨平台的移植性佳
- 圖形使用者介面多元
- 開發工具豐富



<http://www.whylinuxisbetter.net/>

為何多數人不用Linux?



Linux

Windows

遊戲/影音

免費

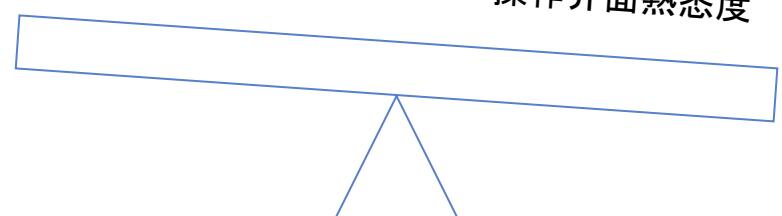
MS-Office

開放原始碼

軟體版本限制

多人多工

操作介面熟悉度

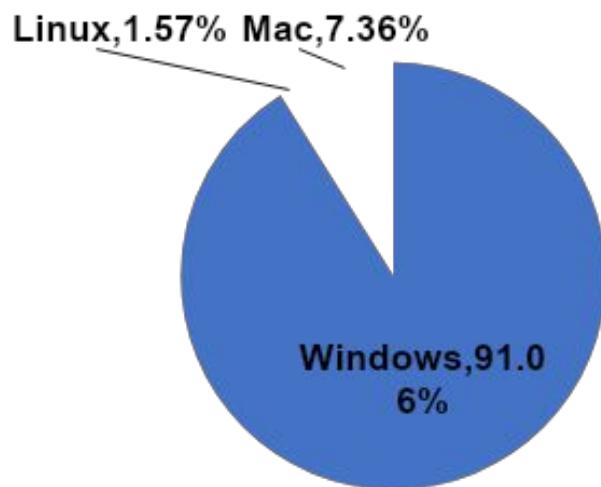


誰愛使用Linux?

- 企業與政府機關的伺服器
- 學生、研究生
- 研發人員
- 技術愛好者、駭客



Desktop 作業系統使用統計



<https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=0>

Public servers on the Internet

Unix-like				Microsoft Windows
All	Linux	FreeBSD	Unknown	
67.8%	35.9%	0.95%	30.9%	32.3%

https://en.wikipedia.org/wiki/Usage_share_of_operating_systems

Linux Users



What my friends think I do

What my mom thinks I do

What I think I do

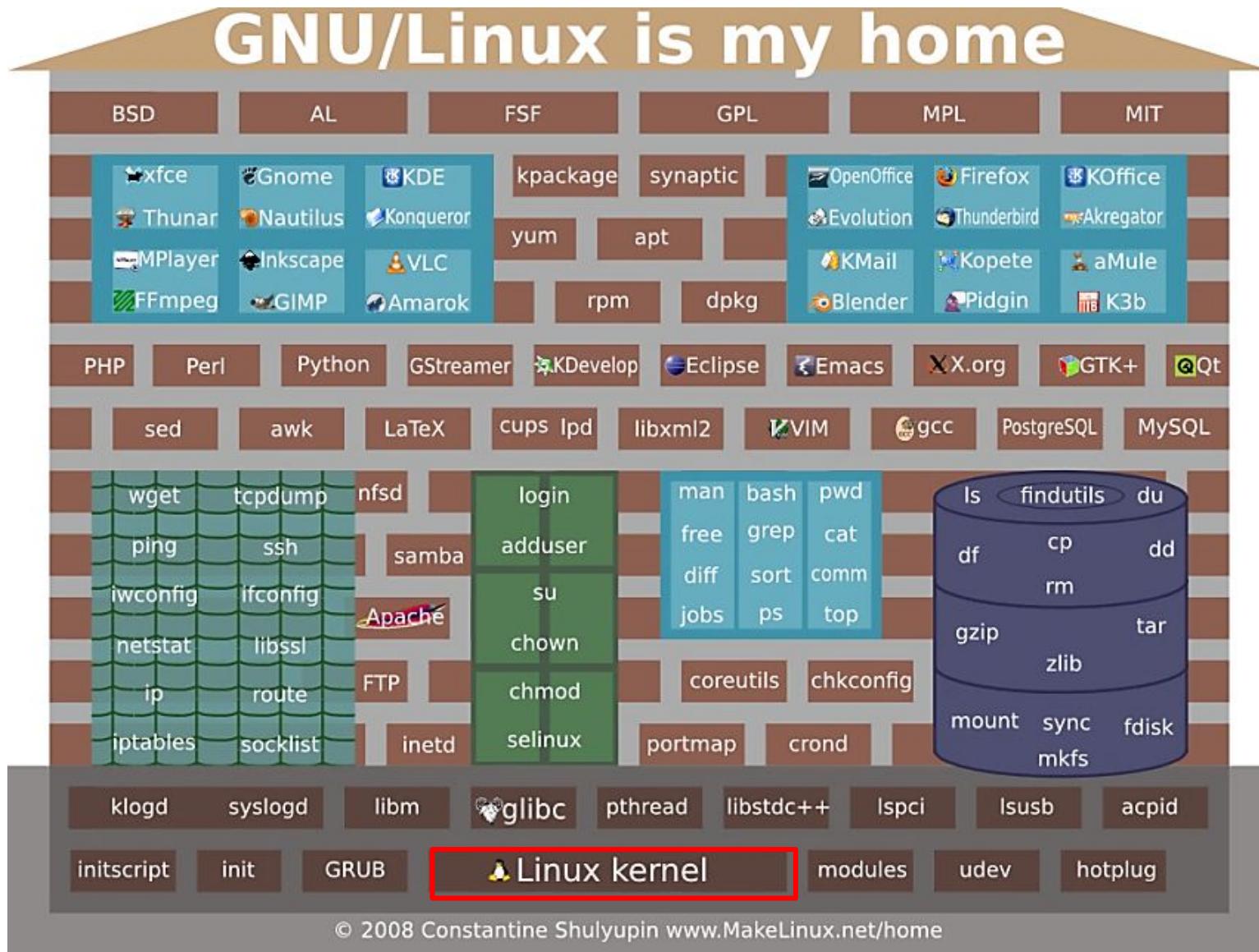


What Mac users think I do

What Windows users think I do

What I really do

資料來源：<https://www.pinterest.com/pin/47006389830524342/>



重大歷史記事

UNIX:1969

Thompson & Ritchie
AT&T Bell Labs

BSD:1978

Berkeley Software
Distribution

Commercial
Vendors:

Sun, HP, IBM, SGI,
DEC

GNU:1984

Richard Stallman,
Free Software
Foundation

POSIX:1986

IEEE Portable
Operating System
unIX

Minix:1987

Andy Tannenbaum

UNIX System V
Release 4:1989
AT&T and Sun

Open Source:1989
GPL ver 1

Linux:1991

Linus Torvalds
Intel 386 (i386)

資料來源 : The Linux Kernel: Introduction. CS591 (Spring 2001)

12

- GNU = GNU's Not Unix (意即「GNU並非 Unix」)
- 官方網頁 www.gnu.org
- 由Richard Stallman在1984年發起
- 目的是推出可取代Unix的自由軟體版本作業系統
- GNU開發了許多應用程式、函式庫(Library)、開發者工具、甚至遊戲等



非洲牛羚

- GPL通用公共授權條款 (General Public License)
- 由Richard Stallman於1989年撰寫，提供各類自由軟體使用，做為主要的授權聲明：
- 自由軟體代表使用者擁有執行、複製、散布、研究、修改、和改善軟體的自由
- 自由軟體講求自由，無關價格
- GPL版本歷史
 - GPLv1:1989
 - GPLv2:1991 (LGPL)
 - GPLv3:2007 (AGPL)



<https://zh.wikipedia.org/wiki/GNU通用公共授權條款>

- 由 Linus Torvalds 實作出最初雛型 (1991)
- Kernel 是作業系統的大腦，負責控制電腦的軟硬體資源，掌控系統運作做過程中的所有程式排程等
- 可與 GNU 軟體及函式庫整合，但不屬於 GNU 計畫
- 使用 GPLv2 授權
- 官網負責發行及維護核心
 - www.kernel.org



Tux企鵝



GNU/Linux 作業系統的組成



應用程式

- 讓使用者操作的軟體，例如瀏覽器

函式庫

- 提供應用程式基本功能

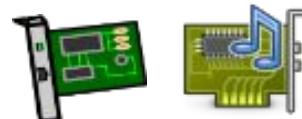
核心

- 核心(Kernel)是作業系統最重要的組成元件



驅動程式

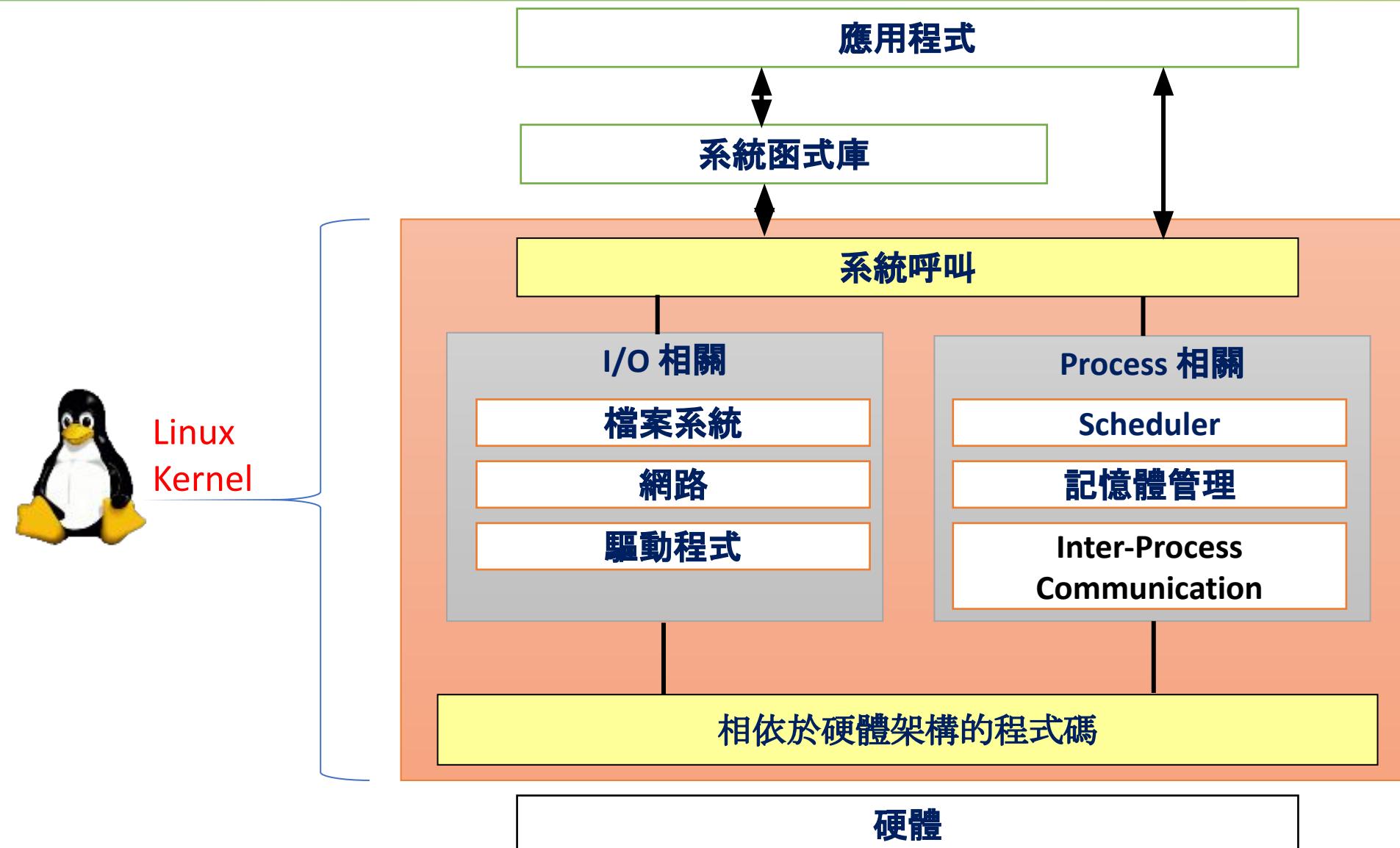
- 負責驅動底層硬體運作



硬體

- 例如：鍵盤、網路卡等

核心組成範例



參考來源 : The Linux Kernel: Introduction. CS591 (Spring 2001)

17

其他學習資源

- 鳥哥的Linux私房菜
 - <http://linux.vbird.org>
- 酷學園 (Study Area)
 - <http://www.study-area.org/linux/linuxfr.htm>
- Linux 台灣
 - <http://www.linux.org.tw/>
- Ubuntu Wiki : GNU/Linux 常用指令
 - http://wiki.ubuntu-tw.org/index.php?title=GNU/Linux_常用指令
- LINUX MAN PAGES ONLINE
 - <http://man.he.net/>

Module 2. 發行版本

- 2-1: RedHat**
- 2-2: Debian/Ubuntu**
- 2-3: CentOS**

Linux發行版本(Distribution)

- 各組織可自行從官網取得Linux核心，再自行整合偏好的應用軟體，推出Linux發行版本以一併包裝、發行或販賣
- 目前流行的有RedHat、CentOS、Ubuntu等

The Linux Kernel Archives

About Contact us FAQ Releases Signatures Site news

Protocol Location

HTTP <https://www.kernel.org/pub/>

GIT <https://git.kernel.org/>

RSYNC <rsync://rsync.kernel.org/pub/>

Latest Stable Kernel: 5.7

mainline:	5.7	2020-05-31	[tarball]	[patch]	[view diff]	[browse]		
stable:	5.6.15	2020-05-27	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	5.4.43	2020-05-27	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.19.125	2020-05-27	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.14.182	2020-05-27	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.9.225	2020-05-27	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.4.225	2020-05-27	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	3.16.84	2020-05-22	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
linux-next:	next-20200529	2020-05-29						[browse]

<https://www.kernel.org/>

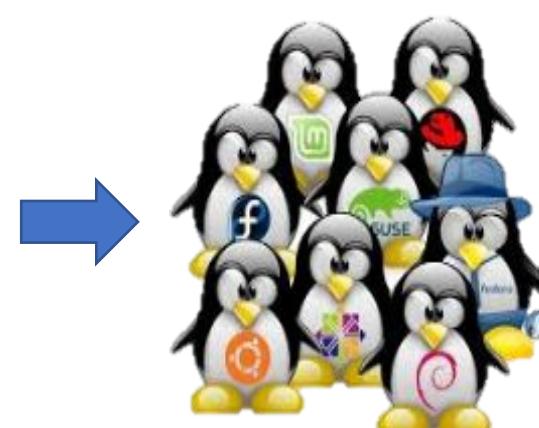
GNU

GNU is a Unix-like operating system that is free software—it respects your freedom. You can install GNU-based versions of GNU which are entirely free software. The GNU System provides a collection of applications, libraries, and developer tools, plus a program to allocate resources and talk to the hardware, known as a kernel.

For a complete list of GNU packages and more, visit www.gnu.org/software/.

#	Description	Homepage	Licence
3dof	Package for three-dimensional drawing with MetaPost output	https://www.gnu.org/software/3dof/	GPLv3
A2ps	Any-to-PsScript filter	http://www.gnu.org/software/a2ps/	GPLv3+orlater
Aboba	Service discovery system for DotGNU	http://savannah.gnu.org/projects/aboba/	GPL
Acc	GNU system accounting utilities	http://www.gnu.org/software/acc/	GPLv3+orlater
Acm	Aerial combat simulation game	http://www.gnu.org/software/acm/	GPLv3+orlater
Activation	JavaBeans™ Activation Framework	https://www.gnu.org/software/classpath/activation/	GPLv3
Adns	Resolver library for C and C++ programs	https://www.gnu.org/software/adns/	GPLv3
Alive	Automatic login and keep-alive utility for internet connections	https://www.gnu.org/software/alive/	GPLv3+orlater
Alus	Processes outgoing mail	https://www.gnu.org/software/alus/	GPLv3+orlater
ApI	Free version of the programming language API	https://www.gnu.org/software/api/	GPLv3+orlater
Archimedes	Software for designing and simulating semiconductor devices	https://www.gnu.org/software/archimedes/	GPLv3
Ari	A terminal logical pixel program	https://www.gnu.org/software/ari/	GPLv3+orlater
Arroundme	Social networking and team interaction software	http://www.gnu.org/software/arroundme/	GPLv3+orlater
Ariane	GNU Ariane web application framework written in Guile Scheme	https://www.gnu.org/software/ariane/	GPLv3+orlater
Aspell	Spell checker	http://aspell.net/	GPLv3.1
Auctex	Integrated environment for editing LaTeX and TeX files	http://www.gnu.org/software/auctex/	GPLv3
Autconf	Produces shell scripts which automatically configure source code	https://www.gnu.org/software/autconf/	GPLv3+orlater
Autconf-archive	Collection of free autconf macros	https://www.gnu.org/software/autconf-archive/	GPLv3+orlater with exception
Autogen	Automated program and test generation	http://www.gnu.org/software/autogen/	GPLv3+orlater
Autouse	Generates usable file links	https://www.gnu.org/software/autouse/	GPLv3+orlater

<https://www.gnu.org/>



Linux發行版本介紹: RHEL

- Red Hat公司成立於1995年，著重於開發企業級Linux伺服器
- Red Hat Enterprise Linux(RHEL)是一個商業市場導向的Linux發行版，被許多大型企業選擇做為建構資訊系統的平台
- <https://www.redhat.com/>

The screenshot shows the official Red Hat website. At the top is a dark navigation bar with the Red Hat logo, followed by links for Products, Solutions, Learning & support, Resources, and Red Hat & open source. Below the navigation, there's a large white header area. On the left, under 'LINUX PLATFORMS', is a section for 'Red Hat Enterprise Linux' described as 'The foundation for your enterprise hybrid cloud'. It includes a paragraph about its role as the world's leading enterprise Linux platform and links to 'Try It free', 'Buy It', and 'Talk to a Red Hatter'. On the right, a grey box highlights 'Red Hat Enterprise Linux 8.1 now available', noting new developer tools, security certifications, and automation capabilities, with a 'See what's new' button.

Linux發行版本介紹:Debian

- Debian是從1993 年由 Ian Murdock發起的，受到當時 Linux 與 GNU 的鼓舞，目標是成為一個公開的發行版
- 第一個穩定版本在1996年發布
- 以創始者Ian及其妻Debra命名 (Deb+Ian = Debian)
- 目前由Debian計劃成員進行開發與維護
- 眾多知名的Linux發行版，如Ubuntu、Knoppix及Kali等皆是在 Debian的基礎上發展而來



- Ubuntu是Linux的後起之秀，已成為主流版本之一
- 由Canonical公司發布並提供商業支援
- 以優良的硬體支援度著名，支援多種硬體平台
- 基於Debian發行版和GNOME桌面環境
- 每半年會發布一個新版本，每2年發布一個長期支援(LTS)版本
- 依各類使用者的需求，衍生出不同版本
 - Edubuntu (教育用)
 - Gobuntu (只使用自由軟體)
 - Kubuntu (KDE桌面環境)
 - Lubuntu (輕量型LXDE桌面環境)



Linux發行版本介紹: CentOS

- CentOS是與RHEL相容的發行版本，代表「社群的企業級作業系統」
 - CentOS = Community + Enterprise OS
- 免費的，被戲稱為窮人版RHEL
- 被多數企業組織採用，做為伺服器或工作站
- <https://www.centos.org/download/>
-



CentOS Linux 版本	次要發行	CD 及 DVD ISO 映像	套件	發行電郵	發行注記	停止支援期
8-Stream	N/A	DVD 及 NetInstall 映像 (連同檢查碼) 已收錄於各 鏡站	RPMs	CentOS Stream RHEL	CentOS Stream RHEL	不適用
8	1 (1911)	DVD 及 NetInstall x86_64 映像已收錄於各 鏡站 (檢查碼) .	RPMs	CentOS RHEL	CentOS RHEL	2029 年 5 月 31 日
7	8 (2003)	DVD、Minimal、Everything、LiveGNOME、LiveKDE 及 NetInstall x86_64 映像已收錄於各 鏡站 (檢查碼) .	RPMs	CentOS RHEL	CentOS RHEL	2024 年 6 月 30 日
6	10	i386 x86_64	RPMs	CentOS RHEL	CentOS RHEL	2020 年 11 月 30 日 **

Module 3. 系統安裝

- 3-1: 虛擬化技術**
- 3-2: 虛擬化軟體操作**
- 3-3: Desktop(桌面版)**

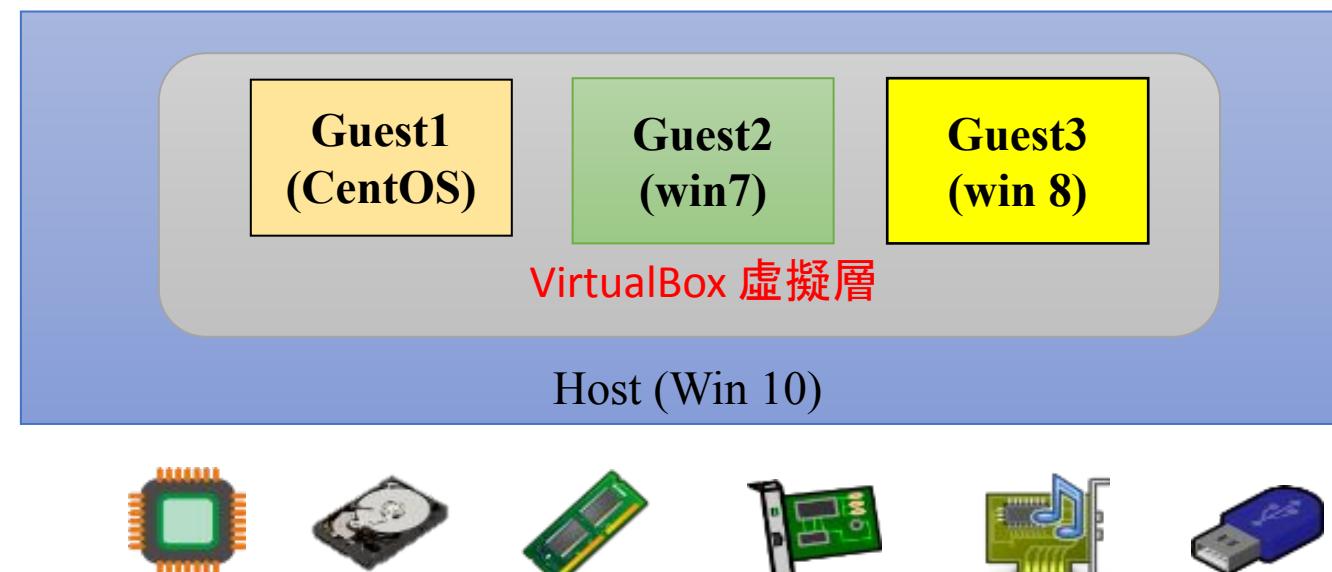
- 下載CentOS ISO
 - <https://www.centos.org/download/>
 - 選擇CentOS 7的DVD ISO (約4GB)
- 安裝VirtualBox
 - <https://www.virtualbox.org/wiki/Downloads>
- 建立虛擬主機
 - 以VirtualBox設定虛擬主機組態
- 安裝CentOS
 - 使用ISO檔案，於虛擬主機安裝CentOS
- 建立快照
 - 備份系統狀態

- CentOS是與Red Hat Linux企業版 (RHEL)相容的發行版本，代表「社群的企業級作業系統」
 - CentOS = Community + Enterprise OS
 - 被多數企業組織採用，做為伺服器或工作站
 - <https://www.centos.org/download/>
- 官網提供下列CentOS7版本：
 - 註：ISO映像檔是將光碟內容備份起來的一種檔案格式

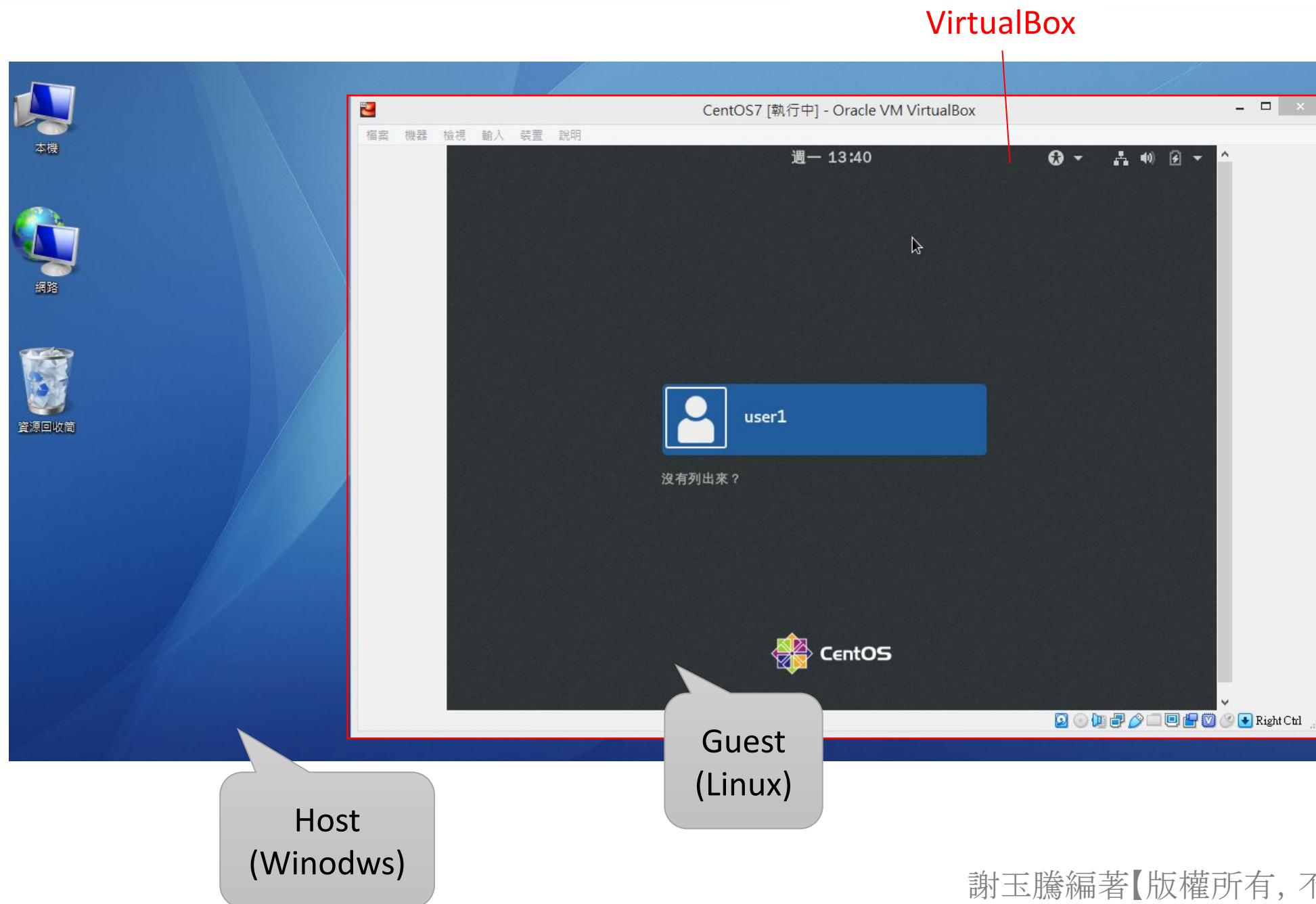


版本	大小	功能完整性
→ DVD ISO	適中，約4GB	含圖形介面，適合一般使用者
Everything ISO	最大，約7GB	功能最完善，適合開發及網管人員
Minimal ISO	最小，約636MB	純文字介面，多數軟體必須另行安裝

- Host : 實體主機
- Guest : 所模擬出的虛擬主機
 - 邏輯上彼此獨立
 - 每個 Guest 可設定不同的硬體資源(記憶體、硬碟、CPU)
 - 每個 Guest 可安裝各自的作業系統，並且可以同時開機運行



虛擬環境



- VirtualBox是一套免費的虛擬機器軟體
 - Guest可輕易新增、刪除、複製、**快照備份**、還原
 - 類似軟體還有Vmware Player、微軟Virtual PC等
 - 常用於練習及測試
 - www.virtualbox.org/wiki/Downloads



VirtualBox 6.0.14 platform packages

- ➔ Windows hosts
- ➔ OS X hosts
- Linux distributions
- ➔ Solaris hosts

請下載Windows hosts版本

VirtualBox 6.0.14 Oracle VM VirtualBox Extension Pack

- ➔ All supported platforms

Support for USB 2.0 and USB 3.0 devices, VirtualBox RDP, disk encryption, NVMe and PXE boot for Intel cards. See [this chapter from the User Manual](#) for an introduction to this Extension Pack. The Extension Pack binaries are released under the [VirtualBox Personal Use and Evaluation License \(PUEL\)](#). Please install the same version extension pack as your installed version of VirtualBox.

在VirtualBox建立新的虛擬主機(Guest)



- 版本選單沒有出現64位元?此時需調整BIOS設定

1. 重開機後立即按下del進入BIOS
2. 進階(F7)
3. 「中央處理器」的進階設定
4. 開啟「Intel®虛擬化技術」

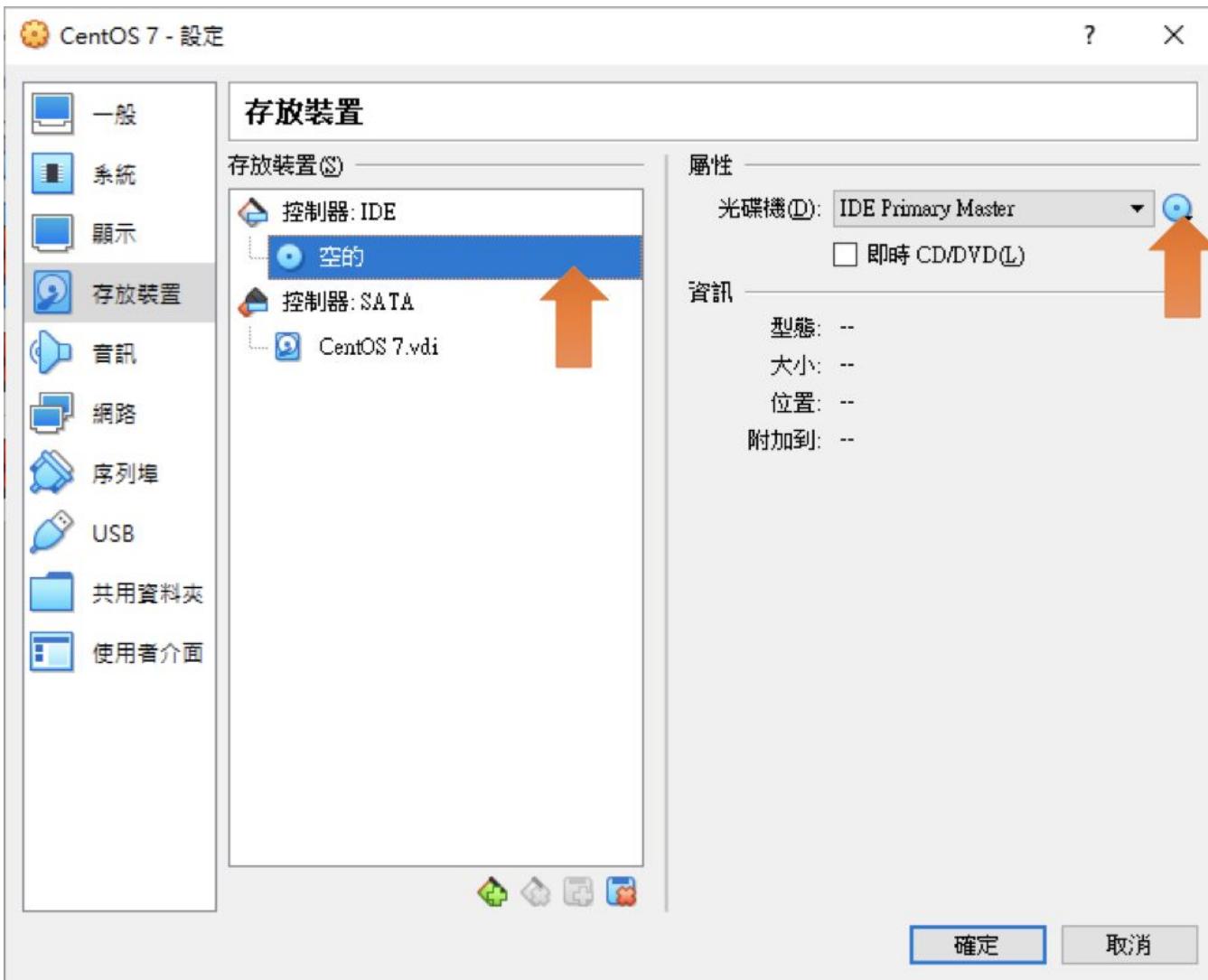


虛擬主機硬體設定完成

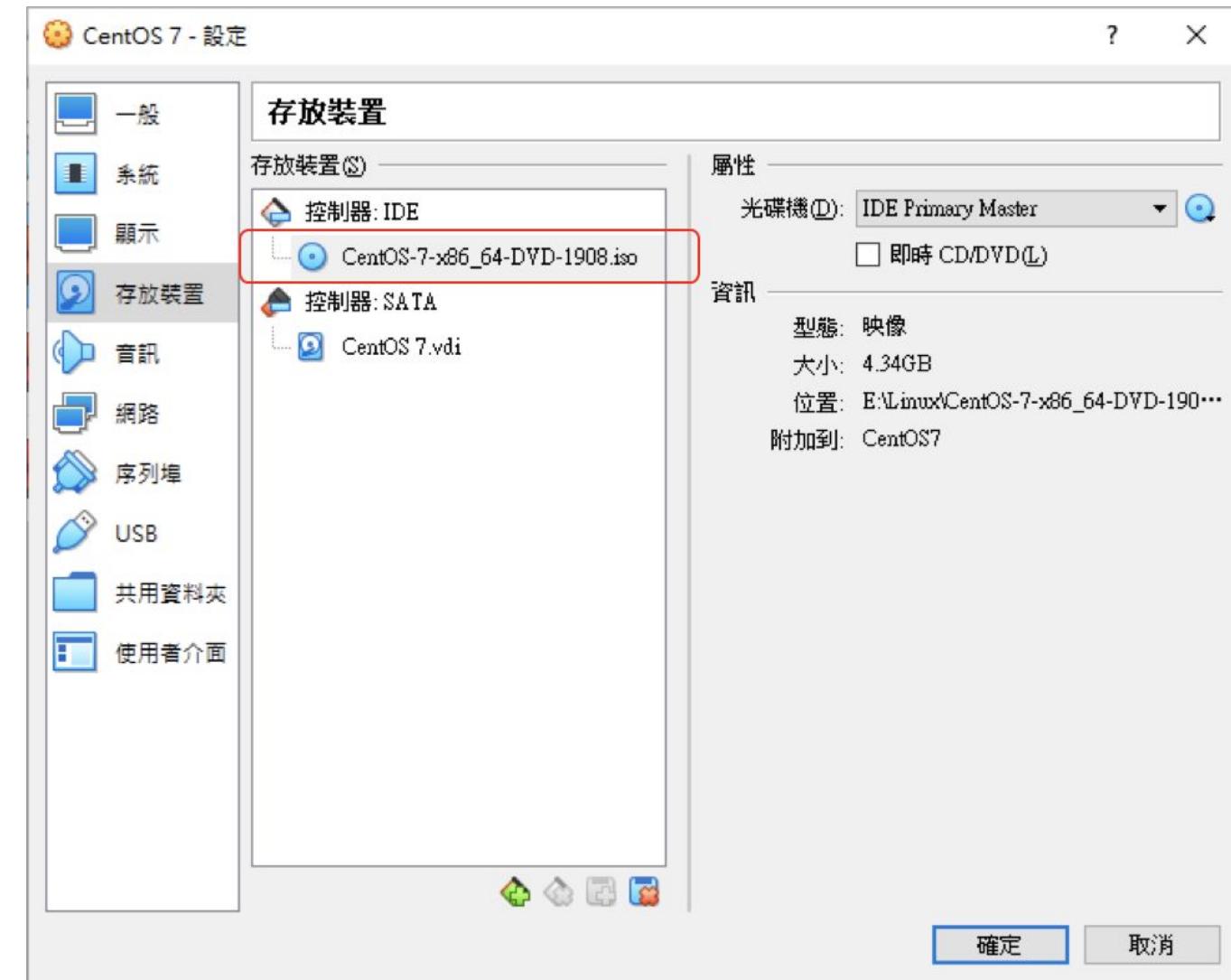


設定使用ISO檔案

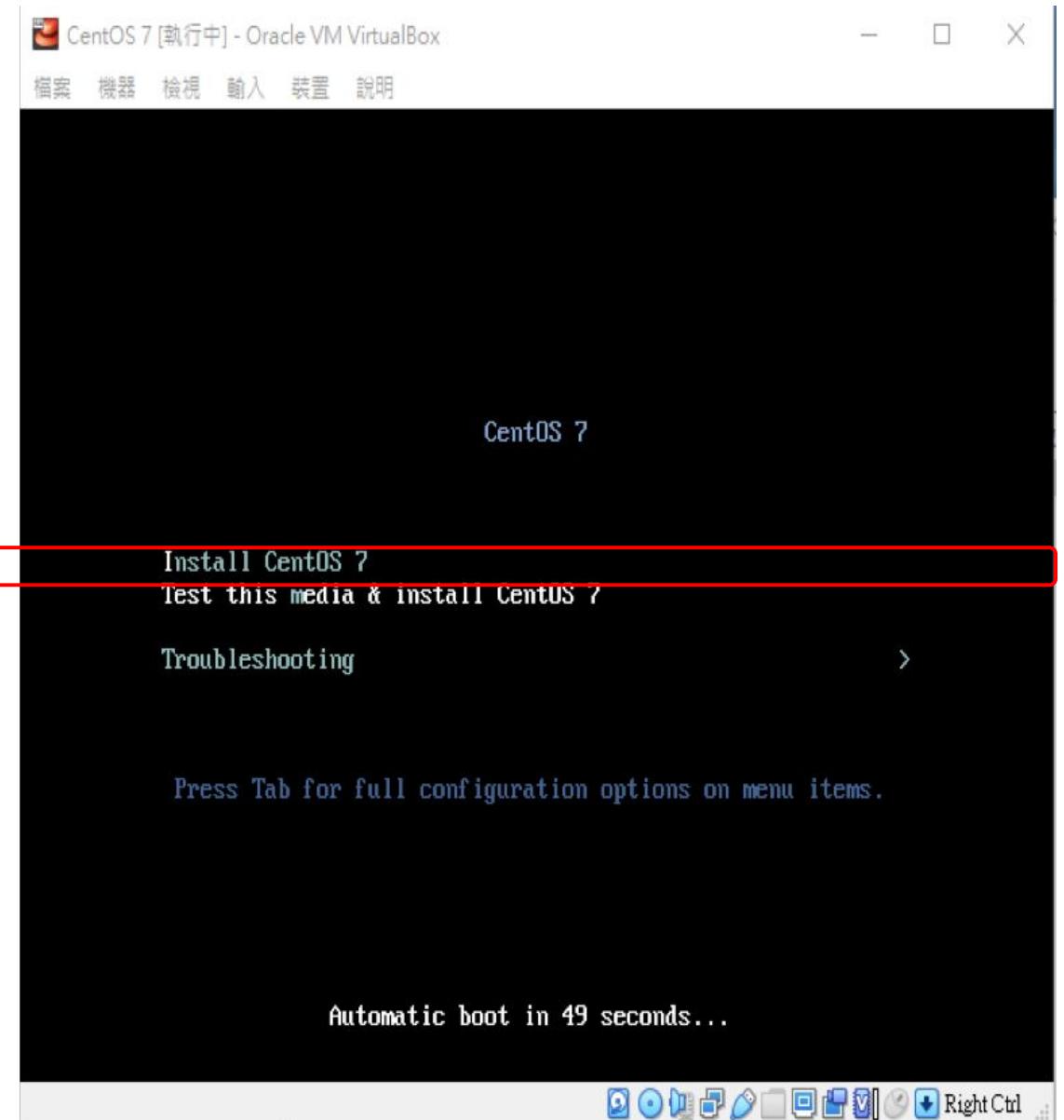
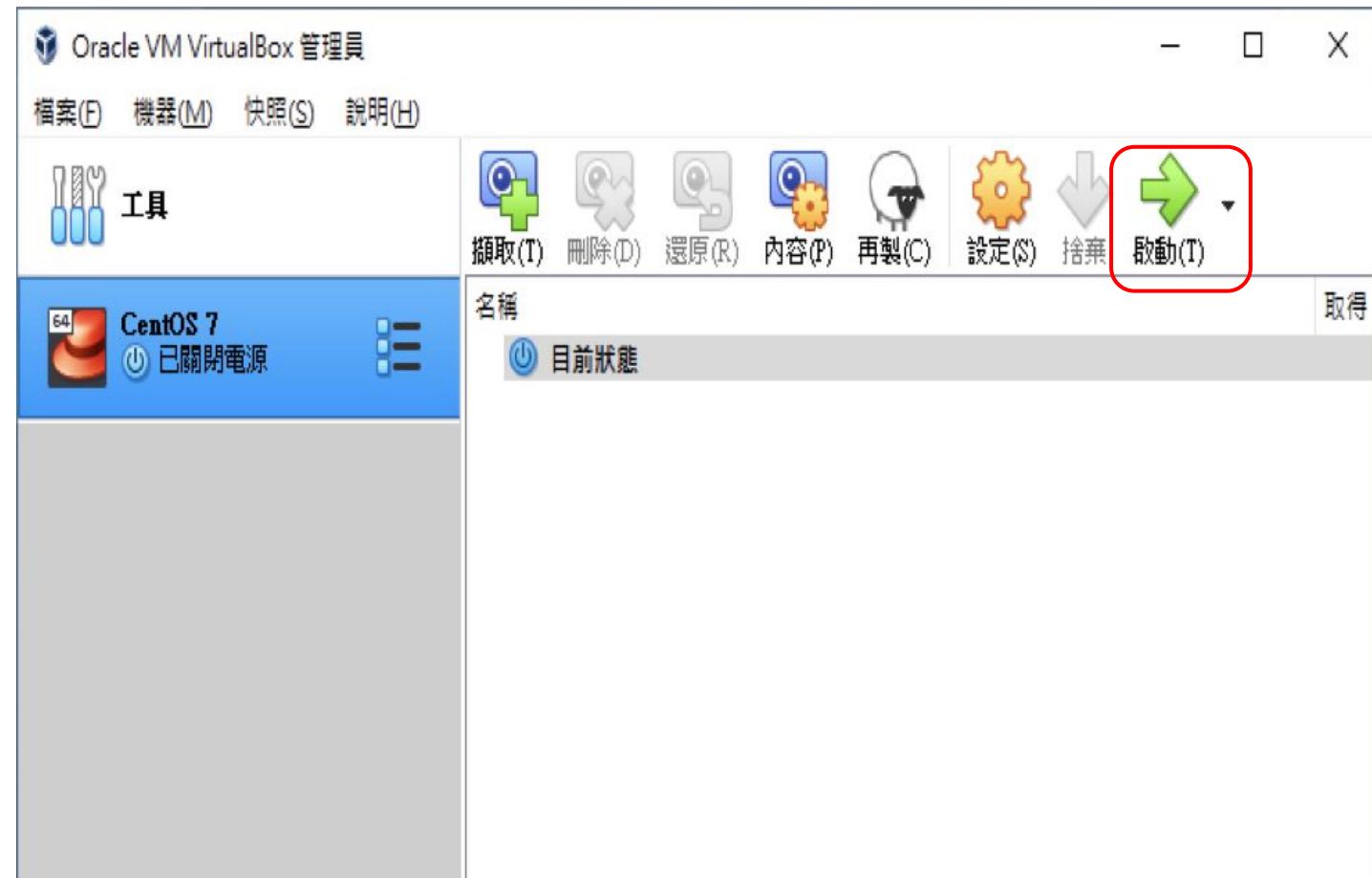
- 還沒使用 ISO檔案



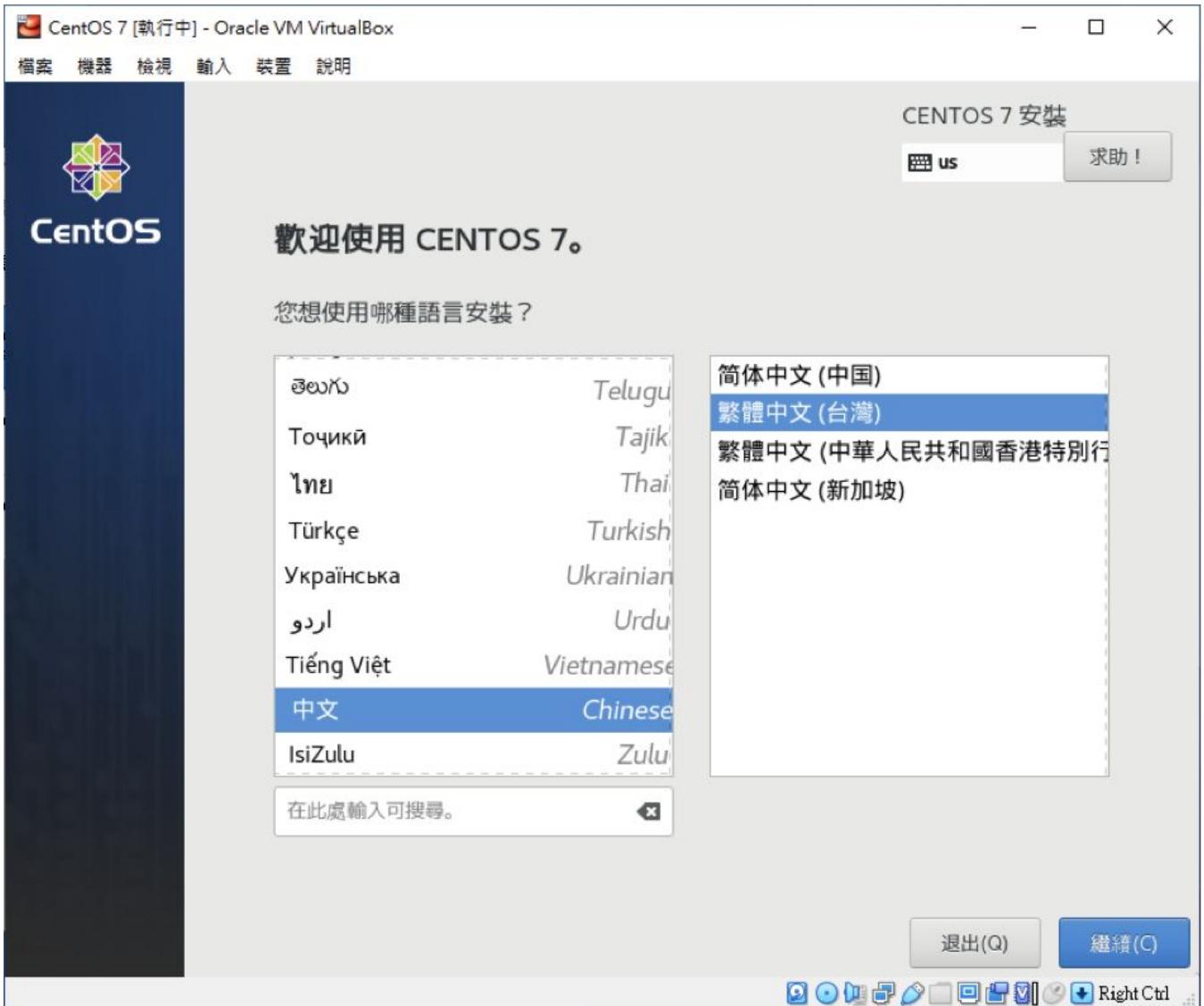
- 使用 ISO檔案



啟動電源以開始安裝



開始安裝CentOS 7



主要的設定畫面



- 設定為GNOME桌面環境
- 啟用網路

用戶設定

- 設定root (管理者)密碼



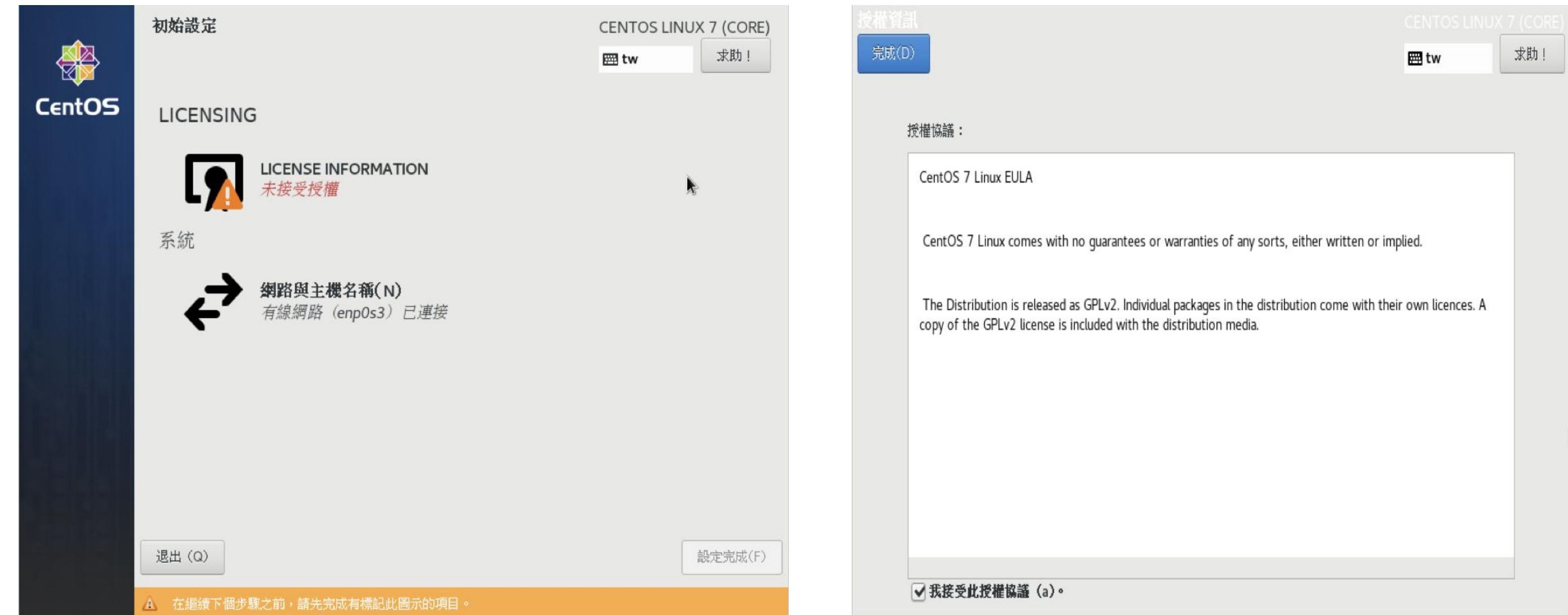
- 另外新增一個帳號供平常使用
- 勾選「讓這位使用者成為管理員」



於虛擬主機上安裝CentOS7

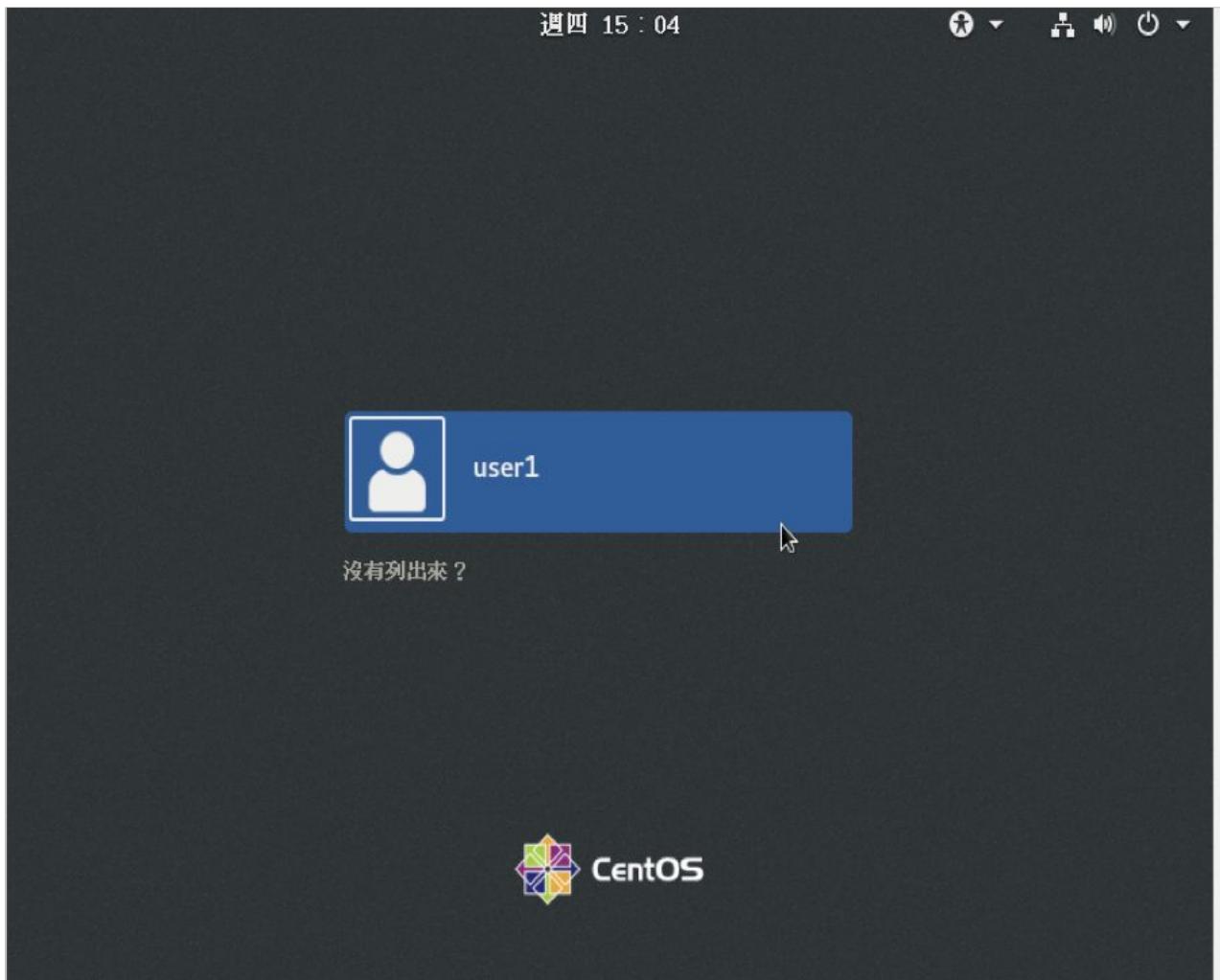


同意授權

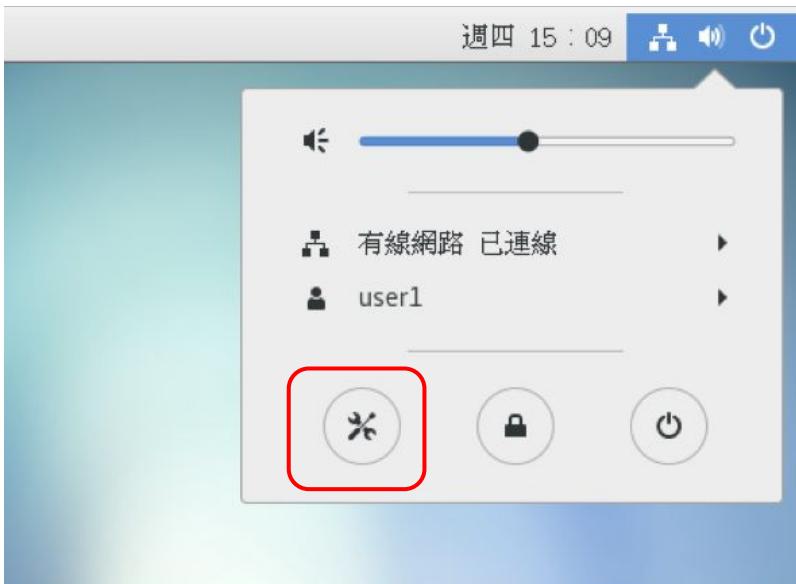


安裝完畢

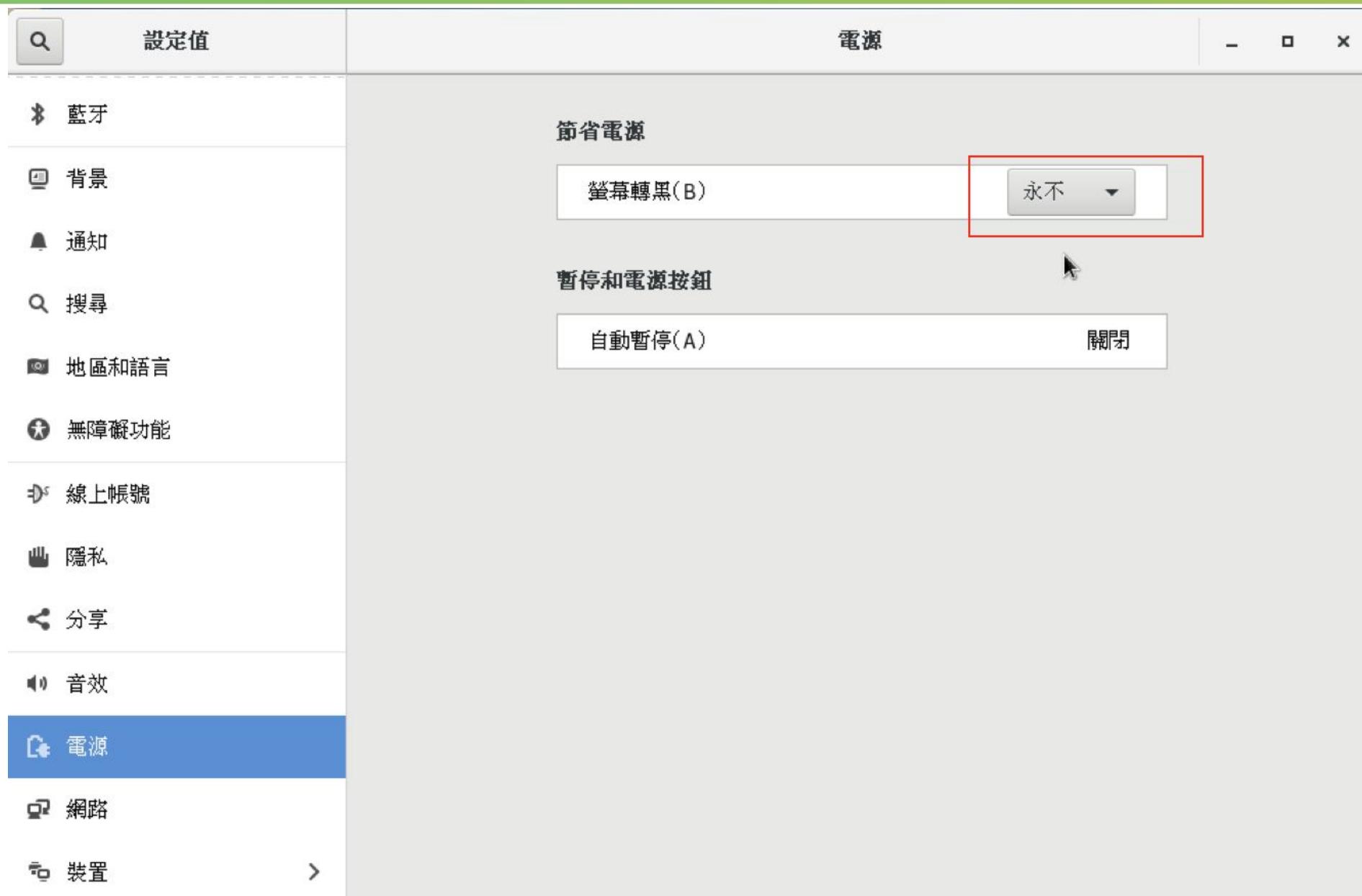
- 可以登入帳號密碼了！



設定作業系統解析度



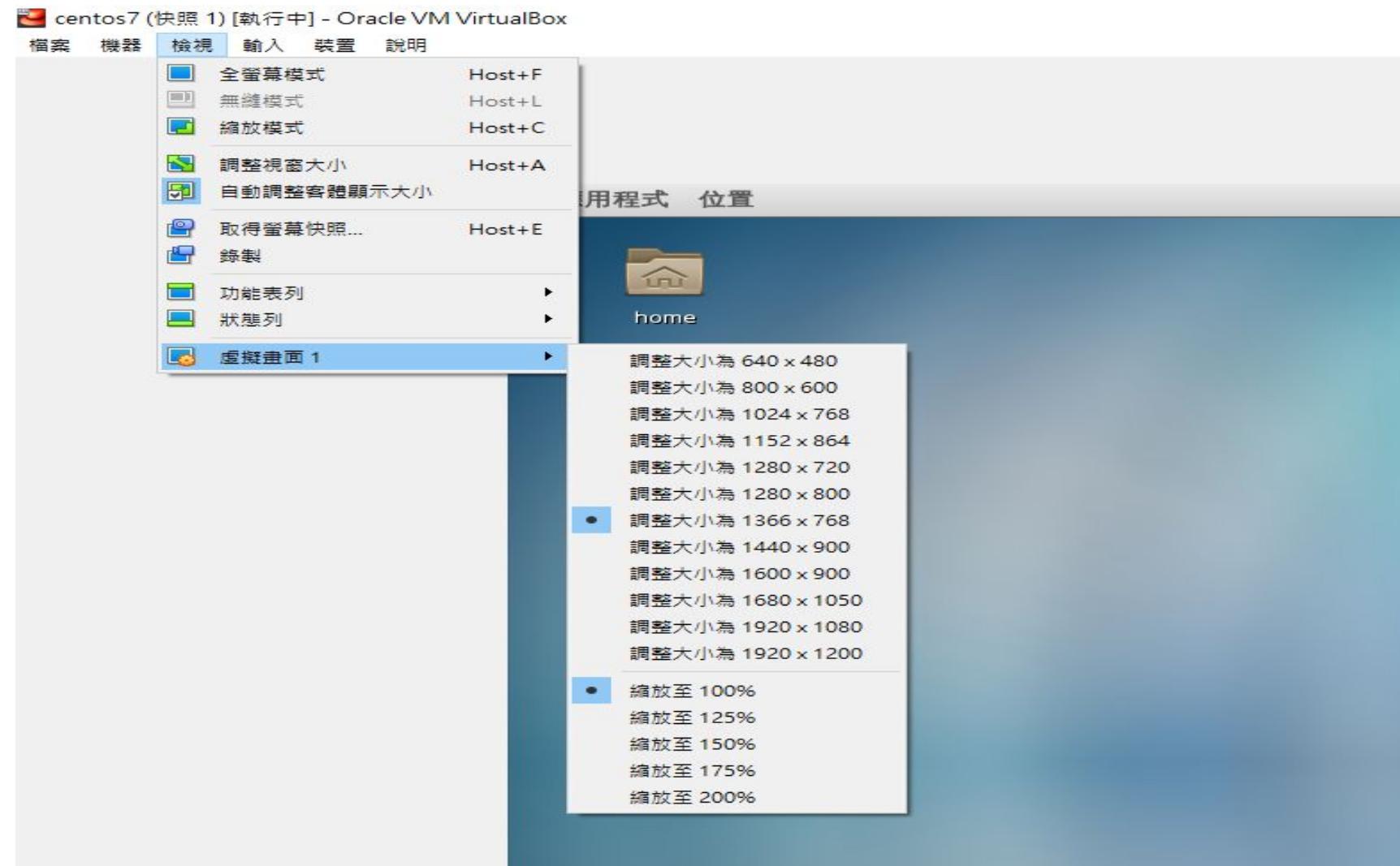
解除螢幕轉黑



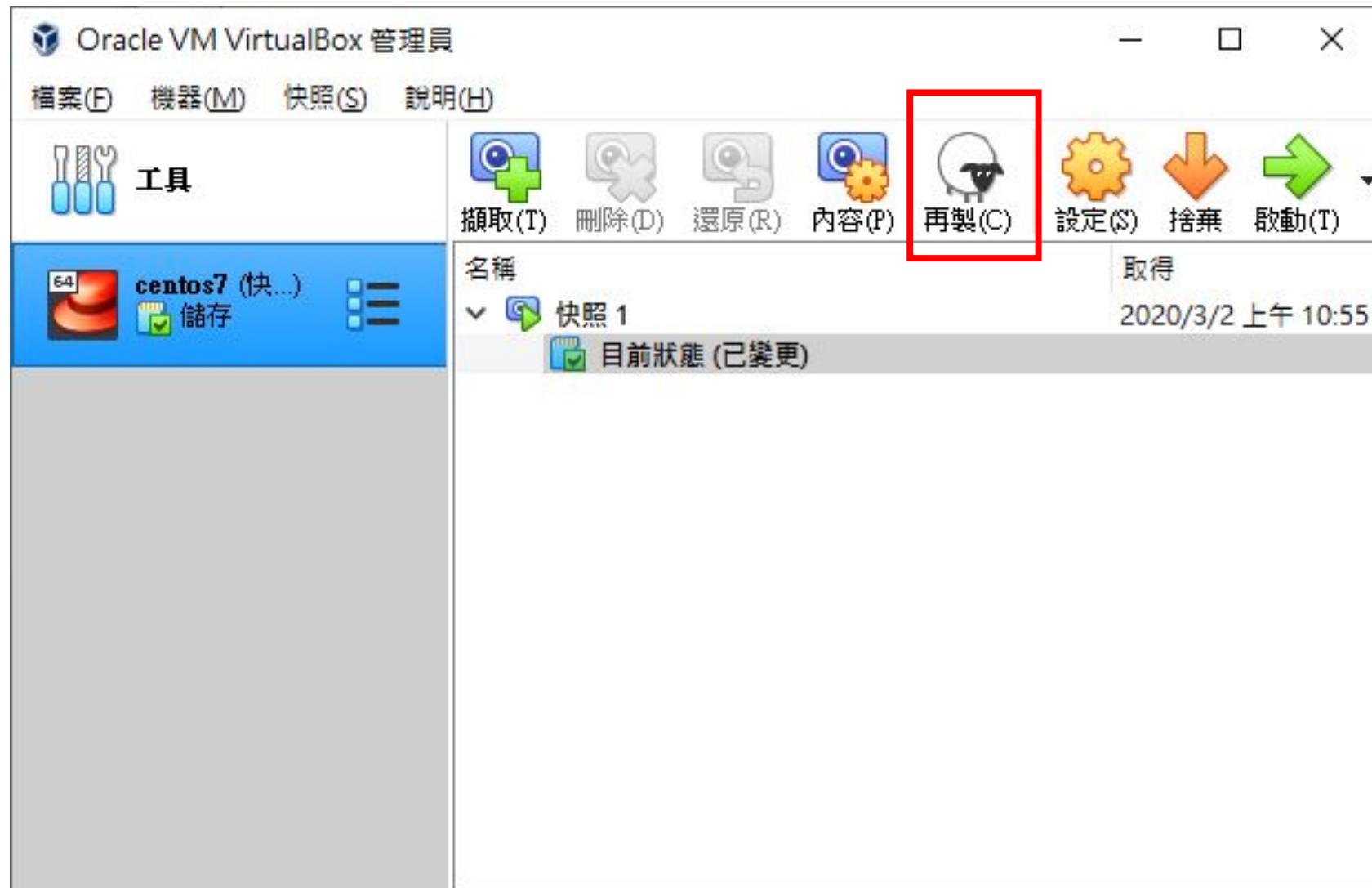
Module 4. 虛擬化工具

- 4-1: 虛擬機器再製
- 4-2: 快照備份

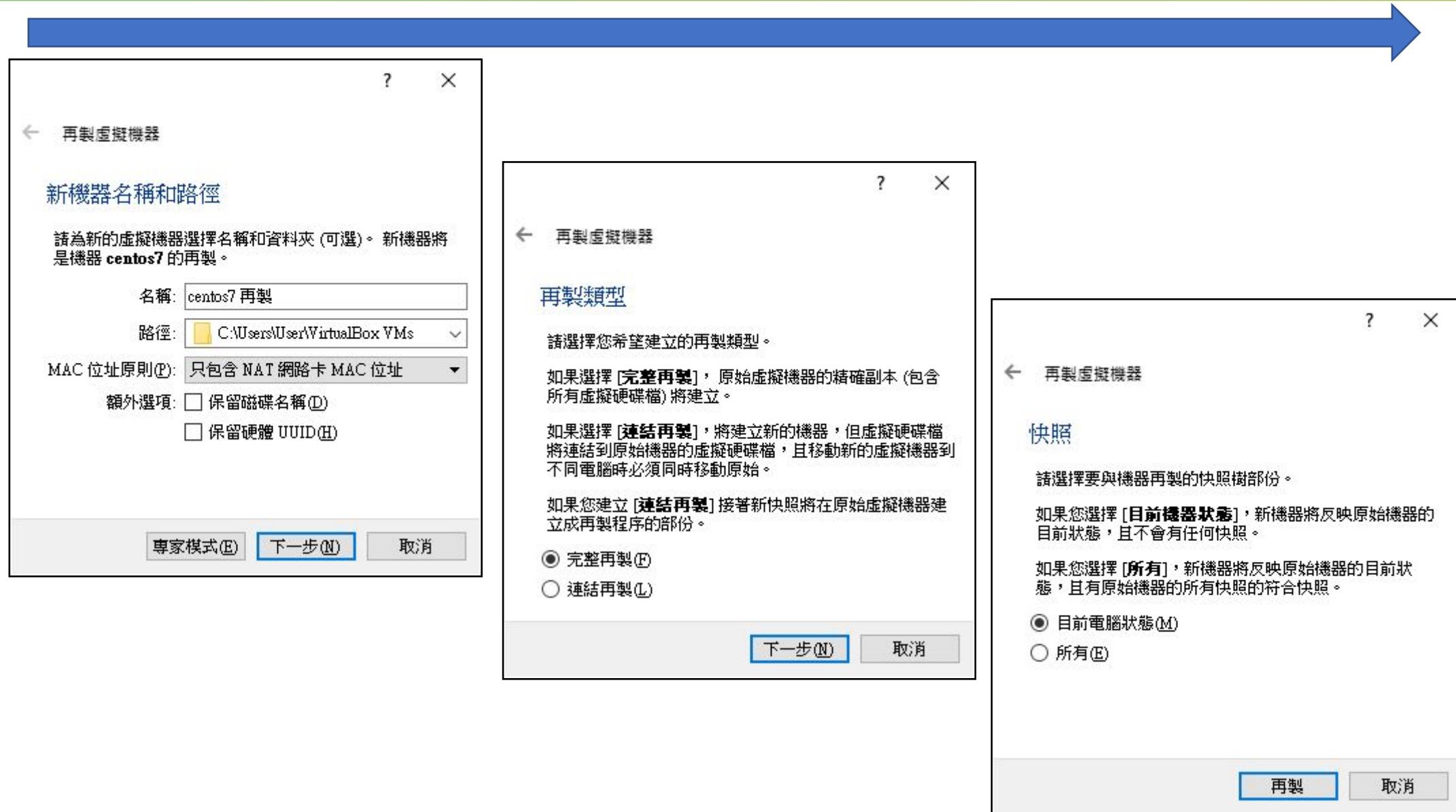
可進一步調整顯示畫面的大小



虛擬機器再製(Clone)

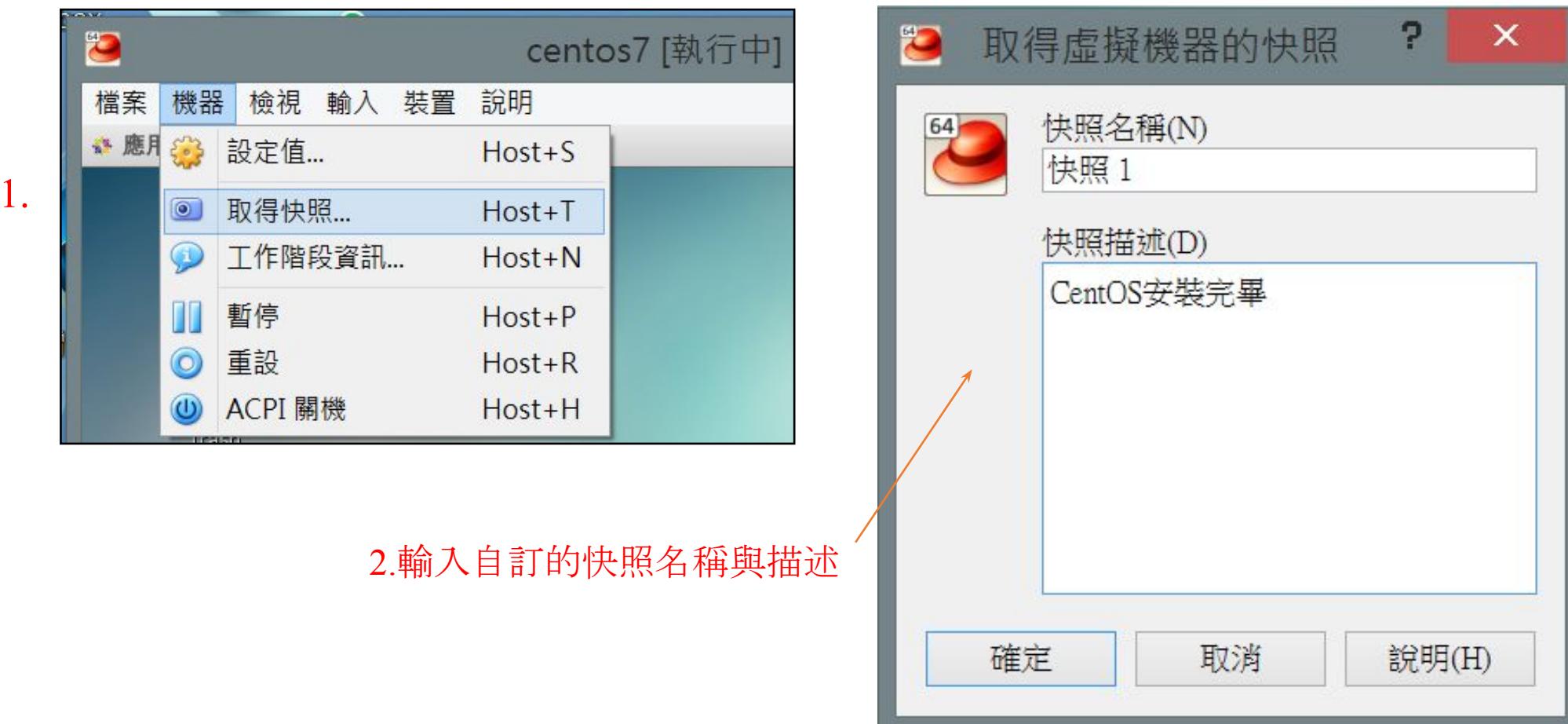


虛擬機器再製(Clone)



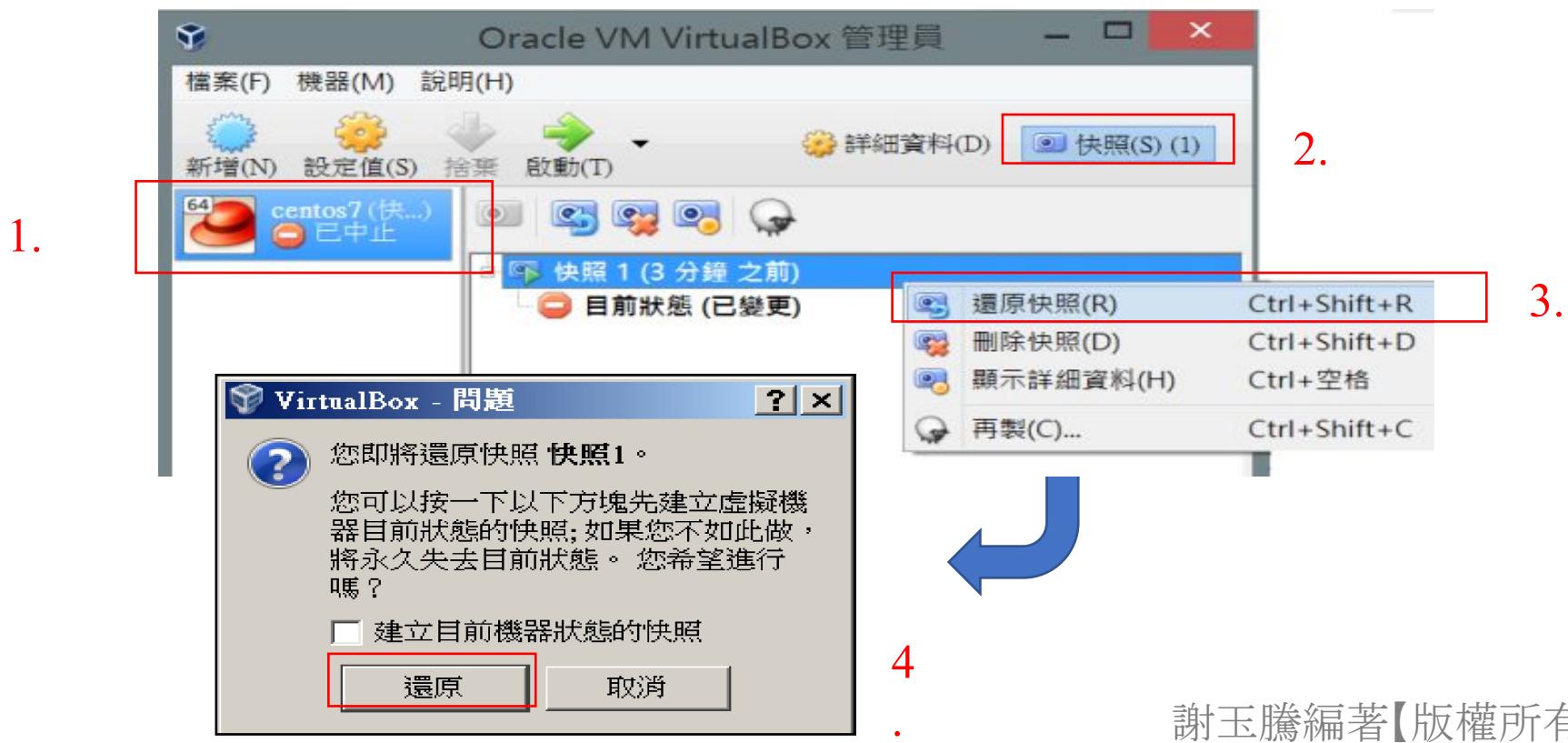
快照備份(Snapshot)

- 啟動CentOS
- 點選VirtualBox選單
 - 機器 □ 取得快照... Host+T



如何還原快照?

1. 關閉CentOS
2. 點選快照(S)
3. 挑選欲還原的快照
4. 按下滑鼠右鍵，點選"還原快照"

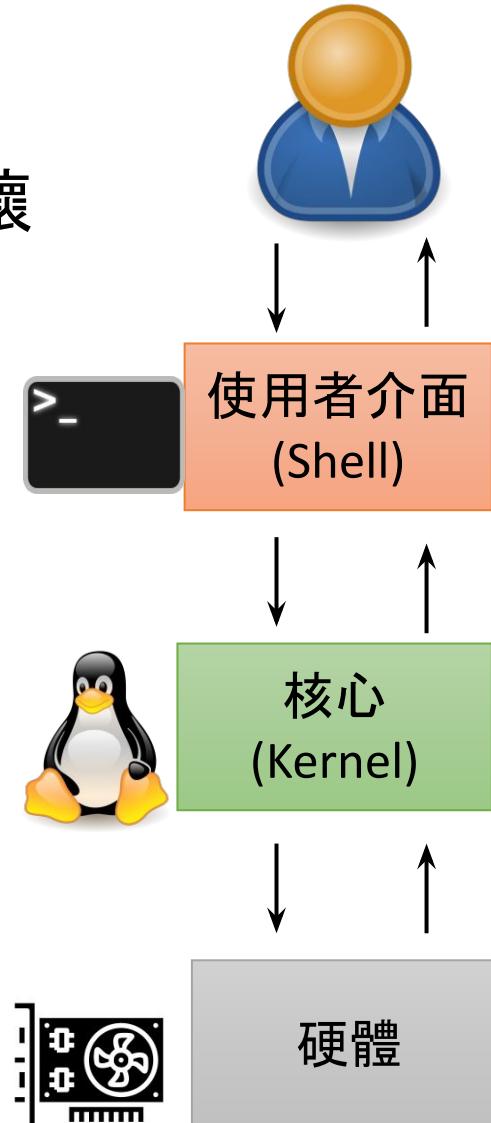


Module 5. shell

- 5-1: shell 基本介紹
- 5-2: 使用bash shell
- 5-3: 登入登出、shell切換

什麼是Shell?

- Shell是系統所提供的使用者介面
 - 讓提供使用者和核心之間進行溝通
 - 如此可避免使用者直接存取核心而造成破壞
- 操作流程例如：
 - 使用者輸入的指令(例如文件列印)
 - 交由Shell進行解釋後傳遞給核心去執行
 - 由核心控制相關軟硬體(例如印表機)
 - 最後呈現輸出結果



- 不同的Shell類型造成不同的操作方式



圖形化介面 (Graphic User Interface, GUI)

- 操作容易
- 介面美觀多元



文字介面 (Command Line Interface, CLI)

- 學習較難，必須熟悉指令
- 直接快速，跨平台的一致性操作
- 自動化 (shell 腳本編寫)
- 支援遠端多人多工操作

Linux 圖形介面

- 由X-Window(也稱為X11或X)函式庫提供圖形操作介面的開發基礎，其訂定了標準化的顯示協定與工具
- 可自行設計使用者畫面的呈現方式
- 主要的桌面環境分為GNOME及KDE兩種

CentOS



Ubuntu



OpenSUSE



Debian



Fedora



Gentoo

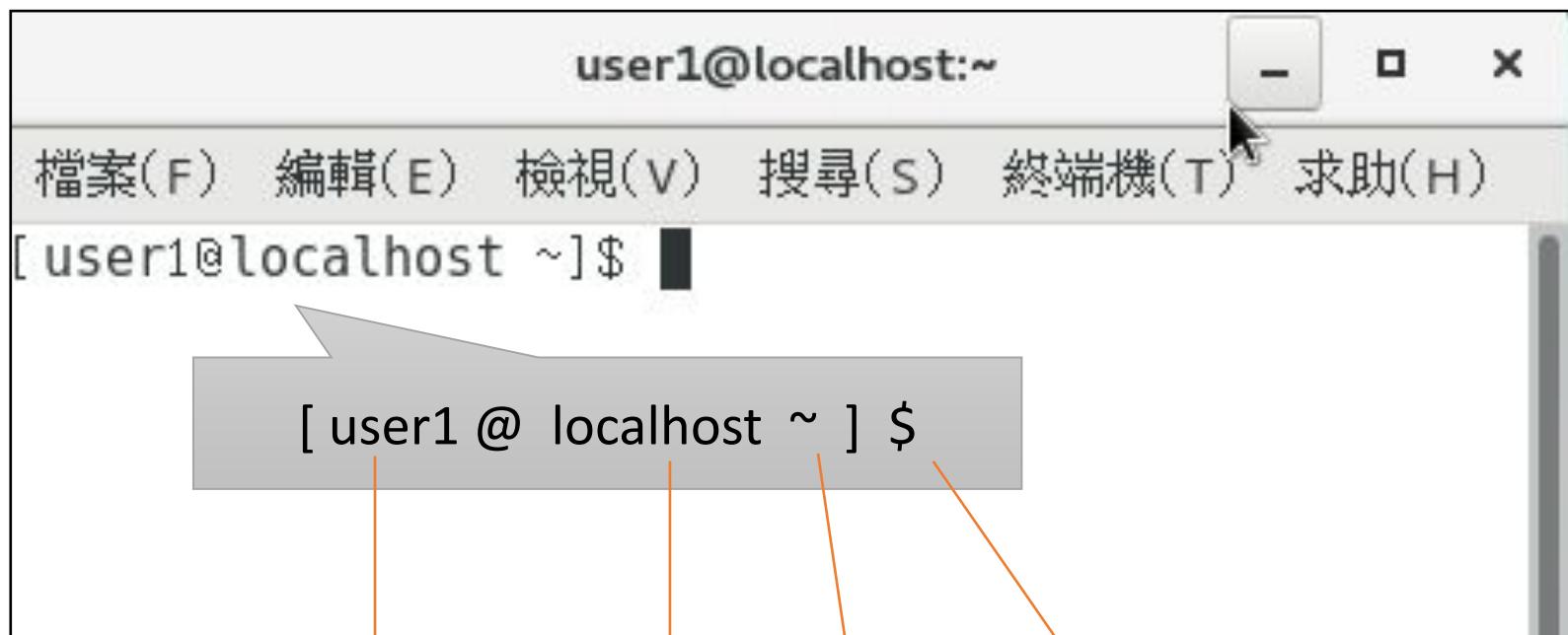


- Linux文字介面以bash (Bourne-Again shell)為主流
 - 其他文字Shell還包含sh、dash、ksh、zsh及csh等
- bash是一隻程式，負責持續接收使用者的輸入(如指令或資料)，並傳遞給系統，並呈現執行結果
- bash支援強大的操作功能：
 - 自動完成
 - 萬用字元
 - 指令執行紀錄
 - 別名



使用終端機軟體

- CentOS選單列->應用程式->系統工具->終端機
 - 會於圖形介面內開啟終端機軟體，內部會啟動一個bash



使用者帳號 主機名稱 目前目錄

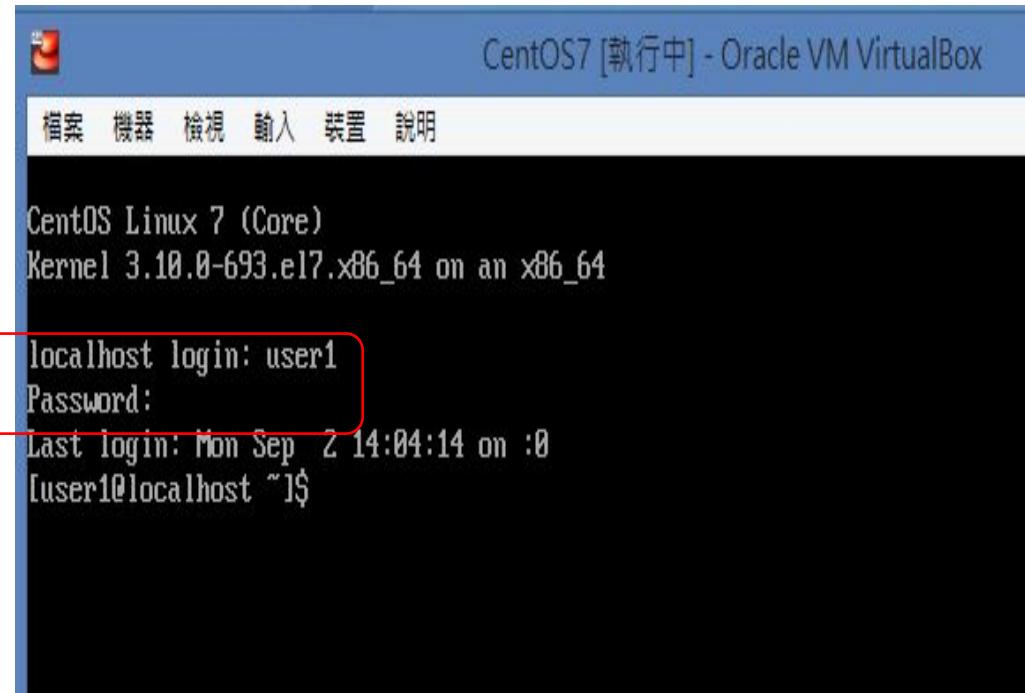
Shell提示符號(prompt)

一般帳號的提示符號為 \$
root帳號的提示符號則為 #

切換至文字介面Shell

- CentOS圖形介面下可切換至虛擬主控台(Virtual Console)所提供的文字介面Shell
 - Ctrl-Alt-F2 ~ Ctrl-Alt-F6
- 若要登出可用exit或logout
- 若要切換回圖形介面則可用Ctrl-Alt-F1

輸入帳號密碼後
開始操作



如何結束當前的Shell

- exit 指令
 - 可結束任何shell
 - 例如圖形介面終端機的shell、文字介面的shell
- logout 指令
 - 只能結束login的shell，例如文字介面shell
 - 但文字介面shell如果被結束，系統仍會自動生出一個新的Shell，如此才能讓下一位使用者進行登入

Module 6. 檔案系統

- 6-1: 檔案系統(ext/xfs)
- 6-2: 檔案階層式標準 FHS
- 6-3: 檔案與目錄路徑

什麼是檔案系統(File System)?

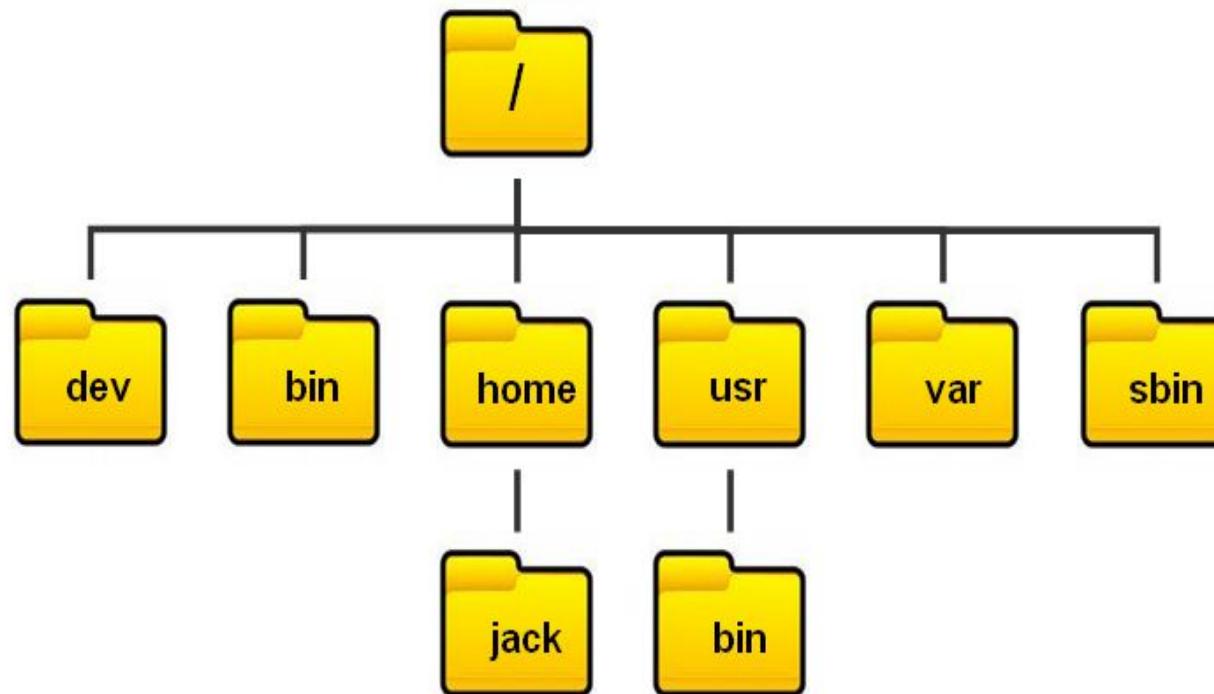
- 檔案系統是一種用來存取目錄或檔案資料的機制
- 常用類型：
 - 微軟作業系統常用NTFS
 - 隨身碟常為Fat32
 - CentOS 6使用EXT4
 - CentOS 7則用XFS
- 檔案系統類型將影響：
 - 檔案路徑存取方式
 - 效能與穩定性
 - 容量上限
 - 容錯能力

File System	Max File Size	Max Partition Size	Journaling
Fat16	2 GB	2 GB	No
Fat32	4 GB	8 TB	No
NTFS	2 TB	256 TB	Yes
ext2	2 TB	32 TB	No
ext3	2 TB	32 TB	Yes
ext4	16 TB	1 EiB	Yes
reiserFS	8 TB	16 TB	Yes
JFS	4PB	32PB	Yes (metadata)
XFS	8 EB	8 EB	Yes (metadata)

註: 1TB=1000GB, 1PB=1000TB, 1EB=1000PB

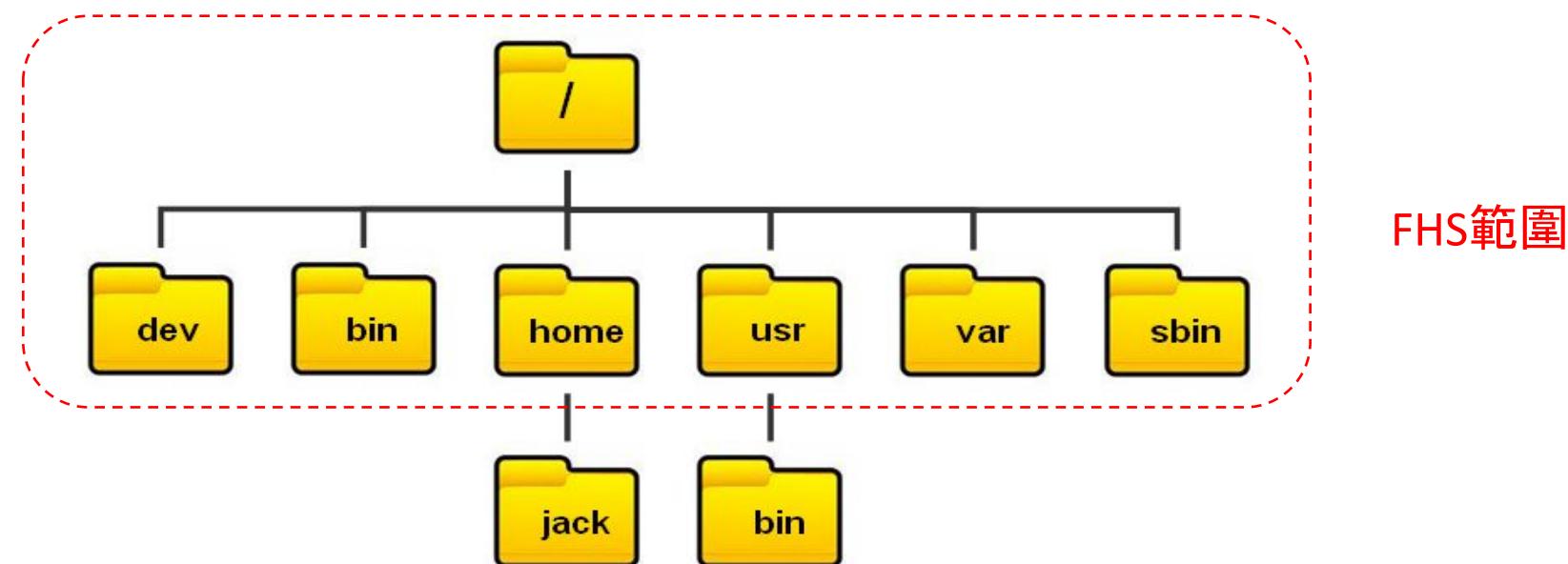
58

- 單一樹狀結構，最頂層為根目錄(以 / 表示)，所有檔案或目錄皆是由根目錄往下延伸



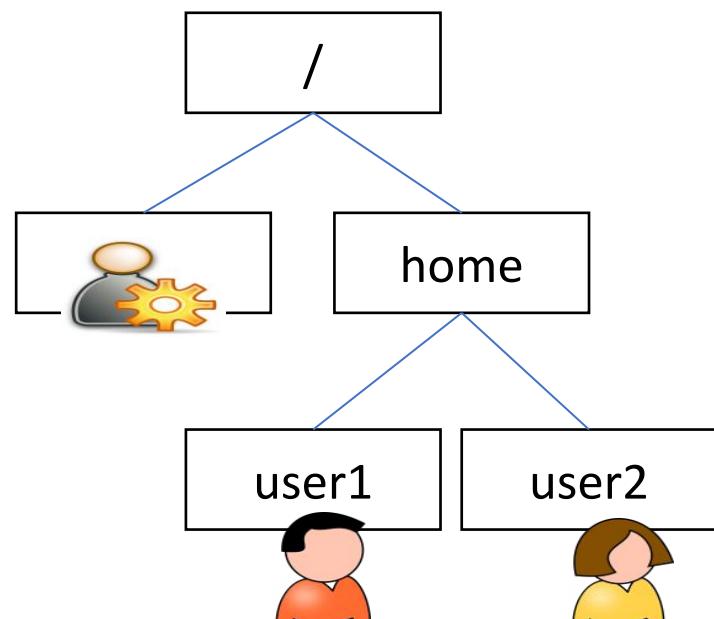
檔案階層式標準

- FHS (FileSystem Hierarchy Standard)
- 定義了Linux作業系統中的主要目錄及目錄內容
 - 只定義根目錄及下一層目錄的內容
 - 更下層目錄則由各Linux發行版本自行定義
- 由Linux基金會維護，目前版本為3.0版



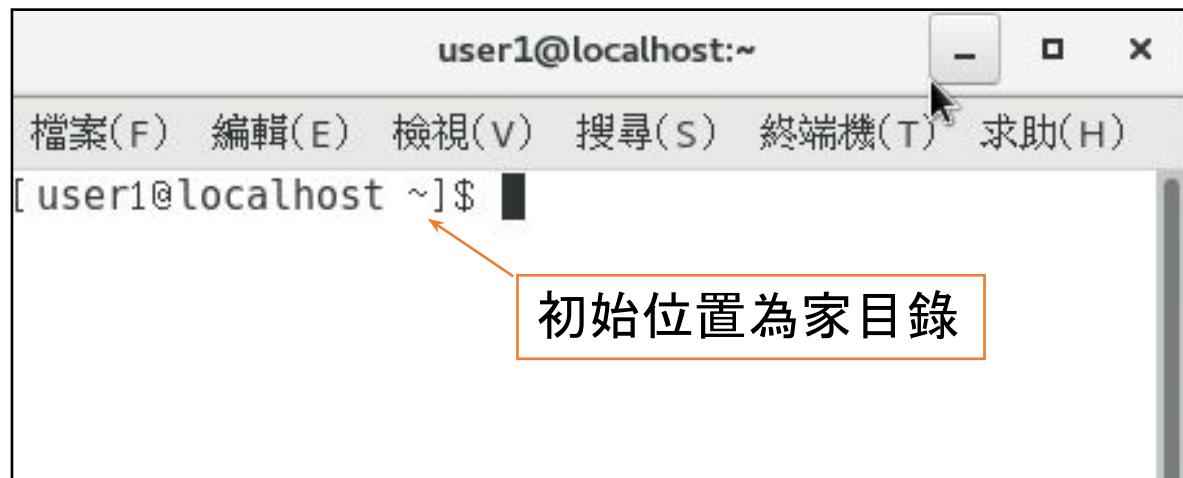
帳號與家目錄

- 每個使用者帳號會擁有自己私人的目錄空間，只有該帳號及系統管理者(root)才有存取該目錄的權限
- 家目錄名稱預設與使用者帳號相同
 - 一般使用者的家目錄為 /home/帳號/, 例如/home/user1/
 - root帳號的家目錄則為 /root/



家目錄的作用

- 當使用者啟動bash時，系統預設會切換至家目錄，也就是預設下達指令的位置
- 波浪符號(~)用來代表該帳號的家目錄，以供方便使用
- 例如若要切換至家目錄，可執行指令 cd ~



系統重要目錄

目錄名稱	說明
/	根目錄
/home/	各個帳號家目錄的集中地
/root/	root帳號的家目錄
/bin/	存放系統中最常用的基本使用者指令
/sbin/	僅供系統管理者執行的指令
/lib/	存放基本的共用函式庫檔案及核心模組驅動
/usr/	全文為 Unix Software Resource, 目錄內包含系統的主要程序、圖形介面所需的檔案、自行安裝的軟體、系統文件等
/tmp/	存放暫存檔案, 所有使用者對該目錄都具有讀寫權限

系統重要目錄

目錄名稱	說明
/dev/	所有裝置與設備，包括鍵盤滑鼠、硬碟、USB等，皆以檔案或目錄方式存在於此目錄中，以便於系統管理
/etc/	存放系統的設定檔案，例如使用者帳戶資訊、主機名稱等
/var/	存放長度可變的檔案，例如伺服器的記錄檔案(log)
/sys/	內含檔案用來檢視及設定系統硬體資訊
/mnt/	光碟或其他媒體預設掛載點(mount point)的地方
/opt/	作為optional檔案和程式的存放目錄，例如非預設安裝的外來軟體常會安裝於此

- 指令、路徑、檔案及目錄名稱一律區分大小寫
- 以 / 閔隔上層目錄與下層目錄
- Linux檔案常不使用附檔名，尤其是執行檔
- 系統會利用屬性(metadata)來記錄檔案類型、權限、修改時間，及檔案大小等，而非依賴副檔名
- 目錄路徑的尾端可加上 / 以輔助(人)辨識
- 路徑範例：
 - /
 - /tmp/
 - /home/user1/file1.txt



絕對路徑 vs. 相對路徑

- 由根目錄 / 為起點，一直寫到目的地
- 範例：
 - /
 - /tmp/
 - /home/user1/
 - /home/user1/f1.txt
 - ~/f1.txt

- 以當前目錄為起點，一直寫到目的地 (一定不是由 / 寫起)
- 以 . 表示當前目錄
- 以 .. 表示上一層目錄
- 範例：
 - test/
 - file1.txt 或 ./file1.txt
 - ../
 - ../../tmp/file2.txt

- 安裝CentOS7時預設的檔案系統格式為?
 1. Fat32
 2. XFS
 3. EXT4
 4. NTFS
- 下列何者是正確的Linux路徑?
 1. /temp/
 2. /user/
 3. ~
 4. C:/windows/
- 帳號user1的家目錄為?
 1. /
 2. /home/
 3. /user1/
 4. /home/user1/

Module 7. 基本指令介紹

- 7-1: 當前資訊查詢(pwd/whoami)
- 7-2: 基本指令操作(ls/cd/Tab/history)
- 7-3: 列印、清除(echo/clear/ reset)

指令範例1

- whoami #印出我目前是哪個帳號
- pwd #印出當前所在目錄

```
[user1@localhost ~]$ whoami
user1
[user1@localhost ~]$ pwd
/home/user1
```

檢視目錄或檔案資訊 (ls)

- **ls [目錄或檔案] #list**

- 可用來顯示目錄內容
- 或是用 -l 來顯示檔案屬性

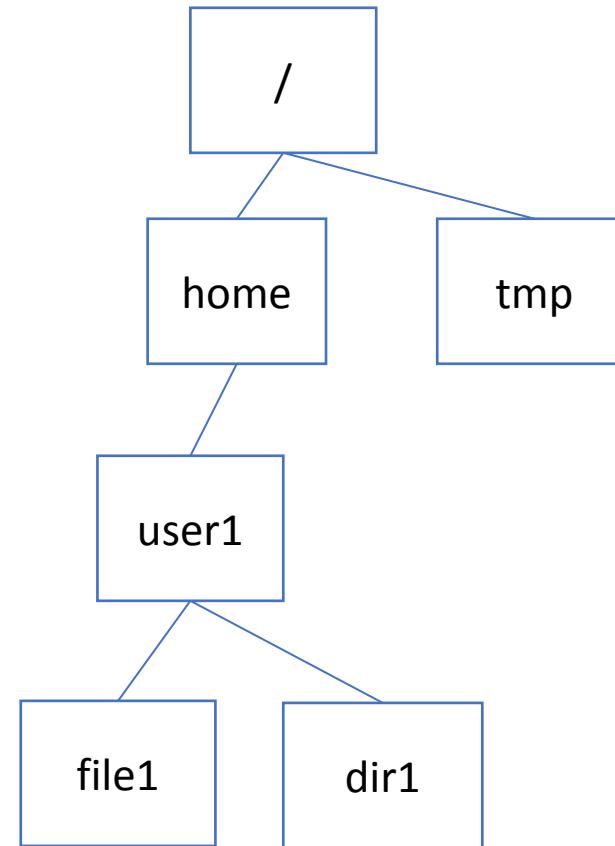
選項	說明
-l	以詳細資訊方式顯示
-d	顯示目錄本身，而非其包含的檔案清單
-a	也顯示隱藏檔 (--all)
-h	以易懂的方式顯示檔案大小
-r	反向排序顯示 (--reverse)
-S	依檔案大小排序顯示
-t	依修改時間排序顯示

指令範例	說明
ls /bin/	顯示/bin/內的檔案(子目錄)清單
ls -l /etc/passwd	顯示該檔案的屬性 (日期、大小等資料)
ls -a -l -h /home/user1	等同 -alh
ls -l -d /home/user1	顯示目錄本身詳細資訊

切換所在目錄 (cd)

- cd 目錄名稱 #Change Directory

指令範例	說明
cd /	切換至根目錄
cd /tmp/	切換至/tmp目錄
cd /usr/lib/	切換至/usr/lib目錄
cd /home/user1/	切換至/home/user1目錄
cd ~ 或 cd	切換至帳號的家目錄
cd ..	切換至上層目錄
cd ../../	切換至上上層目錄
cd -	切換至上次所在的路徑



- 方向鍵 ↑ 顯示上一筆指令操作紀錄
- 方向鍵 ↓ 顯示下一筆指令操作紀錄
- Tab按鍵
 - 自動補齊目前正在輸入的單字，可加快指令輸入速度
 - 這單字可能是檔案、帳號、變數、主機名稱、或指令等
 - 若無法自動補齊，可按2次Tab，會列出候選字

```
[user1@localhost ~]$ ls /etc/init ← 按2次Tab
init.d/  inittab
[user1@localhost ~]$ ls /etc/init█
```

查詢指令操作歷史紀錄 (history)

- **history [N]**

- 當我們以bash登入主機後, bash會主動由帳號家目錄的`~/.bash_history`讀取以前此帳號曾經下過的指令, 並於帳號登出時寫入檔案
- 預設上限1000筆

指令範例	說明
<code>history</code>	列出所有指令操作紀錄
<code>history 5</code>	列出最近5筆指令操作紀錄
<code>history -c</code>	刪除全部紀錄
<code>history -d 2</code>	刪除編號2號的指令紀錄
<code>!!</code>	執行上一個指令 (相當於按↑按鍵後並執行)
<code>!wh</code>	執行上一個名稱為wh開頭的歷史指令
<code>!5</code>	執行編號第5個指令紀錄
<code>!-6</code>	執行倒數第6個指令紀錄

- echo 字串
 - echo test #印出test
 - echo -n test2 #印出test後不換行
 - echo \$PATH #印出環境變數PATH的值
- 字串可用引號標注
 - echo "hello world" #依字串原樣輸出
 - echo hello world #視為2個字串，以一個空格區隔
- 若要一次輸出多行，可使用單引號標記範圍
 - echo '
> 12345
> 67890'

- `clear` #清除終端機畫面
 - 過往訊息仍可以用滑鼠往上捲動顯示
- `reset` #重新初始化螢幕
 - 過往訊息無法用滑鼠往上捲動顯示

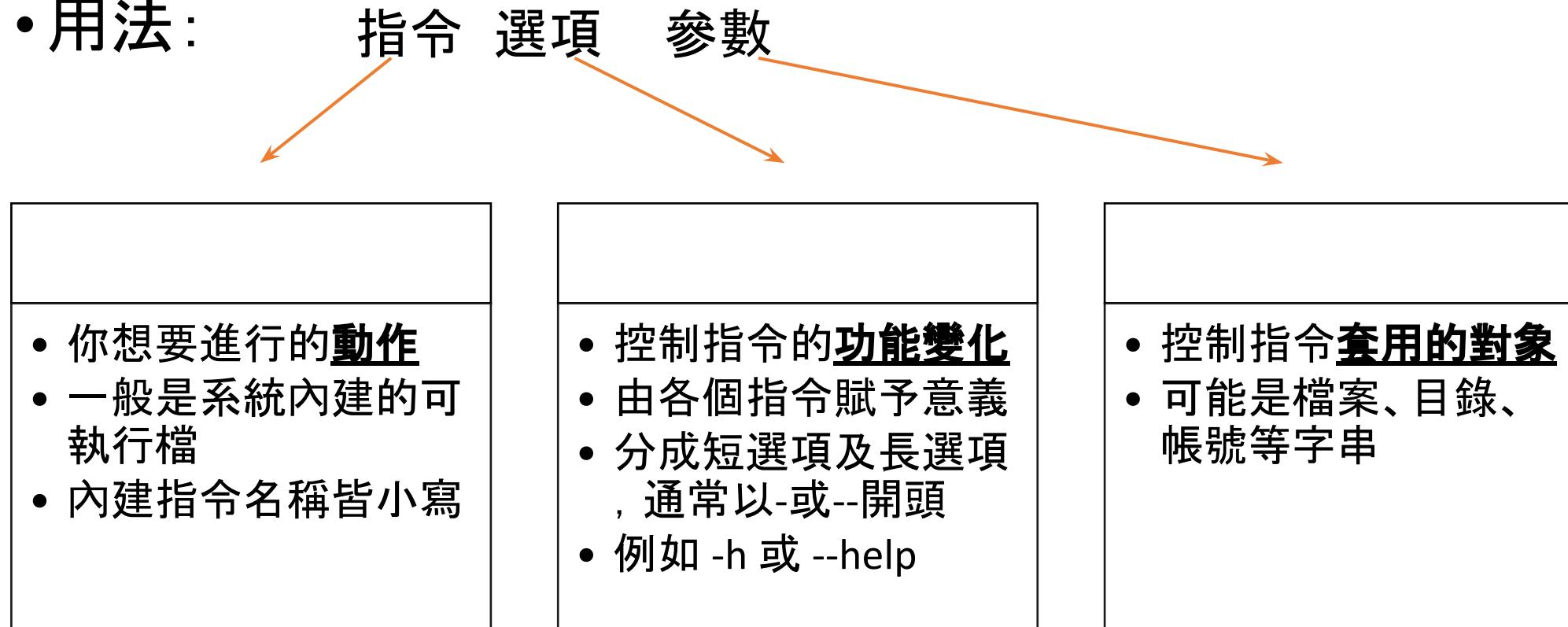


Module 8.

查指令說明

- 8-1: 指令格式
- 8-2: Linux的手冊
- 8-3: Manpage內容

- 大多數的指令都設計為支援0個至多個 “選項”或“參數”
 - 各欄位間以空白或tab作為分隔符號
 - 分隔符號個數不會造成影響
- 每個指令配合不同的選項及參數，可產生多種效果
- 用法：



指令範例2

• cal

- 顯示當月月曆

• cal -m

- 用了-m
- 每周會從星期一起始
- 顯示當月月曆

• cal -m 1 2019

- 每周會從星期一起始
- 顯示2019年1月的月曆

```
[user1@localhost ~]$ cal
         九月 2019
日 一 二 三 四 五 六
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30
```

```
[user1@localhost ~]$ cal -m
         九月 2019
一 二 三 四 五 六 日
1
2 3 4 5 6 7 8
9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
```

```
[user1@localhost ~]$ cal -m 1 2019
        一月 2019
一 二 三 四 五 六 日
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31
```

指令範例3

- **ls**
 - 顯示當前目錄內容
- **ls -l**
 - 顯示當前目錄內容
 - 用了-l選項
 - 列出詳細訊息
- **ls -l -h /etc/passwd**
 - 顯示/etc/passwd檔案的屬性
 - 同時使用了-l及-h選項
 - -l 列出詳細內容
 - -h 以易懂的方式呈現檔案大小

```
[user1@localhost ~]$ ls  
下載 公共 圖片 影片 文件 桌面 模板 音樂
```

```
[user1@localhost ~]$ ls -l  
總計 0  
drwxr-xr-x. 2 user1 user1 6 9月 2 14:04 下載  
drwxr-xr-x. 2 user1 user1 6 9月 2 14:04 公共  
drwxr-xr-x. 2 user1 user1 100 9月 2 14:09 圖片  
drwxr-xr-x. 2 user1 user1 6 9月 2 14:04 影片  
drwxr-xr-x. 2 user1 user1 6 9月 2 14:04 文件  
drwxr-xr-x. 2 user1 user1 6 9月 2 14:04 桌面  
drwxr-xr-x. 2 user1 user1 6 9月 2 14:04 模板  
drwxr-xr-x. 2 user1 user1 6 9月 2 14:04 音樂
```

```
[user1@localhost ~]$ ls -l -h /etc/passwd  
-rw-r--r--. 1 root root 2.1K 9月 2 12:38 /etc/passwd
```

- **man “指令名”**
 - Linux內建的操作手冊
- **info “指令名”**
 - 顯示指令說明
- **whatis “指令名”**
 - 簡潔說明
- **“指令名” --help**
 - 指令自帶的help選項

- man “指令名稱” #可查看指令用法(manual)
 - [] 表示該欄位可以0個或1個
 - ...表示該欄位可以1個以上
- 操作方式
 - h 顯示說明
 - PageUp
 - PageDown
 - / 進行字串搜尋
 - n 往下搜尋
 - N 往上搜尋
 - q 結束離開
- 例如 man “ls”

LS(1)	User Commands
NAME	ls - list directory contents
SYNOPSIS	ls [OPTION]... [FILE]...
DESCRIPTION	<p>List information about the FILEs (the current directory by default). Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.</p> <p>Mandatory arguments to long options are mandatory for short options too.</p> <p>-a, --all do not ignore entries starting with .</p> <p>-A, --almost-all do not list implied . and ..</p> <p>....以下省略...</p>

- 萬一我不知道該查「哪個」指令?
 - 使用關鍵字片段搜尋所有man手冊，以找出可能的指令
- man -k “關鍵字”
- apropos “關鍵字”

我想找出跟系統重開機有關的指令?

```
[ user1@localhost ~]$ man -k reboot
grub2-reboot (8)      - Set the default boot menu entry for the next boot only.
halt (8)               - Halt, power-off or reboot the machine
poweroff (8)           - Halt, power-off or reboot the machine
reboot (2)              - reboot or enable/disable Ctrl-Alt-Del
reboot (8)              - Halt, power-off or reboot the machine
rpc. sm-notify (8)      - send reboot notifications to NFS peers
shutdown (8)            - Halt, power-off or reboot the machine
sm-notify (8)           - send reboot notifications to NFS peers
systemd-reboot.service (8) - System shutdown logic
```

- 依指令性質分為9類
- 數字是用來讓人快速知道該指令的種類

編號	說明
1	使用者在shell環境中可以操作的指令或可執行檔
2	系統核心可呼叫的函數與工具等
3	一些常用的函數與函式庫
4	裝置檔案的說明, 通常在/dev下的檔案
5	設定檔, 或者是某些檔案的格式
6	遊戲
7	慣例與協定等, 例如檔案系統、網路協定
8	系統管理員可用的管理指令
9	跟核心有關的文件

指令說明(info 及 whatis)

- info “指令名稱”

- 結果較多文字型敘述
- 例如 :info “ls”

10.1 `ls': List directory contents

=====

The `ls' program lists information about files (of any type, including directories). Options and file arguments can be intermixed arbitrarily, as usual.
...以下省略...

- whatis “指令名稱”

- 結果非常簡潔
- 例如 :whatis “ls”

ls (1) - list directory contents

• 指令 --help

- 多數(但非全部)指令皆提供選項可檢視該指令之用法
- 例如 :ls --help

```
[user1@localhost ~]$ ls --help
用法 :ls [選項]... [檔案]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.
Mandatory arguments to long options are mandatory for short
options too.
-a, --all          do not ignore entries starting with .
-A, --almost-all   do not list implied . and ..
...以下省略...
```



Module 9. 輔助工具

- 9-1: 補助工具安裝(tmux tree)
- 9-2: tree
- 9-3: tmux

- 在Linux安裝兩個輔助工具, tree 、tmux
 - 需要連到Linux套件伺服器, 安裝前先確認CentOS以連上網路
- 安裝步驟：
 - 在命令列打su
 - 輸入root密碼
 - 執行 yum -y install tree tmux
 - 執行 exit

```
檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)
[user1@centos7 ~]$ su
密碼：
[root@centos7 user1]# yum install -y tmux tree
```

```
=====
Package           Arch    Version      Repository  Size
=====
Installing:
tmux             x86_64  1.8-4.el7   base       243 k
tree              x86_64  1.6.0-10.el7 base       46 k

Transaction Summary
=====
Install 2 Packages

Total download size: 289 k
Installed size: 646 k
Downloading packages:
(1/2): tree-1.6.0-10.el7.x86_64.rpm          | 46 kB  00:00:00
(2/2): tmux-1.8-4.el7.x86_64.rpm            | 243 kB  00:00:00
                                                               456 kB/s  289 kB  00:00:00

Total
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : tmux-1.8-4.el7.x86_64          1/2
  Installing : tree-1.6.0-10.el7.x86_64        2/2
  Verifying   : tree-1.6.0-10.el7.x86_64        1/2
  Verifying   : tmux-1.8-4.el7.x86_64          2/2

Installed:
  tmux.x86_64 0:1.8-4.el7                      tree.x86_64 0:1.6.0-10.el7

Complete!
[root@localhost ~]# _
```

tree (檢視目錄工具)

- tree -N [目錄名稱]...
 - 需要加上選項 -N 才能正常顯示中文名稱

選項	說明
-N	正常顯示中文名稱
-a	顯示所有檔案和目錄
-d	顯示目錄名稱而非內容
-s	列出檔案或目錄大小
-L 3	只顯示當前目錄下指定最大深度的目錄結構(例如3層)

- 範例：
 - tree -N
 - tree -N /tmp

#以樹狀圖列出目錄的內容

```
[user1@localhost ~]$ tree -N
.
├── 下載
│   └── demo
│       └── demo.txt
├── 公共
├── 圖片
├── 影片
├── 文件
├── 桌面
├── 模板
└── 音樂
9 directories, 1 file
```

- tmux 是一種終端機管理工具，可以分割視窗、同時開啟多個終端機
- 執行tmux指令時，會建立一個新的 session
 - 在一個 session 中可以建立多個 windows
 - 在同一個 window 又可以分割多個 panes

三個session, 分別有1、3、5個windows

```
user1@centos7:~
```

檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)

(0) + 0: 1 windows
(1) + 1: 3 windows
(2) + mysession: 5 windows (attached)

```
mysession 0: user1@centos7: ~* centos7 23:57 01-Sep-21
```

```
user1@centos7:~
```

檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)

(0) 0: user1@centos7: ~* "centos7"
(1) 1: user1@centos7: ~ "centos7"
(2) 2: user1@centos7: ~ "centos7"
(3) 3: user1@centos7: ~ "centos7"
(4) 4: user1@centos7: ~- "centos7"

```
mysession 0: user1@centos7: ~* centos7 23:58 01-Sep-21
```

第三個會議裡的5個windows

其中一個window裡面開啟了三個panes

```
user1@centos7:~
```

檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)

[user1@centos7 ~]\$

```
[ user1@centos7 ~]$ [ user1@centos7 ~]$
```

```
mysession 0: user1@centos7: ~* centos7 23:58 01-Sep-21
```

系統指令

session外系統指令	
tmux	開啟並進入一個會議(系統自動命名)
tmux new -s <session_name>	開啟並進入一個自己命名的會議
tmux ls	查詢當前有哪些會議
tmux a	重回上一個離開的會議
tmux a -t <session_name>	回到指定(名稱)的會議
session內系統指令	
Ctrl + b -> ?	查詢指令
Ctrl + b -> d	離開會議
Ctrl + b -> s	切換會議

window、pane指令

session內window指令	
Ctrl + b -> c	新增window
Ctrl + b -> &	刪除window
Ctrl + b -> w	打開window列表, 用於切換window
Ctrl + b -> ,	重新命名當前window
session內pane指令	
Ctrl + b -> “	上下分割pane
Ctrl + b -> %	左右分割pane
Ctrl + b -> x	關閉當前pane
Ctrl + b -> z	最大化當前pane, 再重複一次則回覆原pane大小
Ctrl + b -> 方向鍵	切換pane
Ctrl + b + 方向鍵	調整pane(Mac下失去此功能)

Module 10.

基本檔案管理

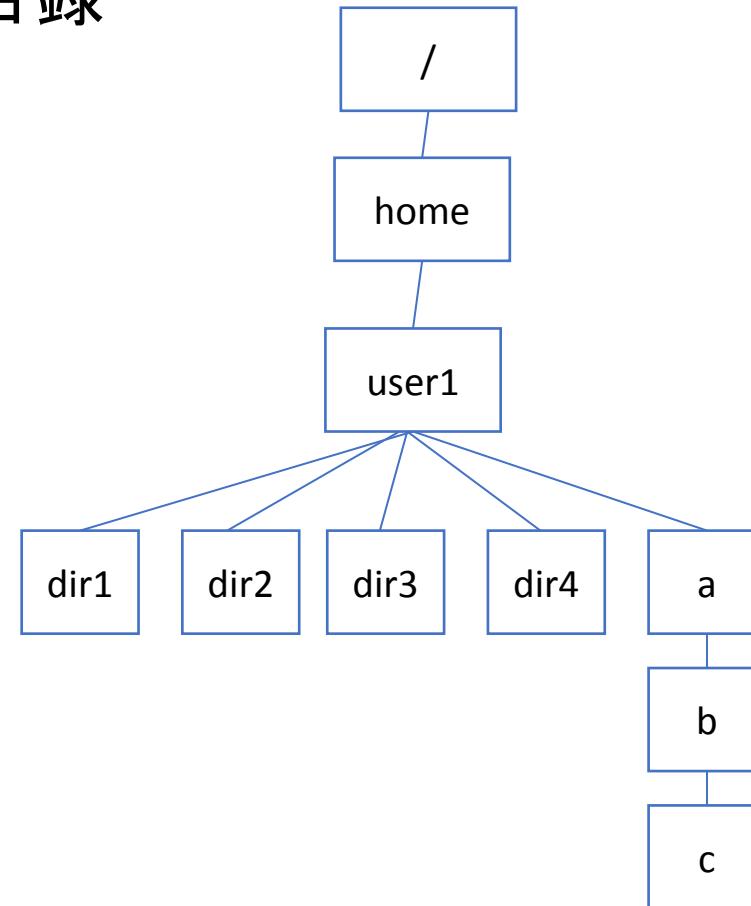
- 10-1: 目錄管理(`mkdir rmdir`)
- 10-2: 檔案複製(`cp`)
- 10-3: 檔案移動與刪除(`mv rm`)

新增目錄mkdir)

- mkdir 目錄名稱 #make directory
 - 不允許建立絕對路徑相同的檔案或目錄

選項	說明
-p	自動產生路徑串中尚未存在的目錄

指令範例	說明
mkdir dir1	於當前目錄內建立dir1
mkdir dir2 dir3	於當前目錄內同時建立2個目錄
mkdir /home/user1/dir4	使用絕對路徑建立目錄
mkdir -p a/b/c/	若a及b及c原本都不存在, 就必須使用-p



刪除空目錄(rmdir)

- rmdir 目錄名稱 #remove directory
 - 非空的目錄就不會被刪除

選項	說明
-p	刪除指定目錄後，若它的上一層目錄也變成空目錄時，就會被刪除

指令範例	說明
rmdir dir1	使用相對路徑刪除空目錄dir1
rmdir /home/user1/dir2	使用絕對路徑刪除空目錄dir2
rmdir dir3 dir4	同時刪除空目錄dir3與dir4
rmdir -p a/b/c	一次刪除階層式空目錄(a , b , c)
rmdir d1/d2/d3	只會刪掉空目錄d3
rmdir *	刪除當前目錄下的所有空目錄

新增檔案(touch) / 修改時間戳記

- touch 檔案名稱

- 不允許建立絕對路徑相同的檔案或目錄



指令範例	說明
touch file1	新增file1
touch file2 file3 /tmp/file4	同時新增多個檔案
touch {1..3}.txt	新增1.txt 2.txt 3.txt

- touch也可用來更改檔案(目錄)被存取或修改的時戳 (timestamp)



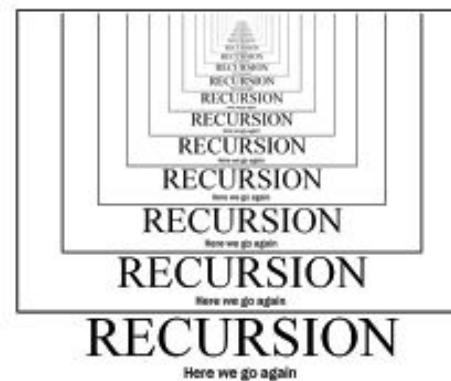
指令範例	說明
touch -t 201401020304 file1	將檔案時戳修改成 2014/01/02 3:04
touch file1	將檔案時戳改成當前時間
touch dir1	將目錄時戳改成當前時間

刪除檔案或目錄(rm)

- rm 檔案(目錄)名稱 #remove

選項	說明
-i	互動模式(interactive), 在刪除前會向使用者確認(y/n?)
-f	強制刪除(force), 意即不會向使用者確認
-r 或 -R 或 --recursive	遞迴刪除, 必須用遞迴方式才能刪除其子目錄及檔案

指令範例	說明
rm file1	將file1刪除
rm -r dir1	將dir1目錄完整刪除



複製檔案或目錄(cp)

- cp 來源... 目標 #copy
 - 可使用相對路徑或絕對路徑
 - 可以有多個來源(檔案或目錄), 但此時目標只能是目錄

選項	意義
-r	遞迴(recursive)複製, 用於目錄的複製
-a	相當於 -p -d -r 一起併用 (常用)
-p	連同檔案的屬性一起複製過去, 而非使用預設屬性(備份常用)
-d	若來源檔為連結檔的屬性(link), 則複製連結檔屬性而非檔案本身
-i	若目標檔已經存在時, 複製前會向使用者確認
-u	更新(update), 若目標時間比較舊或不存在時, 才會進行複製

cp範例

指令範例	意義
cp file1 file2	將file1複製成file2
cp -r dir1 dir2	若目錄dir2不存在, 就將dir1複製為dir2, 若目錄dir2已存在, 就把dir1複製到dir2內
cp file1 /tmp/	複製file1到/tmp/ 內
cp file1 /tmp/file2	複製file1到/tmp/ 內, 並命名為file2
cp -a dir1/ /tmp/	把dir1複製到/tmp/ 內, 並保留屬性
cp file1 file2 /tmp/	把file1及file2複製到/tmp/ 內
cp -u file1 file3	當file3不存在或是比file1舊的時後才會進行複製

移動檔案或目錄(mv)

- mv 來源 目標 #move
 - 可使用相對路徑或絕對路徑
 - 可以有多個來源，但此時目標只能是目錄
 - 可用也於更改名稱

選項	意義
-i	若目標檔已經存在時，移動前會向使用者確認
-f	強制直接覆寫
-u	更新(update)，若目標時間比較舊才會覆寫
-n	不會覆寫已存在的檔案

mv範例

範例	意義
<code>mv file1 file2</code>	將file1移動成file2 (所以也有更名的效果)
<code>mv dir1 dir2</code>	若dir2目錄不存在，則將dir1更名為dir2 若dir2目錄已存在，就把dir1移動至dir2/內
<code>mv file1 dir1/</code>	將file1移動到dir1目錄下\
<code>mv file1 dir1/file2</code>	將file1移動至dir1目錄下，並更名成file2
<code>mv * ../</code>	移動當前目錄下的所有檔案(及子目錄)到上一級目錄
<code>mv -n file1 dir1/</code>	不會覆寫已存在的檔案
<code>mv /tmp/file2 .</code>	將/tmp/file2移動到當前目錄下

- 將檔案或目錄名稱前面加上 . 就可將其隱藏
- 例如將file1.txt改名為 .file1.txt 就變成隱藏檔
 - mv file1.txt .file1.txt
 - ls #看不到隱藏檔/目錄
 - ls -a #看到所有檔案/目錄
 - mv .file1.txt file1.txt #將隱藏取消

Module 11. 文字檔處理工具

- 11-1: 文字檔計數(wc)
- 11-2: 文字檔檢視(cat head tail)
- 11-3: 文字檔瀏覽(less)

- **wc [檔案] #印出行數, 字數, 字元數**

- `wc ~/.bashrc`

- 107 498 3486 /home/user1/.bashrc

- **wc #從鍵盤輸入做計算**

- 輸入資料後, 同時按下 Ctrl 及 d 按鍵, 作為結束符號

選項	意義
-l	計算行數 (line)
-w	計算字數 (word)
-c	計算字元數 (byte)

向Bash shell送出結束訊號

- Ctrl c
 - 送出中斷信號給目前正在執行中的指令
 - 視為非正常終止
- Ctrl d
 - 送出一個EOF(End of File)字元給正在執行中的指令
 - 通常用來終止操控台的輸入
 - 視為正常的終止

- cat 檔案 #concatenate

指令範例	說明
cat file1	顯示file1內容
cat -b file1	會顯示行號, 僅針對非空白行做行號顯示
cat -n file1	會顯示行號, 連同空白行也會有行號

- cat也可以用來顯示”從鍵盤輸入的內容”

- 執行cat, 然後輸入字串(可多行), 按下Ctrl d 送出結束符號

- tac 檔案 #反向顯示

- tac file1

- nl 檔案 #顯示行號 (效果同cat -b)

- nl file1

顯示檔案頭尾內容(head / tail)

- head 檔案 #顯示檔案指定的前面行數

指令範例	說明
head file1	預設會顯示前10行
head -n 20 file1	顯示前20行
head -n -40 file1	顯示除了最後40行外的所有行

- tail 檔案 #顯示檔案的最後幾行

指令範例	說明
tail file1	預設顯示末10行
tail -n 20 file1	顯示末20行
tail -n +50 file1	顯示除開頭50行外的所有行數

分頁顯示檔案內容 (more / less)

- more 檔案
- less 檔案
- 範例：
 - less /etc/passwd

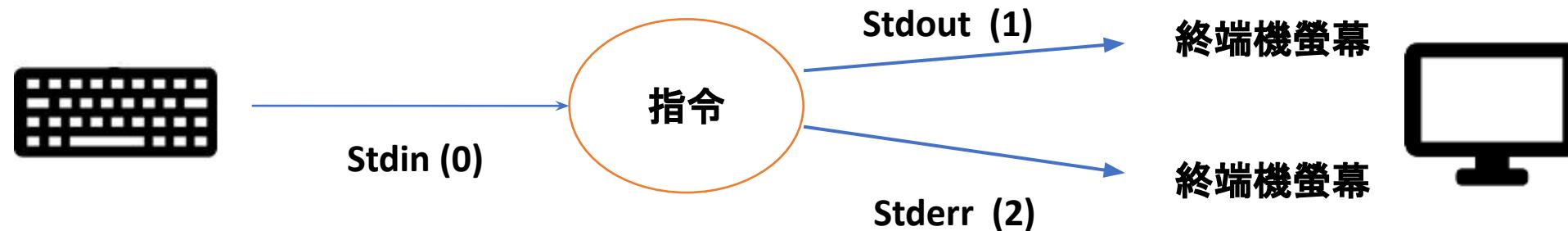
less新增
的功能

按鍵	說明
Enter	向下翻一行
空白鍵	向下翻一頁
b	往回翻一頁
:f	顯示出檔名及目前行數
q	立刻結束
PageUp	向上(下)翻一頁
PageDown	向下翻一頁
/ 字串	往後搜尋字串
? 字串	往前搜尋字串
n	重複前一個搜尋

Module 12. I/O重導與管線

- 12-1: 輸出重導(> >>)
- 12-2: 輸入導向(<)
- 12-3: 管線(|)

Linux 輸出資料流



Handle 代號	名稱	描述	預設裝置	用途
0	stdin	標準輸入	鍵盤	大多數指令可從stdin讀取輸入，並將輸出寫到stdout
1	stdout	標準輸出	終端機螢幕	
2	stderr	標準錯誤輸出	終端機螢幕	專供用來輸出錯誤訊息

- 可將stdout與stderr的資料流重新導向，分開處理
 - 例如將stdout記錄於檔案內，而stderr顯示在螢幕
 - 或是將stderr導向至/dev/null，可丟棄錯誤訊息

資料流導出符號

符號	說明
>	重導stdout, <u>覆蓋掉</u> 原始檔案內容
>>	重導stdout, <u>附加在</u> 原始檔案內容後端
2>	重導stderr, <u>覆蓋掉</u> 原始檔案內容
2>>	重導stderr, <u>附加在</u> 原始檔案內容後端
&>	重導stderr與stdout, <u>並覆蓋掉</u> 原始檔案內容
&>>	重導stderr與stdout, <u>附加在</u> 原始檔案內容之後

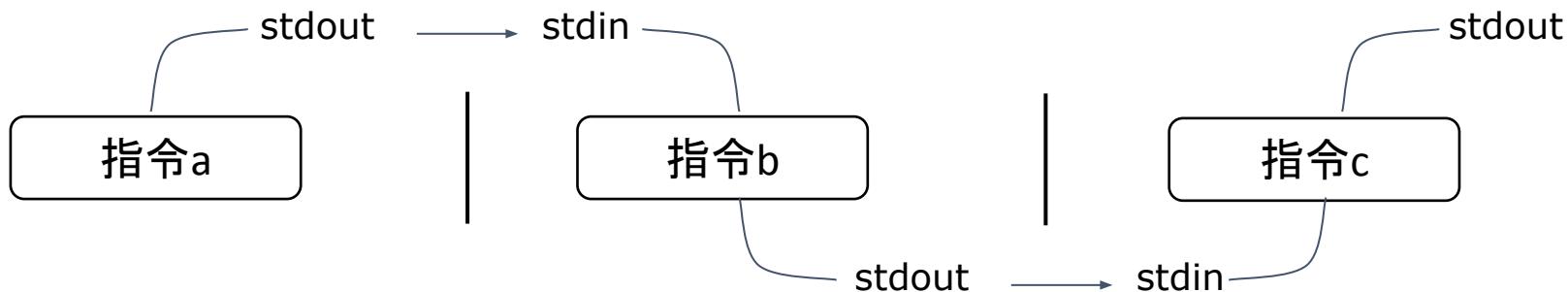
資料流導出符號範例

範例	說明
history > a.txt	將歷史紀錄寫入a.txt
head /etc/passwd > b.txt	將前10行內容寫入b.txt
cat a.txt b.txt > c.txt	將a及b檔案的內容依序寫入c.txt
echo "Mary" >> login.txt	login.txt檔案最後一行新增“Mary”
指令 2> /dev/null	錯誤訊息導向至/dev/null
指令 &> log.txt	正確及錯誤結果都會寫入log.txt
指令 2>> error.txt	錯誤訊息附加至error.txt
指令 &>> log.txt	正確及錯誤結果都會附加至log.txt
指令 >correct.log 2>error.log	兩種結果分開儲存

- `wc < file1.txt`
 - 以file1.txt取代stdin作為資料來源
 - 但實際上，直接使用 `wc file1.txt` 即可(以參數指定來源)

管線(pipe)

- 把前一個指令的stdout拿來當做後一個指令的stdin
- 指令a | 指令b | 指令c ... 以此類推
- 範例：
 - history | less #把history的結果進行分頁顯示
 - head /etc/passwd | wc #計算前10行的行數/字數

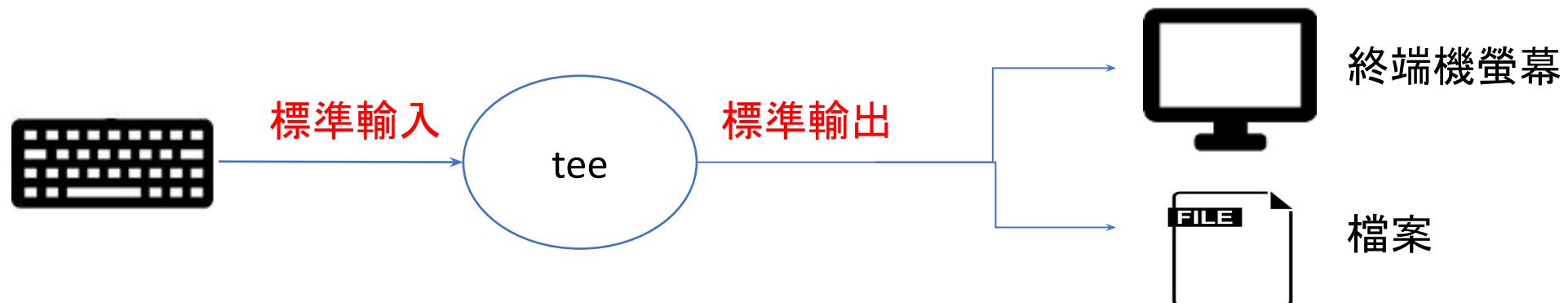


管線(pipe)的使用限制

- 管線後面接的指令必須能接收 stdin
 - less,head,wc,grep 等處理文字的指令可以放在管線後面
 - ls ,cp, cd等則不能放在管線後面
- 只能將stdout往後面指令丟，會忽略stderr
 - 所以若前面指令只產生錯誤結果，後面指令收到的stdout就會是空白
 - 例如: ls -l “錯誤路徑” | wc -l
 - wc收到空白資料，故計算結果為 0

雙重導向(tee)

- tee 檔案 #同時將資料流分送到螢幕以及檔案
 - -a 表示append, 以累加方式寫入檔案尾端



- 範例：
 - echo hello | tee log.txt #螢幕及檔案都有hello
 - echo hello2 | tee -a log.txt #將hello2附加到log.txt
 - ls -l | tee list.txt | less #結果寫到檔案, 而螢幕分頁顯示

`指令`

• 嵌入指令

\$(指令)

• 嵌入指令

- 嵌入的指令會優先執行，將執行結果拿來使用
 - `叫做反單引號 (就是跟~同位置的那個按鍵)`
- 範例：
 - echo "Today is `date` ,Hello" #先執行date再執行echo
 - echo "Today is \$(date) ,Hello" #先執行date再執行echo
 - Today is Wed Dec 21 04:40:54 CET 2016,Hello

Module 13.

資料串流指令

- 13-1: 字元轉換(tr)
- 13-2: 篩選資料(cut)
- 13-3: 排序與唯一(sort uniq)

- 可以用來進行文字訊息的替換或是刪除一段訊息當中的文字
 - -d : 刪除訊息當中的SET1這個字串
 - -s : 取代重複的字元
- 範例
 - 將所有的小寫變成大寫
 - cat /etc/passwd | tr '[a-z]' '[A-Z]'
 - 將冒號 (:) 刪除
 - cat /etc/passwd | tr -d ":"
 - 將字元'o' 取代成字元'@', 遇到連續的o只會轉成1個@
 - cat /etc/passwd | tr -s 'o' '@'

- seq 起始值 [累加值] 結束值
- 範例
 - seq 1 10
 - 產生1,2,3,...,10
 - seq 1 2 11
 - 產生1,3,5,7,9,11
 - seq -w 1 2 11
 - 產生 01,03,05,07,09,11 (對齊)

排序(sort)

- sort 檔案

- 將檔案中的資料排序，並在螢幕顯示結果
- 預設視為**字串形態**，比較字元的ASCII碼順序

選項	說明
-r	反向(由大到小)排序
-n	以數值由小到大排序，例如20會排在132的前面
-M	以月份排序，例如 Aug, DEC 等等的排序方法
-f	忽略大小寫的差異，例如 A 與 a 視為編碼相同
-u	排序並過濾重複的資料
-t 分隔字元	指定排序時所用的欄位元分隔字元
-k 數字	以第幾個欄位來排序

指令範例	說明
sort file1	字典順序，100 會排在94前面
sort -n file1	數值順序，94 會排在100前面
sort -n -k 2 file1	以第2筆欄位數值排序

file1

Curry 94
 Lin 74
 Paul 93
 Kobe 84
 Jordan 100

單一(uniq)

- uniq 可以將相鄰重複的文字整行刪除，留下不重複的資料
- 未相鄰的行就無法處理，所以一般會進行排序再處理

```
[ user1@localhost ~]$ cat test.txt
Mary
hello world
hello world
Mary
Mary
[ user1@localhost ~]$ uniq test.txt
Mary
hello world
Mary
```

檔案分割(split) 與合併(cat)

- 先產生範例檔案

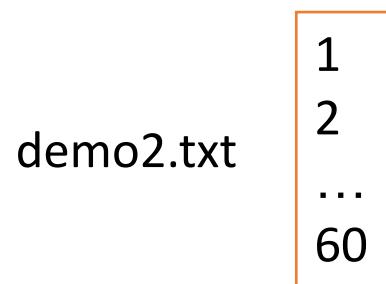
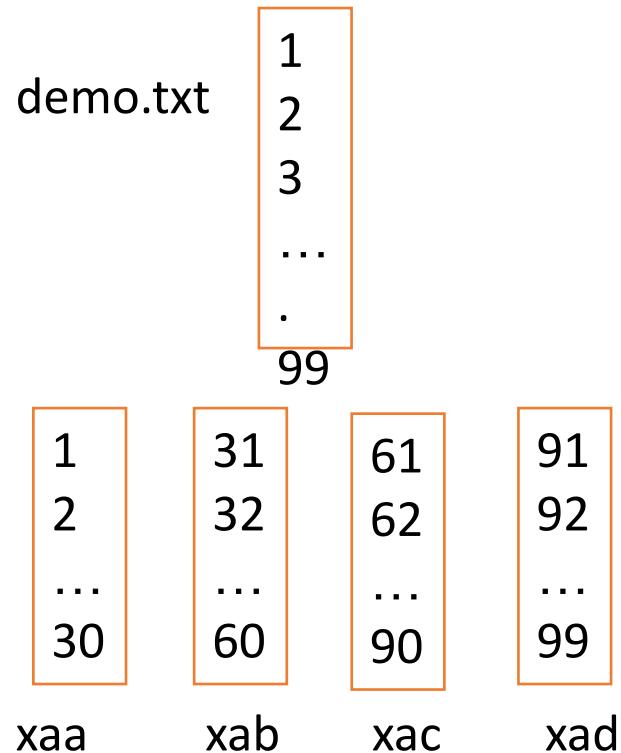
- seq 1 99 > demo.txt

- 對檔案分割

- split -l 30 demo.txt
 - 每30行切分為一個小檔案,
 - 依序產生xaa、xab、xac、及xad

- 檔案內容上下合併

- cat xaa xab > demo2.txt



檔案內容左右合併 (paste / join)

- **paste**

- 結果預設會以tab隔開
- -d 指定結果的分隔符號

a.txt

```
Tom 100
Mary 89
Jojo 60
```

b.txt

```
Tom A
Mary B
Jojo D
Jan F
```

- **join**

- 將兩個檔案中，有相同資料的那一行合併在一起
- 若原始檔案有很多欄位，則會比對第1筆欄位的資料是否相同
 - 預設以空白字元作為分隔符號
 - -t 指定其他分隔符號

```
[user1@localhost ~]$ paste a.txt b.txt
Tom 100 Tom A
Mary 89 Mary B
Jojo 60 Jojo D
Jan F
[user1@localhost ~]$ join a.txt b.txt
Tom 100 A
Mary 89 B
Jojo 60 D
```

剪切訊息的內容(cut)

- cut可將一行文字當中，取出想要的部分

選項	說明
-c 字元	依字元抓取
-f 數字N	取出第N筆欄位，左起由1起算
-d 分隔符號	分隔符號可以用任何字元，預設是TAB符號。 但若分隔符號為空白時，則可用單引號包起來，例如 -d ''

指令範例	說明
ls -l cut -c 1-10	抓取第1至第10個字元
ls -l cut -d '空白鍵' -f 3	以空白分隔，抓取第3欄
echo \$PATH cut -d : -f 2	以:為分隔，抓取第2欄
cut -d : -f 5 /etc/passwd	抓取檔案第5欄
cut -d : -f 1,3 /etc/passwd	抓取檔案第1欄及第3欄
cut -d : -f 2-4 /etc/passwd	抓取檔案第2,3,4欄

Module 14. 檔案搜尋

- 14-1: 尋找執行檔(which)
- 14-2: 萬用字元
- 14-3: 尋找文件(find)

系統如何尋找執行檔？

- Linux利用環境變數PATH來記錄指令的所在目錄清單
- 由多個目錄組成，以:區隔
- 執行 echo \$PATH 可以檢視內容
 - 例如/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin
- 依序在各目錄內搜尋可執行檔(指令)，一旦找到後就不
再繼續找

顯示指令位置 (which / whereis)

- which “指令” #顯示指令的位置, 只在\$PATH內搜尋
 - which cp 顯示 /usr/bin/cp
- whereis “指令” #搜尋二進位檔案、原始碼或說明文件
 - whereis cp
 - cp: /usr/bin/cp /usr/share/man/man1/cp.1.gz
/usr/share/man/man1p/cp.1p.gz

- 也可執行「man 7 glob」查詢用法

字元	功能	範例
*	比對0個以上任意字元	c* 表示以c開頭的所有檔案/目錄
?	比對恰有一個任意字元	c?? 表示以c開頭，且名稱長度為3個字元
[]	比對一個括號內指定的字元	file[159] 表示file1或file5或file9
[-]	比對在編碼順序範圍內的一個字元	[0-9] 表示0至9之間任一個字元 [b-d]* 表示開頭為b或c或d的所有檔案/目錄
[^]	反向選擇 (黑名單)	[^abc]代表非a, b, c的其他任一個字元
{ , }	以逗號為間隔進行列舉	cp {*.txt, *.pdf} /tmp 表示將pdf及txt複製到/tmp touch {1..200}.txt 表示產生檔案1.txt ~ 200.txt

Shell萬用字元範例

指令範例	說明
<code>ls /bin/h*t</code>	列出/bin/目錄內h開頭,t結尾的檔案或目錄
<code>ls /bin/y*</code>	列出/bin/目錄內y開頭的檔案或目錄
<code>ls /bin/*[0-9]*</code>	列出/bin/目錄內名稱包含數字的檔案或目錄
<code>ls /bin/[^a-x]??</code>	列出/bin/目錄內非a至x開頭, 3個字元的檔案或目錄
<code>ls *.txt</code>	列出當前目錄下附檔名為txt

- type “指令”

- type cp 顯示 cp is /usr/bin/cp
- type ll 顯示 ll is aliased to 'ls -l --color=auto'
- type cd 顯示 cd is a shell builtin

- file “檔案或目錄”

- file /etc/passwd #文字檔

/etc/passwd: ASCII text

- file /bin/ #目錄

/bin/: directory

- file /bin/cp #執行檔

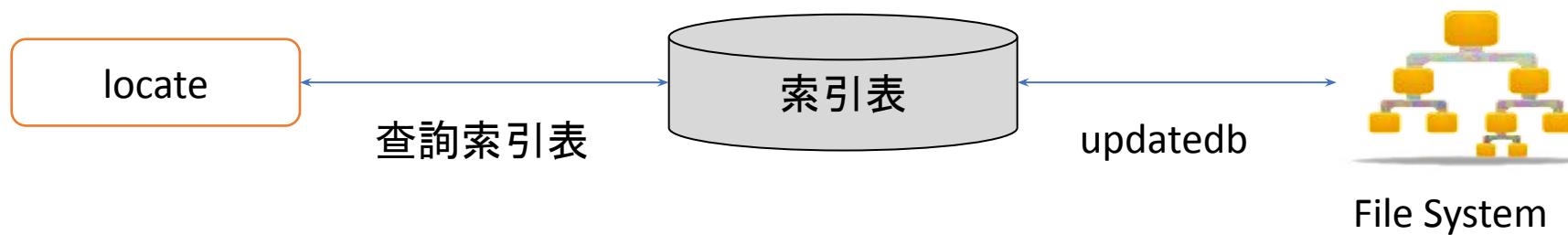
/bin/cp: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.32,
BuildID[sha1]=0xe649cde2b13c057cc6a5574808fb5f026f7b8555, stripped

搜尋檔案或目錄位置(locate)

- locate “**檔案關鍵字**”

- 以“關鍵字”搜尋檔案(或目錄)的路徑名稱
- 用查索引表的方式，比翻硬碟快，但若未更新索引表，搜尋結果可能失真
- 以root身分執行 updatedb

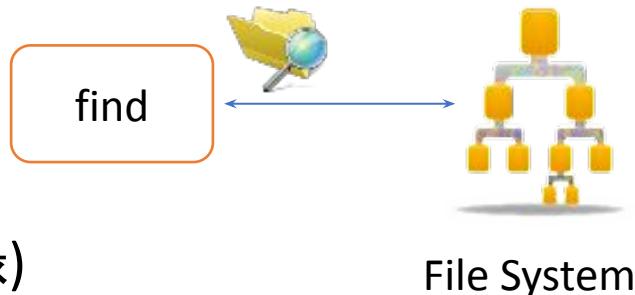
指令範例	說明
locate user1	在絕對路徑中出現user1字串的都會列出
locate -b user1	基本檔名有出現user1字串的才會列出
locate -c user1	只顯示找到的數量
locate “*.img”	找副檔名為img的檔案



尋找檔案位置(find)

- find [搜尋路徑]... [選項]... 字串

- 翻硬碟搜尋檔案，速度較慢
- 搜尋路徑可為多個目錄，以空白分隔
- 若無指定路徑則預設是搜尋**當前目錄**(及子目錄)



根據	用法	說明
名稱	-name 檔名	檔名若是使用萬用字元則須使用引號
使用者	-user 帳號 -group 群組	也可以用 -uid 也可以用 -gid
檔案類型	-type f -type d	f表示為檔案 d表示為目錄
時間	-mtime 4	4天前當天被更動過
	-mtime +4	4天前(不含第4天)內被更動過
	-mtime -4	4天內被更動過
	-newer file1	比 file1 還要新的
檔案大小	-size [+/-]SIZE	c代表byte, k代表KB, M代表MB, G代表GB
	-empty	空的檔案或目錄
搜尋深度	-maxdepth N	進入指定目錄下層目錄時，最深不超過N層

find範例

指令	說明
find /lib/ -name "java"	找出/lib/目錄(及子目錄)內的java
find /lib/ -maxdepth 1 -name "java"	搜尋/lib/目錄(不含子目錄)內的java
find /boot/ /usr/ -name "*.img"	列出/boot/及/usr/目錄內的img檔案
find /bin/ -name "???"	列出/bin/內名稱為2個字元的檔案或目錄
find /boot/ -type d	列出/boot/內所有目錄
find /boot/ -user root -type f	列出/boot/屬於root的檔案
find /bin/ -size +500k	列出/bin/內大於500KB的檔案或目錄
find /bin/ -size -5c	列出/bin/內小於5 byte的檔案或目錄
find /boot/ -type f -size +2M	列出/boot/內大於2MB的檔案
find ~ -newer /etc/passwd	列出家目錄內比/etc/passwd新的檔案或目錄
find . -empty -type f	列出當前目錄下的空檔案

find 配合 exec

- 利用find找出大量目標後，想要直接套用某個指令？
- find /path/ -name “filename” -exec 指令 {} \;
 - {} 表示所找到的檔案或目錄
- 範例：
 - 找出所有大於500MB的檔案並刪除
 - find / -size +500M -exec rm {} \;
 - 將 /tmp(及子目錄)內找到的*.txt複製到家目錄內
 - find /tmp/ -type f -name “*.txt” -exec cp {} ~ \;
 - 後面記得要加上 \; 結束

```
root@kali:~# find . -name "[0-9].txt"
./1.txt
./3.txt
./2.txt
root@kali:~# find . -name "[0-9].txt" -exec rm {}
find: 「-exec」後缺少了參數
root@kali:~# find . -name "[0-9].txt" -exec rm {} \;
```

Module 15.

檔案內容搜尋

- 15-1: grep指令
- 15-2: 搜尋固定字串
- 15-3: 正規表達式

在檔案內搜尋字串(grep)

- grep [選項] 比對字串 [檔案] #預設顯示字串所在整行

選項	說明
-o	只印出相符的字串
-n	印出字串所在的那一行編號
-H	印出字串所在檔名
-r	也搜尋子目錄
-v	反向選擇
-i 或 --ignore-case	不區分大小寫
-l或--file-with-matches	列出檔案內容符合指定的樣式的檔案名稱
-L或--files-without-match	列出檔案內容不符合指定的樣式的檔案名稱
-w	須全字相符
-c或--count	印出具相符結果的總行數
-s或--no-messages	不要顯示錯誤資訊
-m N	只列出前N個
-A 顯示行數	找到字串的後(after)幾行也會顯示出來
-B 顯示行數	找到字串的前(before)幾行也會顯示出來

grep範例

指令範例	說明
grep -n -w "home" /etc/passwd	顯示/etc/passwd檔案內出現home的地方
grep -o "home" /etc/passwd	只印出"home", 不會印出其他字元
grep -c "home" /etc/passwd	顯示/etc/passwd檔案內含有home字串的行數
grep -H "home" /etc/*	找出/etc/目錄下含有home字串的所有檔案
history grep "is"	列出指令記錄內出現is字串記錄
grep -Hn "is" 接著輸入 This is a test 畫面顯示(standard input): 1:this is a test 按Ctrl d 離開	從標準輸入搜尋is字串

- 也稱為正規(則)表式法, regex, regexp
- 使用一些特定符號來定義文字的樣式, 如身分證字號
- 可用來取代規律, 重複的繁瑣動作
- 限制 : 必須是純文字格式
- 比對方式為從左至右
- 搭配文字編輯器或其他文字處理工具做搜尋, 刪除或取代等動作, 例如:
 - vim、sed、awk、grep、grep -E、egrep
 - 搜尋正規表示法時建議用單(雙)引號標註起來
- man 7 regex

延伸正規表
示法才適用

符號	意義	範例
()	界定分組匹配範圍	(Exe)([0-9])-([0-9]) #例如 Exe1-2
(字串1 字串2)	#字串1或字串2	<ul style="list-style-type: none"> • (YES NO) #YES或是NO • g(la oo)d #glad或good
^	樣版出現在行首	(^They)
\$	樣版出現在行尾	(mp3)\$
\<	樣版出現在字首	\<(goo)
\>	樣版出現在字尾	(gle)\>

範例：

```
grep -n “^root” /etc/passwd #只找出未於行首的root字串
```

- 欲得知“root”字串曾在/etc/passwd內出現幾次，可使用哪個指令？
 1. grep -c “root” /etc/passwd
 2. grep --count “root” /etc/passwd
 3. grep -o “root” /etc/passwd | wc -l
 4. grep -n “root” /etc/passwd

Module 16.

延伸正規表達式

- 16-1: 任意字元
- 16-2: 重複字元
- 16-3: 範圍字元

比對「一個字元」的符號

符號	意義	範例
.	任意一個字元 (不可為空字元)	<ul style="list-style-type: none"> r.t 可符合 rat、r2t、r@t，但不符合rt c..y 可符合 copy、ca2y、c!\$y
[]	中括號中任一字元的內容	<ul style="list-style-type: none"> [abz] 表示 a 或 b 或 z 其中一個字元 [!>] 表示 ! 或 >
[-]	表示一組連續的範圍	<ul style="list-style-type: none"> [A-Za-z] 表示一個英文字元 [0-9] 表示一個數字字元 [0-25] 表示可為 0,1,2,5
[^]	排除中括號內的任何一個字元	<ul style="list-style-type: none"> [^13] 表示不能是1或3 [^0-9] 表示不可為數字字元

- 範例：

- grep -E -n “r.t” /etc/passwd
- grep -E -o “h..e” /etc/passwd

#印出符合字串所在的整行
#只顯示符合的字串

- 一律以字串來看待要比對的項目
- [0-255] 並不會比對0~255的數字，而是被視為 0-2,5,5 也就是等同於[0125]

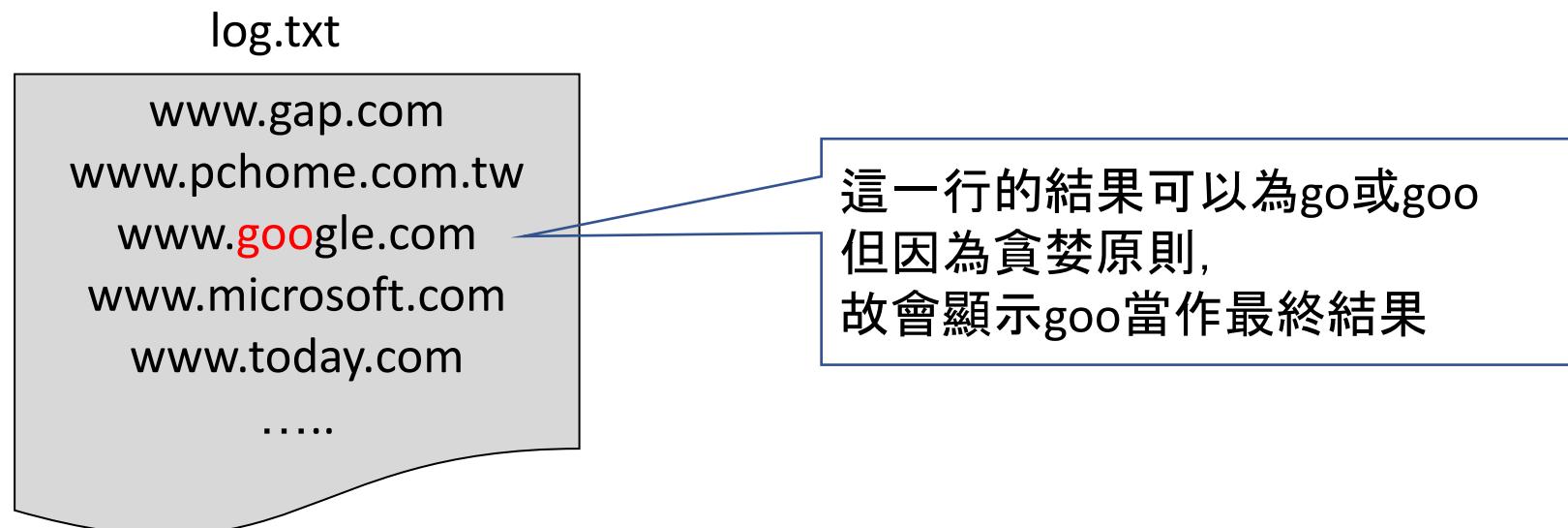
目的	範例
找出數字	<code>egrep '[0-9]' /etc/passwd</code>
	<code>egrep '[:digit:]' /etc/passwd</code>
找出0~9	<code>egrep -w '[0-9]' /etc/passwd</code>
找出0~99	<code>egrep -w '([0-9] ([1-9][0-9]))' /etc/passwd</code>

- 用來以表達前面的「樣版」重複出現的次數
 - 樣版可能是一個字元, 或是由小括弧括起來的一長串

符號	意義	範例
?	樣版出現0次或1次	ra?t 可符合 rt 或 rat
*	樣版出現0次以上	ra*t 可符合 rt 或 rat 或 raat
+	樣板重複出現1次以上	r.+t 可符合 rat、rent、r@t, 但不能rt
{n}	樣板重複出現n次	ba(na){2} 符合banana
{n,m}	樣板重複出現n到m次	go{1,3}gle 符合gogle或google或gooogle

貪婪原則

- 預設是從相符的字串中，挑選最長的當作最終結果
 - 例如若用以下本文搜尋 go+
 - grep -E -o "go+" log.txt



- \ 跳脫符號用來讓後面的字元失去特殊功用，轉為純字元
- 後面可使用 ^.[\${()|*+?{\} \ 等字元
- 範例：
 - 以 \. 表示 . 這個字元
 - 以 \\ 表示 \ 這個字元
- []內不需使用跳脫字元
 - grep -E [a-z.-] ~/.bashrc #找出小寫字母以及.和-

意義字符 (Linux專屬)

- defined by the POSIX standard

符號	意義	效用
[:alnum:]	所有英文字大小寫和數字	等同於[A-Za-z0-9]
[:alpha:]	所有英文字大小寫	等同於[A-Za-z]
[:digit:]	所有數字	等同於[0-9]
[:xdigit:]	代表16進位的數字類型	等同於[0-9A-Fa-f]
[:lower:]	所有英文字小寫	等同於[a-z]
[:upper:]	所有英文字大寫	等同於[A-Z]
[:punct:]	所有標點符號	: “ ‘ ? ! ; : # \$
[:blank:]	空白	空白鍵及Tab鍵
[:space:]	空白	空白鍵及Tab鍵及CR
[:cntrl:]	控制按鍵	包括 CR, LF, Tab, Del.. 等

符號	意義	效用
\w	數字、字母、底線	[A-Za-z0-9_]
\W	\w的反向	[^A-Za-z0-9_]

- 備註：各系統(語言)對Regex的支援度不同
 - 例如在python可用\d代表數字，但Linux grep卻不支援

練習:針對檔案”GPL”搜尋其中內容

- 首先 cd /usr/share/doc/centos-release/

項次	指令	說明
1	grep -inw “license” GPL	印出行數,不分大小寫,全字相符
2	grep -E -o “<.+>” GPL	找出被標籤<>的字串
3	grep -E -n “^you” GPL	句首是you
4	grep -E -n “and\$” GPL	句尾是and
5	grep -E “^o.*and\$” GPL	句首是o, 且句尾是and
6	grep -E -o “..cept” GPL	任意兩個字元跟著字串“cept”，只印出相符字串
7	grep -E -w “(a an)” GPL	找a或an的單字(word)
8	grep -E “[^m]ode” GPL	找.ode, 但第一個字元不可為m
9	grep -E -on “^[A-Z].+” GPL	找句首為大寫字元的整行
10	grep -E -on “^[A-Z].*\.\$” GPL	找句首為大寫字元, 句尾為.的整句
11	grep -E -on “(copy)?right” GPL	找copyright或right
12	grep -E -i “[AEIOU]{3}” GPL	找連續3個母音,不分大小寫
13	grep -E -w -o “[[:alpha:]]{14,18}” GPL	找長度是14~18的英文單字

其他範例

使用者帳號

- [A-Za-z0-9_-.]{3,16}

身分證字號

- [A-Za-z][12][0-9]{8}

電子郵件

- [A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,6}

IPv4位址

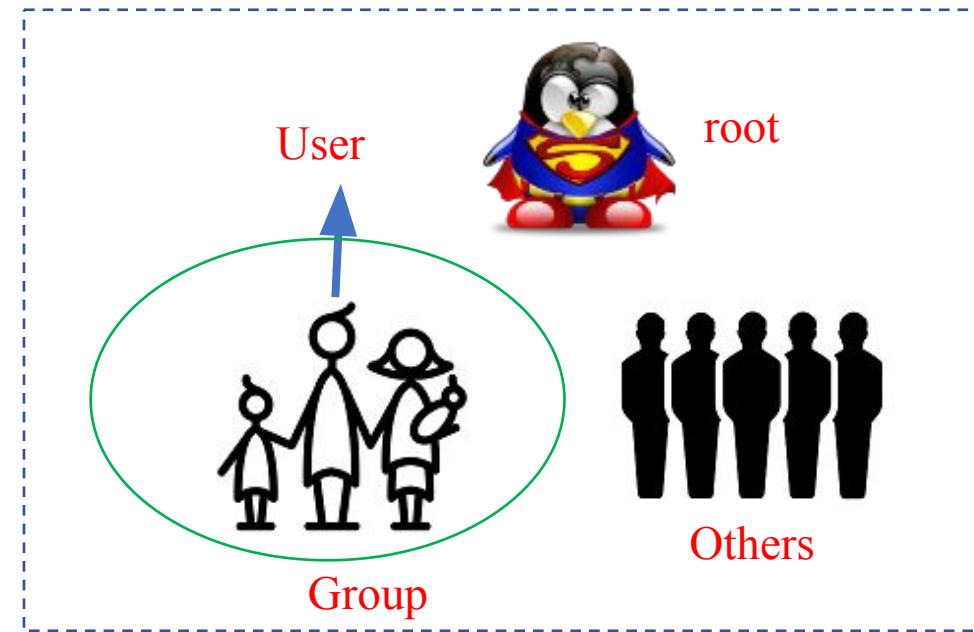
- (([0-9] | [1-9][0-9] | 1[0-9]{2} | 2[0-4][0-9] | 25[0-5])\.){3}(([0-9] | [1-9][0-9] | 1[0-9]{2} | 2[0-4][0-9] | 25[0-5])

- 找出g開頭與g結尾的字串，當中的字元可有可無
 1. g^*g
 2. $g.^*g$
 3. $g.+g$
 4. $g?g$
- 如何搜尋檔案file1.txt中，含有單引號的行？
- 如何搜尋檔案file1.txt中，含有cs,css,csss,... 等字串的行，並列出行號？

Module 17. 檔案目錄權限架構

- 17-1: 檔案權限
- 17-2: 存取類型
- 17-3: 檢視權限

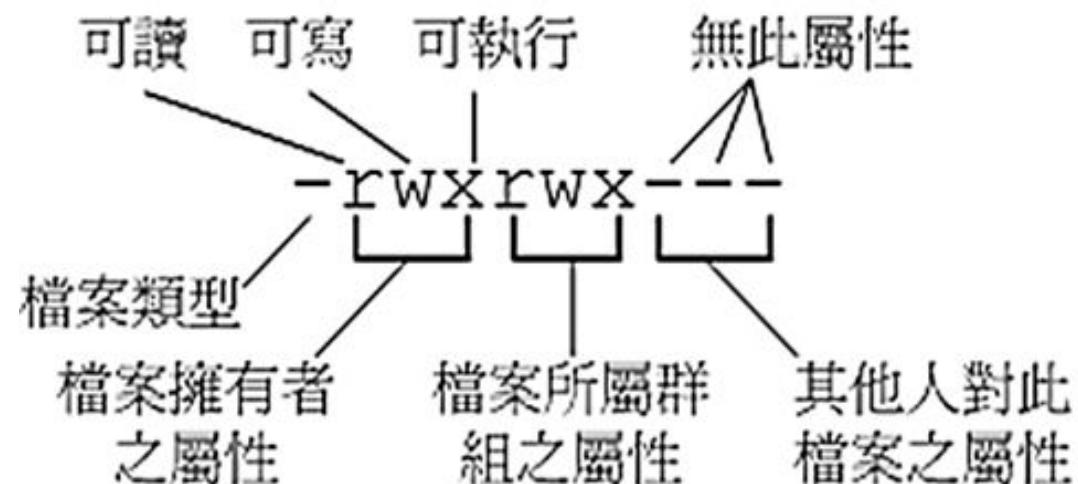
- 每個檔案(目錄)可依不同身分給予相對應的權限
- 身分：
 - 檔案擁用者 (User)
 - 群組成員 (Group)
 - 其他人(Others)
- 權限：
 - 讀 (read)
 - 寫 (write)
 - 執行 (execute)



ls -la 檢視檔案資訊

-rwxrwx---	1	user	group1	3180	2010-03-15	16:59	.bashrc
	連結數	擁有者	群組	檔案大小 (bytes)	檔案最後存取	日期及時間	檔名

- 每個檔案(或目錄)都有權限位元，於建立時由系統給予預設值，擁有者可以進行權限修改
 - 第1個位元代表檔案(目錄)類型
 - 表示為一般檔案
 - d 表示為目錄
 - l 表示為 Soft link
 - 後9個位元為權限位元
 - r 表示read權限
 - w 表示write權限
 - x 表示excute權限



權限說明

權限	對於檔案	對於目錄
讀取 r	<ul style="list-style-type: none">檢視檔案內文cat、tac、less、more、tail、head	<ul style="list-style-type: none">檢視目錄的內容物ls
寫入 w	<ul style="list-style-type: none">修改內容後，可存檔保留變更vim、gedit、joe、nano	<ul style="list-style-type: none">可異動該目錄的內容物(新增/修改/刪除檔案及子目錄)touch、mkdir、rmdir、rm、mv
執行 x	<ul style="list-style-type: none">將檔案當成程式來執行./file1	<ul style="list-style-type: none">允許進入到該目錄內cd

- 需同時具備該目錄的w及x
- 練習：
 - 有個帳號user1，家目錄為/home/user1，目錄權限檢視為

```
drwxr-xr-x 26 user1 user1 4096 Feb 6 19:23 user1
```

- 請在家目錄內建立一個只有root能w的檔案

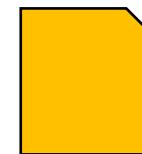
- cd /home/user1/
 - sudo touch rootfile.txt

```
-rw-r--r-- 1 root root 0 Feb 1 19:23 rootfile.txt
```

/home/user1/
(drwxr-xr-x)

- 請問：

- user1可否編輯檔案內容？
 - user1可否刪除檔案？ 
 - 也就是 rm -f rootfile.txt 可以執行成功嗎？



rootfile.txt
(-rw-r--r--)

群組權限的判斷是依據帳號的主要群組

- 假設檔案的 ls -l 顯示結果如下：
 - -rwxrw---- 1 test test 0 Feb 1 19:23 file1.txt

- 若帳號user1的主要群組為user1，次要群組為test
 - user1 對此檔案的權限為 - - -
- 若將帳號user1的主要群組改為test
 - 可使用usermod修改
 - user1對此檔案的權限為 r w -

權限位元範例

- drwxrwx-r-- user1(擁有者) user1(群組) dir1
- 請問以下帳號對此目錄有何權限？

帳號	主要群組	次要群組	權限
user1	user1	X	r w x
user2	user2	X	r - -
user3	user1	X	r w -
user4	user4	user1	r - -

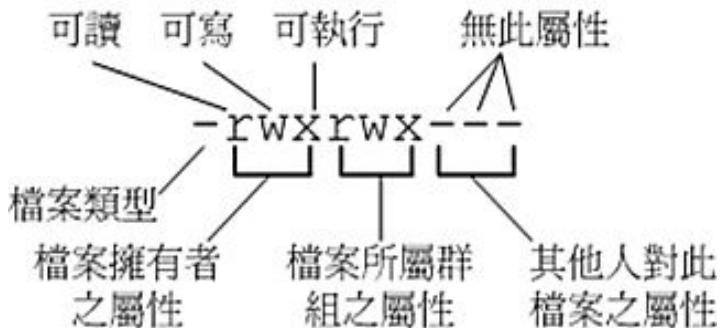
Module 18. 檔案目錄權限管理

- 18-1: 更改權限
- 18-2: 更改擁有者與群組
- 18-3: 特殊權限

變更檔案權限(chmod)

- 檔案的擁有者可以更改檔案權限

chmod	u (user)	+ (加入)	r	檔案或目錄
	g (group)	- (移除)	w	
	o (others)	= (設定)	x	
	a (all)			



- 各角色的權限可獨立或一起設定

指令範例	說明
chmod u=rwx,g=rx,o=r file1	權限會變成 rwxr-xr--
chmod u=rwx,o=r file1	只設定了u跟o, 所以群組權限會維持原樣
chmod o= file1	others的所有權限都關掉
chmod a=rw file1	所有人設定為可讀可寫
chmod a+x dir1	所有人都加上執行權限
chmod o-x file1	關掉others的執行權限

chmod 也可使用數字設定

- chmod 權限數字 [-R] 檔案或目錄

- r=4, w=2, x=1, -=0
 - 例如: rwx rw- r-- 可看成
 $(4+2+1)(4+2+0)(4+0+0) \square 764$

指令範例	說明
chmod 740 file1	權限會變成rwxr----
chmod -R 700 dir1/*	dir1內所有檔案及子目錄的權限皆變成rwx-----

- 以下為錯誤的設定語法:

- chmod o=2 file1 #不能用英數字夾雜
- chmod rwxrw-r-- file1 #不能直接用權限符號

練習：目錄權限

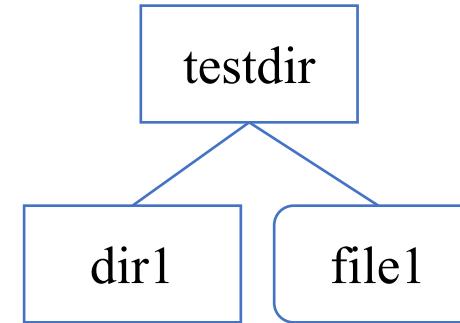
• 依序執行下列指令：

- mkdir testdir
- ls -l
- cd testdir
- touch file1
- mkdir dir1
- cd ..
- chmod -w testdir
- cd testdir
- touch file2
- cd ..
- chmod -x testdir
- cd testdir
- ls -l testdir

drwxrwxr-x
因為有x權限
因為有w權限
因為有w權限

移除w權限
仍可以切換至目錄內，因仍有x權限
無法建立檔案了

移除execute權限
無法進入目錄
可看到檔案清單，因仍有r (亂碼)



更改檔案擁有者(chown)

- chown 使用時機?

- 例如把檔案複製給其他人使用時，若擁有者未改成對方，可能會造成操作權限不足

- chown [-R] 帳號 檔案/目錄

- 將檔案或目錄的擁有者修改
 - -R：子目錄內的檔案也會改
 - 需root權限

- 範例：

- sudo chown user1 file1
 - sudo chown user1:group2 file1

#將擁有者改為user1
#擁有者改為user1,
群組改為group2

- chgrp [-R] 新群組名稱 檔案/目錄
 - 將檔案或目錄所屬群組改為新群組
 - 群組名稱必須是已存在系統中的
 - 需root權限
 - -R : 子目錄內的檔案也會改
- 範例：
 - sudo chgrp group2 file1 #將file1所屬群組改為group2
 - sudo chgrp group2 dir1 #將dir1所屬群組改為group2



- 若要cd切換至目錄dir1內，需具備哪種權限？
 1. r
 2. w
 3. x
 4. r及x
- 以下何者為正確的檔案權限設定指令？
 1. chmod 664 file1.txt
 2. chmod rw-rw-r-- file1.txt
 3. chmod u=6,g=6,o=4 file1.txt
 4. chmod ugo=664 file1.txt

- SUID(Set UID)
 - 限適用於執行檔(binary program)
 - 使用者對該檔案必須具有 x 的權限
 - 只有在執行過程中，使用者可以取得擁有者(owner)的權限
- 範例：
 - 讓所有人都可以執行passwd來修改自己的密碼
 - 但擁有者是root, 只有在執行時讓他取得root權利以用來變更帳號設定檔

```
user1@localhost:~  
檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)  
[ user1@localhost ~]$ ls -l /usr/bin/passwd  
-rwsr-xr-x. 1 root root 27832 6月 10 2014 /usr/bin/passwd
```

/tmp

是所有帳號均可以寫入的一個暫存目錄，但若權限設定為777，則會照成你放的資料可以被其他人刪掉，為了解決此問題，/tmp會加上一個Sticky bit 的特殊權限，當使用者在該目錄下建立檔案或目錄時，只有自己與root有權利刪除該檔案或目錄

```
[ user1@localhost ~]$ ls -ld /tmp
drwxrwxrwt. 14 root root 4096 7月 2 08:42 /tmp
```

Module 19.

檔案打包壓縮

- 19-1: 檔案壓縮
- 19-2: 檔案打包
- 19-3: 檔案解壓縮

- 壓縮檔的附檔名只是為了識別作用

*.zip	以 zip 程式壓縮的檔案
*.gz	以 gzip 程式壓縮的檔案
*.bz2	以 bzip2 程式壓縮的檔案
*.xz	以 xz 程式壓縮的檔案

- 搭配tar使用

*.tar	以 tar 程式打包的檔案, 未壓縮
.tar.gz(.tgz)	以 tar 程式打包的檔案, 並以gzip壓縮
tar.bz2	以 tar 程式打包的檔案, 並以bzip2壓縮
*.tar.xz	以 tar 程式打包的檔案, 並以xz壓縮

zip 及 gzip

- zip / unzip

- zip file1.zip file1 #壓縮file1產生file1.zip
- unzip file1.zip #解壓縮file1.zip

- gzip / gunzip

- 只能壓縮一個檔案, 無法將多個檔案壓縮為一個檔案
- gzip file1 #壓縮後生成file1.gz, 並刪除原始檔案
- gzip -9 file2 #可以加1~9表示壓縮程度, 預設是6
- gzip -d file1.gz #解壓縮
- gunzip file2.gz #解壓縮

- bzip2 / bunzip2 (壓縮效率較佳)

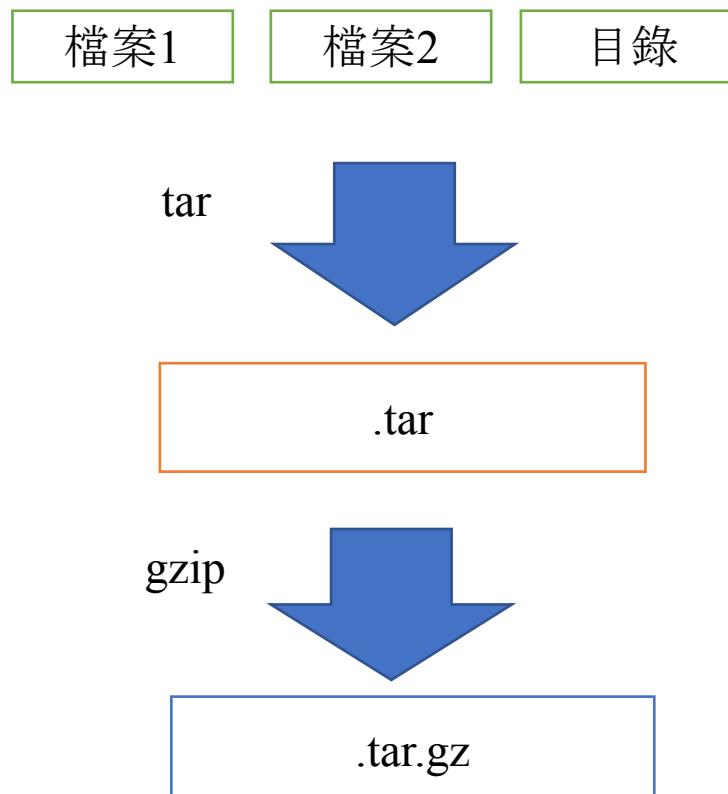
- bzip2 file1 #壓縮後生成file1.bz2, 但刪除原始檔
- bzip2 -k file1 #壓縮後生成file1.bz2, 但保留原始檔
- bzip2 -d file1.bz2 #解壓縮
- bunzip2 file1.bz2 #解壓縮

- xz / unxz (壓縮效率更好)

- xz file1 #壓縮後生成file1.xz, 並刪除原始檔
- xz -k file1 #壓縮後生成file1.xz, 但保留原始檔
- xz -d file1.xz #解壓縮, 並刪除壓縮檔
- unxz file1.xz #解壓縮, 並刪除壓縮檔

- 用於將多個檔案合併為一個檔案，以利備份和壓縮

- tar -cf test.tar dir1 #將dir1下所有檔案合併到test.tar
- gzip test.tar #再用gzip產生test.tar.gz



選項	說明
-c	建立檔案
-x	解開檔案
-v	顯示過程
-t	查看內容
-f 檔名	要被處理的檔案名稱
-z	由gzip處理
-j	由bzip2處理
-J	由xz處理
-C 目的地路徑	指定目的地路徑

- 把dir1打包，然後壓縮成不同的壓縮檔格式

- tar -zcf test.tar.gz dir1
 - tar -zcf test.tgz dir1
 - tar -jcf test.tar.bz2 dir1
 - tar -Jcf test.tar.xz dir1

- 檢視tar檔的內容

- tar -tvf test.tar.gz #只顯示內容物，未真實解開

- 解開tar壓縮檔

- tar -zxf test.tar.gz
 - tar -zxf test.tgz
 - tar -zxf test.tar.gz -C /tmp #解開來放到/tmp/下
 - tar -jxf test.tar.bz2
 - tar -Jxf test.tar.xz

tar配合壓縮範例

- 檢視tar檔的內容

- tar -tvf test.tar.gz #只顯示內容物, 未真實解開

- 解開tar壓縮檔

- tar -zxf test.tar.gz
 - tar -zxf test.tgz
 - tar -zxf test.tar.gz -C /tmp #解開來放到/tmp/下
 - tar -jxf test.tar.bz2
 - tar -Jxf test.tar.xz

- **壓縮**

- `tar -zcvf file1.tar.gz dir1`

- **解壓縮**

- `tar -zxvf file1.tar.gz`

Module 20. 文字編輯器

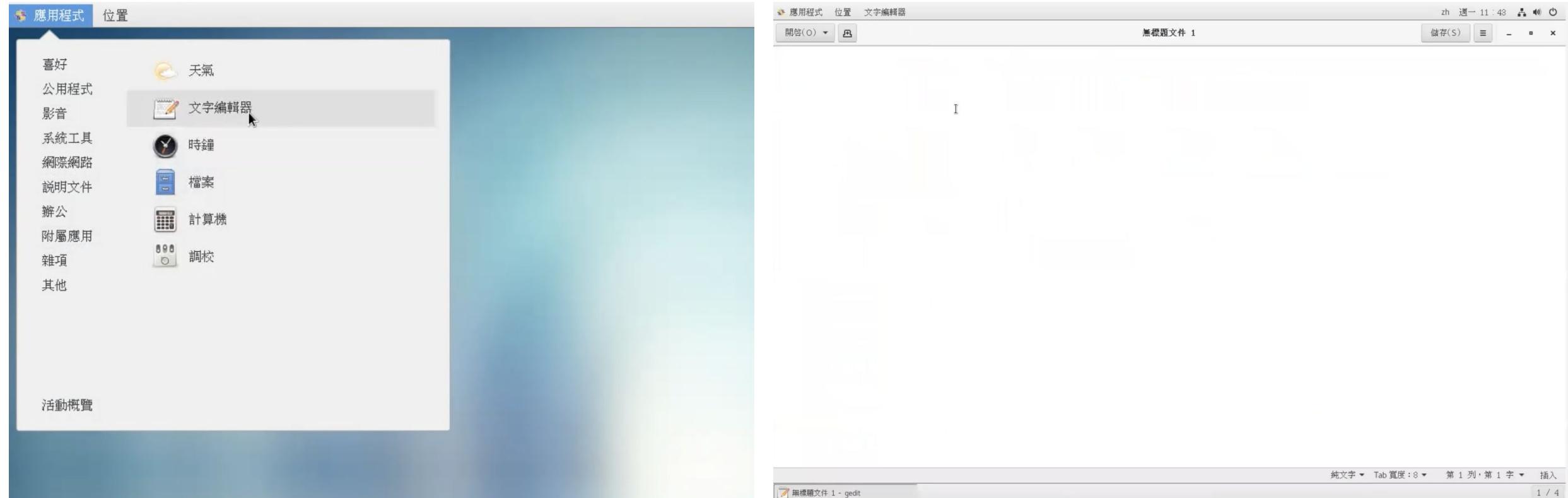
- 20-1: 文字編輯器介紹
- 20-2: gedit
- 20-3: nano

- **文字編輯器**
 - 用來編輯文字的工具，例如windows的記事本
- **圖形介面文字編輯器**
 - gedit
- **純文字介面文字編輯器**
 - nano
 - vi/vim

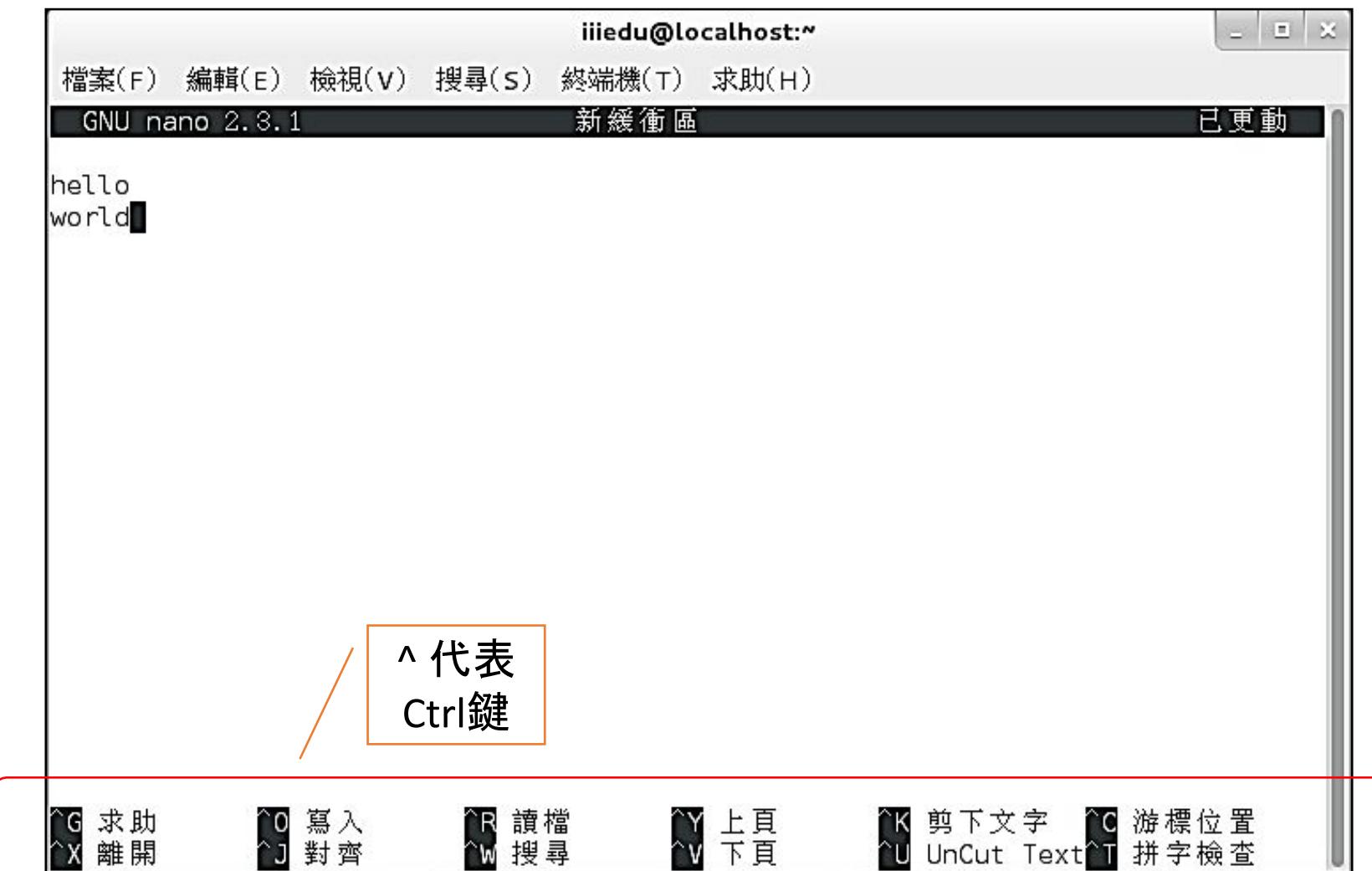
gedit

- gedit

- GNU弄出來的文字編輯器
- 可以下指令或者在圖形介面打開
- 如果在純文字介面時，無法使用



- 一套在純文字介面下可以使用的文字編輯器



- 複製一整行:Alt + 6
- 剪下一整行:Ctrl + K
- 貼上:Ctrl + U
- 搜尋:Ctrl + W
 - 然後輸入關鍵字, 按下Enter, 按下Alt+W可以跳到下一筆
- 翻到上一頁:Ctrl + Y
- 翻到下一頁:Ctrl + V
- 保存:Ctrl + O
- 退出:Ctrl + X

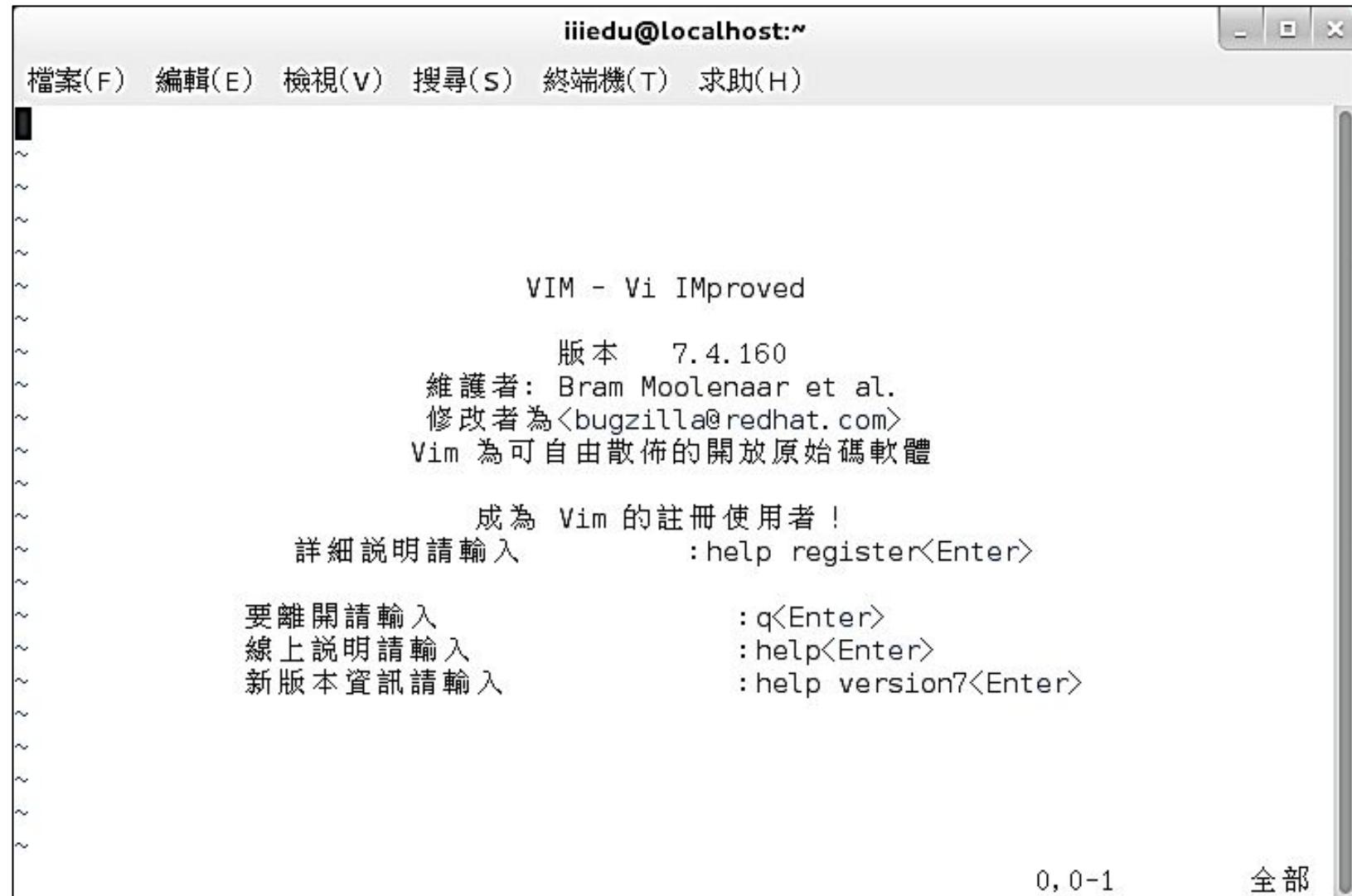
Module 21.

文字編輯器

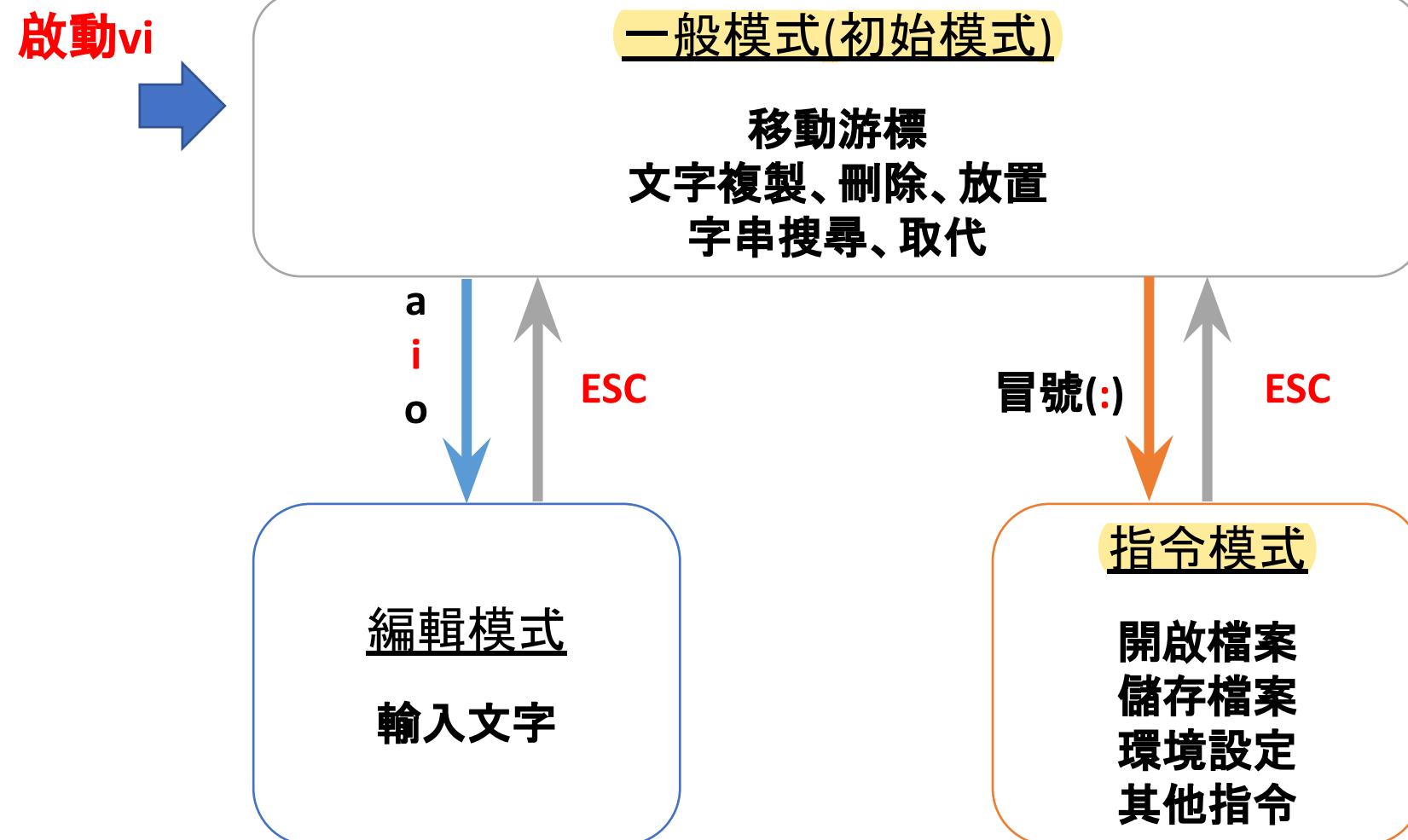
vi/vim

- 21-1: 指令模式
(command mode)
- 21-2: 編輯模式
(edit mode)
- 21-3: 命令列模式
(command line mode)

Vi / Vim

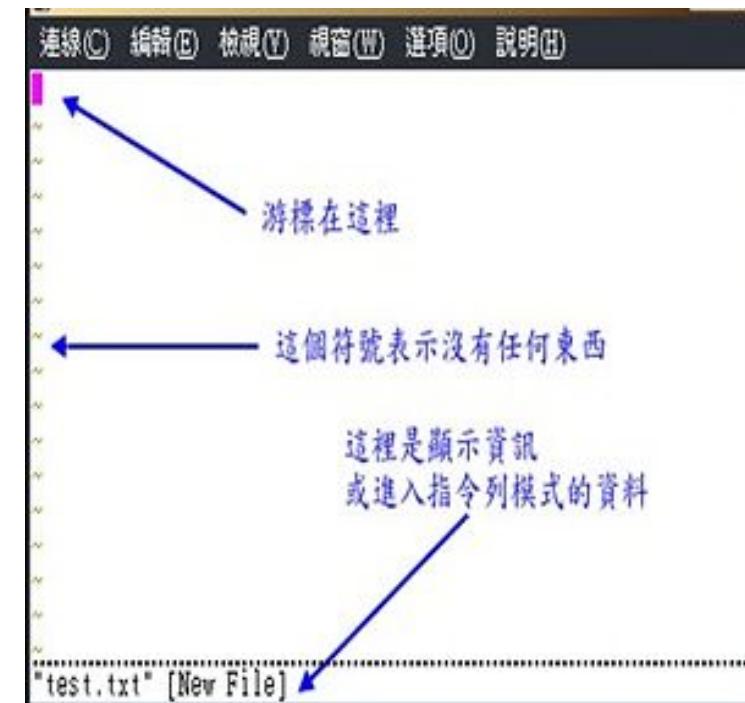


vi (vim) 的操作模式



vi啟動 (初始模式)

- vi test.txt #開啟(或自動新增)test.txt
 - 或是先開啟vi, 然後執行 :e test.txt
- 預設會進入一般模式
 - 可進行複製、貼上、刪除
 - 無法輸入文字
 - 隨時可按下Esc回到一般模式



- 按下 a 或 i 或 o 就可以輸入文字內容

a

- append
- 從目前游標位置的下一個字元開始輸入

i

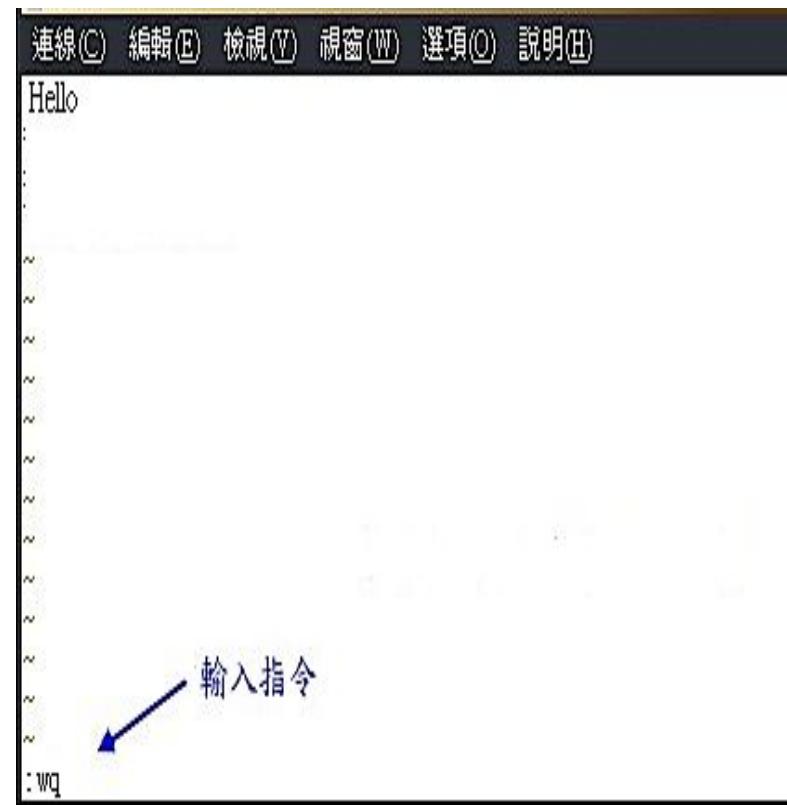
- insert
- 從目前游標位置插入新輸入的字元

o

- open
- 新增加一行，並將游標移到下一行的開頭

存檔及結束 (指令列模式)

- 先按**Esc**回到一般模式
- 先按**:**進入指令列模式
 - h #help
 - w #存檔
 - w 檔名 #另存新檔
 - wq 或 zz #存檔並離開vi
 - q! #強制離開vi
 - 5,9 w 檔名 #將第5~9行存到檔案



快速移動游標 (一般模式)

按鍵	說明
h 或 ←	左
j 或 ↓	下
k 或 ↑	上
l 或 →	右
0	移動到此行的開頭
\$	移動到此行的結尾
5G 或 :5 (命令模式)	移動到檔案的第5行
1G	移動到檔案的第一行
G	移動到檔案的最後一行

按鍵	說明
w	移到下一個word
b	移動至上一個word
(移動至上一句 (註1)
)	移動至下一句
{	移至該段落之首 (註2)
}	移至該段落之末
%	移動至相對應的 (), { }, [] 等符號
H	移動至現在螢幕畫面的開頭
M	移動至現在螢幕畫面的中間
L	移動至現在螢幕畫面的結尾

- (註1) 句子(sentence) 是指以『!』、『.』或『?』結束的一串字
- (註2) 段落(paragraph)是指以空白行隔開的文字

快速編輯指令(一般模式)

功能類別	指令	功能描述
刪除 (delete)	x	刪除游標所在位置的字元 (如同Delete)
	X	倒退鍵 (如同Backspace)
	dl	刪除一個字元 (delete literal)
	dw	從游標位置開始刪除, 一直到下一個單字(delete word)
	D	從游標位置開始刪除, 一直到行末
	dd	刪除游標所在的一整行
	5dd	刪除游標開始的5行
	:6,9d	刪除6-9行 (命令模式)
複製 (yank)	yy	複製游標所在的一整行
	3yy	複製從游標開始的3行
	yw	複製目前單字 (yank word)
放置 (put)	p	將複製(或刪除)的內容插入到游標位置的後面
	P	將複製(或刪除)的內容插入到游標位置的前面

搭配使用(一般模式)

yG	->	當前行複製到最後一行
yw	->	複製一個word
y0	->	複製到行首, 不含游標所處字元
y\$	->	複製致行尾, 含游標所在處字元
y7G	->	複製到第7行

練習：

1GyG ->
1GyGGp ->

搜尋及其他 (一般模式)

功能類別	指令	功能描述
搜尋	/字串	往後前搜尋字串, 按n找下一筆, N找上一筆
	?字串	往前搜尋字串
其他	J	將目前行與下面的行合併為一行
	u	undo 最後的修改
	Ctrl-r	redo
	.	重複上次的動作

- 可設定當前的操作環境

指令	說明
:set number (或:set nu)	設定資料的行號
:set nonumber (或:set nonu)	取消行號設定
:set ai	自動內縮 (auto-indent)
:set noai	取消自動內縮
:set ignorecase	搜尋不分大小寫
:set noignorecase	搜尋區分大小寫
:set shiftwidth=4	內縮設定為4個空白

- 使用者可編輯家目錄下的 vim 設定檔
 - `vim ~/.vimrc`

設定字串	說明
<code>set number</code>	顯示列號
<code>syntax on</code>	語法高亮度顯示
<code>set hlsearch</code>	標記搜尋到的字串
<code>set autoindent</code>	自動縮排
<code>set ruler</code>	顯示說明
<code>set showmode</code>	顯示編輯狀態
<code>set tabstop=4</code>	設定tab鍵的字元數
<code>set enc=utf8</code>	文字編碼加入utf8
<code>set ignorecase</code>	搜尋不分大小寫
<code>set background=dark</code>	設定顯示的亮度

V1/V1m 圖解鍵盤指令

version 1.1
April 1st, 06
翻譯: 2006-5-24

Esc

命令
模式

V1 / V1m 圖解鍵盤指令



動作

移動游標，或定義欲操作的範圍

指令

直接執行的指令

紅色指標進入編輯模式

操作

後接用以表示操作範圍的指令

extra

特殊功能

需額外輸入

q*

後接字元構成的參數

w,e,b指令

b(小寫): guux(foo, bar, baz);

B(大寫): guux(foo, bar, baz);

主要ex指令:

:w (儲存), :q (退出), :q! (不儲存退出)
 :e f (開啟文件 f),
 :%s/x/y/g (以 'y' 全文替換 'x'),
 :h (輔助文件 in vim), :new (新建文件 in vim)

其它重要指令:

Ctrl-R: 重複 (vim),
 Ctrl-F/B: 向前(下)翻頁/向後(上)翻頁,
 Ctrl-E/Y: 向前(下)一列/向後(上)一列,
 Ctrl-V: 切換visual模式 (vim only)

visual模式:

游標移動選擇區域，並執行特定操作 (vim only)

備註:

- (1) 在 複製/貼上/刪除 指令前使用 "x (x=a..z)" 使用指令的暫存器 (如: "ay\$ 複製該行目前位置至行尾的內容到暫存器'a')
- (2) 命令前添加數字 重複指定次數的操作 (如: 2p, d2w, 5i, d4j)
- (3) 重複游標所在字元處指定的操作 (dd = 刪除本行, >> = 行首縮排)
- (4) ZZ 儲存離開, ZQ 不儲存離開
- (5) zt: 移動游標所在行至畫面頂端, zb: 底端, zz: 中央
- (6) gg: 文件開端 (vim only), gf: 開啟游標處的文件名稱 (vim only)

- 將 /etc/passwd 複製到家目錄下
 - cp /etc/passwd ~
- 以 vi 開啟 ~ /passwd 後，練習以下操作：
 - 游標快速切換至第 10 行
 - 複製 3 行內容
 - 游標快速切換至第 20 行，貼上
 - 刪除第 10 行~ 第 12 行所有內容
 - 搜尋字串 “root”
 - 轉為“插入”模式，在檔案開頭新增一行 ABCDEabcde
 - 換行再加入“1234567890”
 - 存檔後退出

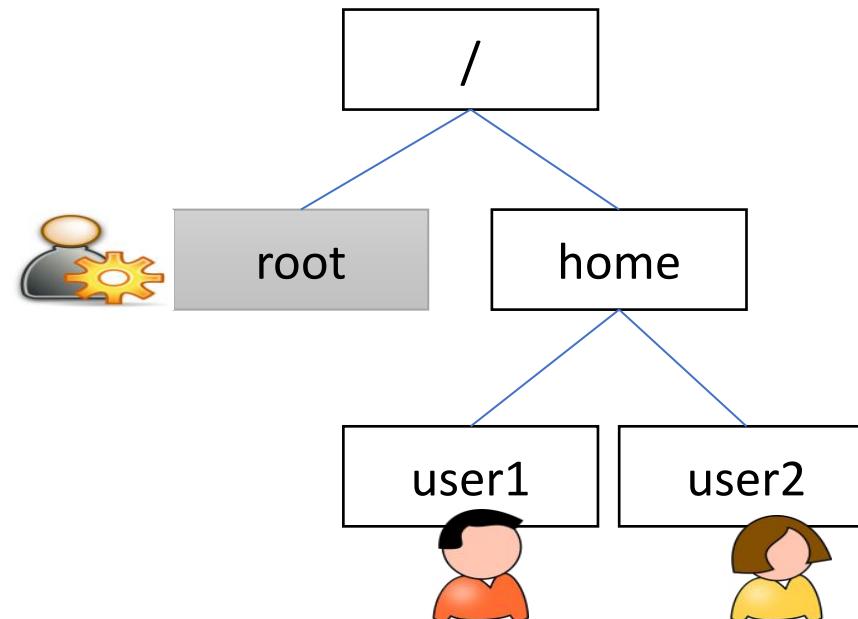
Module 22. 帳號管理

- 22-1: 帳號相關檔案
- 22-2: 新增/刪除使用者
- 22-3: 修改帳號資訊

- 帳虧名稱是唯一的、不可重複，且大小寫有別
- 命名規則
 - 帳虧名稱組成可以為字母或數字或底線或 - 或 .
 - 開頭必須是字母或 "_"
 - 中間不能留有空白字元
- 帳虧編號(UID)
 - 每個帳虧會對應一個不重複的編號(User ID, 簡稱UID)

UID	身分	帳虧範例
0	root	root
1~999	系統服務帳虧	cdrom、daemon
1000~65535	一般使用者	user1 (1000) user2 (1001)

- root帳號為擁有最高權限的系統管理員
 - 可以任意讀取、修改、刪除檔案或目錄，以及設定系統
 - 儘量少用root帳號登入，減少因手誤而損壞系統的風險
- 家目錄為 /root/
- UID是0, GID也是0
- shell提示符號為 #



帳號相關的設定檔

/etc/passwd

- 記錄系統帳號相關資訊

/etc/shadow

- 記錄使用者帳號密碼相關資訊

/etc/passwd

user1 : x : 1000 : 1000 : user1, , , : /home/user1:/bin/bash

/etc/shadow

user1 : \$1\$VM7t54w3\$EoDTTR/XvqUet.DDuKv1:15972:0:99999:7:::

- 每行代表一個帳號，以 : 分隔以下欄位

1. 使用者帳號，1到32字元內
2. 密碼，顯示x表示已改存於/etc/shadow
3. 帳號的ID (UID)
4. 帳號所屬主要群組的ID (GID)
5. 此帳號的備註
6. 家目錄的路徑
7. 使用者登入後預設的shell

預設任何人皆可
檢視此檔案內容

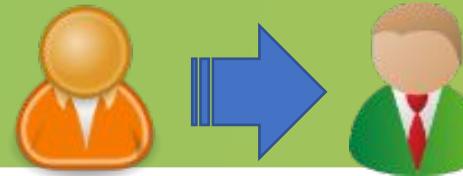
```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
...
user1:x:1000:1000:user1,,,:/home/user1:/bin/sh
```

- 每行代表一個帳號，以 : 分隔以下欄位

1. 帳號名稱
2. 加密後的密碼
3. 密碼最後改變的日期 (從1970年1月1日開始算)
4. 密碼不可被更動的天數 (相對於第3欄)
5. 這組密碼還有多久會過期 (相對於第3欄)
6. 警告更改密碼間隔天數 (相對於第5欄)
7. 密碼過期後帳號還有多久會解除 (相對於第5欄)
8. 帳號還有多久會過期(從1970年1月1日開始算)
9. 保留欄位

預設只有root或shadow群組的帳號才可以檢視

```
root:$6$QTk8cUgy$aezCPP3OxErgMJnfVtdZV2snM600:16108:0:99999:7:::  
daemon:*:15749:0:99999:7:::  
bin:*:15749:0:99999:7:::  
sys:*:15749:0:99999:7:::  
...  
user1:$6$CQFNyE0Z8l7zlUth$VM7t54w3EoDTTR/FTXvqUet.DDuKv1:15972:0:99999:7:::
```



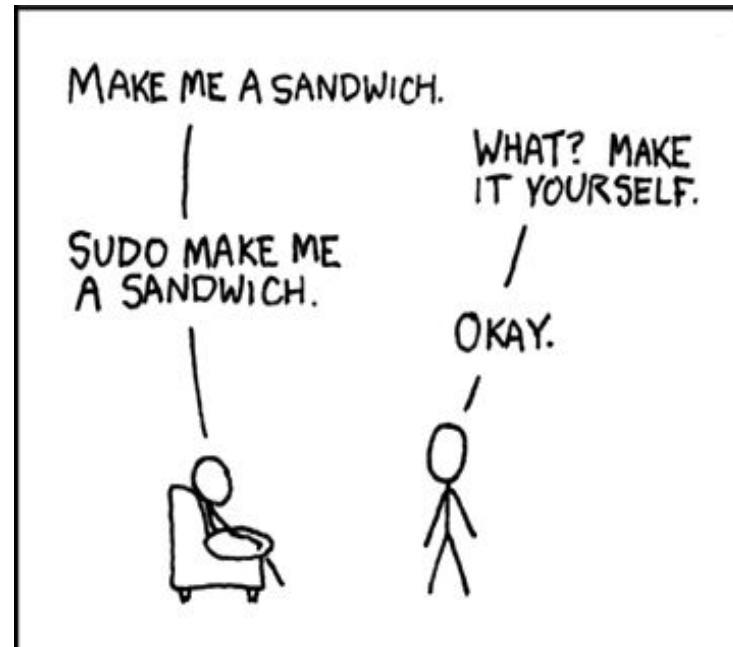
- su [帳號]
 - 用於切換目前使用的帳號，需輸入新帳號的密碼
 - 預設會切換成 root
 - 切換後會產生新的shell (子Shell) 讓新帳號使用
 - 要返回原帳號可使用 exit (結束子Shell, 返回至原shell)
 - 選項 - : 代表使用login-shell的變數檔案讀取方式來登入系統
- 範例：
 - su -root #切換成root, 使用root的mail與環境變數等設定
 - su root #切換成root, 使用原帳號的mail與環境變數等設定
 - exit #返回原帳號

請觀察兩者切換後的
絕對路徑差別

```
[iiiedu@localhost ~]$ su - root  
密碼：  
上一次登入：五 1月 3 14:51:57 CST 2020在 pts/0  
[root@localhost ~]# pwd  
/root  
[root@localhost ~]# exit  
logout  
[iiiedu@localhost ~]$ su root  
密碼：  
[root@localhost iiiedu]# █
```

使用sudo關鍵字

- sudo 指令
 - 暫時用root權限執行該指令，但執行完畢後權限立即消失
 - 使用時必須輸入目前帳號的密碼，以驗證其身分資格
 - 每次密碼可維持5分鐘內不需再輸入密碼
- 使用sudo的好處：
 - 易於管理
 - 只在必要時才給予root權限
 - 可提升管理上的安全性



<https://zh.wikipedia.org/wiki/Sudo>

<https://www.garron.me/en/linux/visudo-command-sudoers-file-sudo-default-editor.html>

- 利用sudo取得root權限，執行updatedb以更新檔案索引表

```
[user1@localhost ~]$ sudo updatedb  
  
We trust you have received the usual lecture from the local System Administrator. It usually boils down to these three things:  
  
#1) Respect the privacy of others.  
#2) Think before you type.  
#3) With great power comes great responsibility.  
  
[sudo] password for user1:  
[user1@localhost ~]$ █
```

需輸入目前帳號的密碼

執行完畢，立即切換回一般身分



誰能使用sudo?

- 只有寫於/etc/sudoers設定檔內的帳號才可以使用 sudo
- 讓user2 擁有使用sudo的權力
 - 以root權限執行visudo
 - 會以vi編輯設定檔(/etc/sudoers), 並於存檔前幫你做語法檢查

```
[root@localhost ~]# visudo

....(前面省略)....
root  ALL=(ALL)    ALL  -> 先找到這一行
user2 ALL=(ALL)    ALL  -> 這一行才是你要新增的
....(底下省略)....
```

wheel群組成員也可以使用sudo

- 安裝Linux系統時所用的管理員帳號(wheel群組的成員)也可以使用sudo
 - 雖然/etc/sudoers沒有設定user1的權力，但是藉由設定了wheel群組的權力，而wheel是user1的次要群組，所以user1可以使用sudo

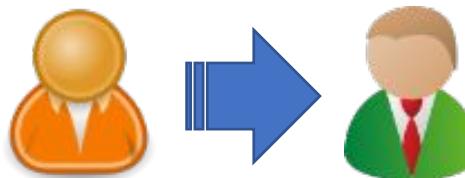
```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)            ALL
```

- 所以？其實只要加入wheel這個群組就有資格使用sudo了
- 以root權限執行
 - sudo usermod -G wheel user3 #將user3次要群組設定為wheel

切換成root帳號的第二種方式

- 要切換成root帳號，除了用su以外，也可以用sudo -i
- 好處為不需輸入root密碼，而是自身密碼
- 切換回原本帳號也是用exit

```
[ user1@localhost ~]$ sudo -i
[sudo] password for user1:
[ root@localhost ~]# pwd
/root
[ root@localhost ~]# exit
logout
[ user1@localhost ~]$ █
```



<https://wiki.centos.org/zh-tw/TipsAndTricks/BecomingRoot>

更改帳號的密碼(passwd)



- passwd [帳號] #**更改帳號的密碼**
 - 不指定帳號則預設會更改目前帳號的密碼

```
user1@localhost ~]$ passwd
正在變更 user1 的密碼。
(目前的)UNIX 密碼:
輸入新的 UNIX 密碼:
再次輸入新的 UNIX 密碼:
passwd: 密碼已成功地變更
```

新密碼若設定太簡單，會被系統拒絕

- **以root權限更改其他帳號的密碼，就不需輸入原始密碼**

```
user1@localhost ~]$ sudo passwd user2
輸入新的 UNIX 密碼:
再次輸入新的 UNIX 密碼:
passwd: 密碼已成功地變更
```

更改帳號的密碼(chpasswd)

- echo 帳號:密碼 | chpasswd
- 適合用於自動化腳本，可批量設定密碼
- 會讀入未加密的密碼，並進行加密後寫入/etc/shadow
- 會造成history內有該筆密碼紀錄
- 範例：以root身分將user2的密碼改為12345
 - echo user2:12345 | sudo chpasswd

- 以連續數字建立多個帳號範例
 - 以下依序建立student1~student9, 密碼皆為12345
 - 需以root身分執行該腳本

```
#!/bin/bash

for((i=1;i<10;i++))

do

echo "Adding student""$i"

useradd student"$i"

echo student"$i":12345 | chpasswd

done
```



更改密碼期限(chage)

- chage -l user2 #列出詳細資訊
- chage user2 #設定期限

```
user1@localhost ~]$ sudo chage user2
正在為 user2 修改年齡訊息
請輸入新值, 或直接按 ENTER 鍵以使用預設值
密碼期限最小值 [2]:
密碼期限最大值 [60]:
最近一次密碼修改時間 (YYYY-MM-DD) [2015-01-02]:
密碼過期警告 [7]:
密碼失效 [-1]: 30
帳戶過期時間 (YYYY-MM-DD) [-1]:
```



- 最少相隔2天, 最多60天內才能改變密碼
- 密碼過期7天前發出警告, 過期30天後密碼失效
- 帳戶從不過期



- useradd 帳號名稱
 - 需root權限
 - 會在/etc/passwd內新增一行帳號紀錄
 - 會自動新增一個同名的群組
 - 會建立該帳號的家目錄、信件夾
 - 但**不會自動建立密碼**, 所以帳號仍無法登入
- 範例：
 - useradd user3 #建立帳號後會自動建立同名群組
 - passwd user3 #建立密碼

修改帳號(usermod)



- usermod [選項] 帳號
 - 需root權限

選項	說明
-u --uid UID	設定新UID
-d --home HOME_DIR	設定家目錄
-g --gid GROUP	設定主要群組
-G --groups GROUPS	設定次要群組
a apend	附加次要群組
-L --lock	鎖定帳號
-U --unlock	解鎖帳號

指令範例	說明
sudo usermod -g user1 user3	將user3的主要群組變更為user1
sudo usermod -g user3 user3	將user3的主要群組變更為user3
sudo usermod -G user1 user3	user3的次要群組設定成 user1
sudo usermod -aG group2 user3	user3的次要群組多加了一個group2
sudo usermod -L user3	user3將無法登入



- userdel 帳號

- 若只是暫時不用該帳號，可以只鎖定，不需刪除
- 需用root權限

選項	說明
-f	強制移除帳號
-r	移除家目錄及其他檔案（例如信箱或工作排程）

- 範例：

- `userdel -r -f user3 #強制刪除user3`

查詢帳號資訊



users

- 顯示目前登入系統的帳號
- 資訊較簡略

who

- 顯示目前登入系統的帳號

w

- 顯示目前登入系統的帳號
- w user1 只顯示該帳號

查詢帳號是屬於哪個群組(id或groups)

groups [帳號]

- 查詢該帳號所屬的群組

id [帳號]

- 印出帳號群組相關資訊

- groups user1
 - user1 : user1 wheel
- id user1
 - uid=1000(user1) gid=1000(user1) groups=1000(user1),10(wheel)
- id -u user1 #只印出UID
 - 1000
- id -G user1 #印出此帳號所有的群組ID
 - 1000 10



列出每個帳號最近登入時間(lastlog)

- lastlog

使用者名	埠號	來自	最後登入時間
root	pts/4		— 11月 2 23:52:19 +0800 2015
bin			**從未登入過**
daemon			**從未登入過**
adm			**從未登入過**
lp			**從未登入過**
...			
省略篇幅			
...			
pulse			**從未登入過**
gdm	:0		二 12月 1 00:42:47 +0800 2015
gnome-initial-setup			**從未登入過**
user1	pts/1	localhost	二 12月 1 17:01:11 +0800 2015



• last



user2	pts/1	localhost	Tue Dec 1 23:06 - 23:06 (00:00)
user1	pts/0	:0	Tue Dec 1 16:24 still logged in
user1	:0	:0	Tue Dec 1 16:24 still logged in
reboot	system boot	3.10.0-229.20.1.	Tue Dec 1 00:42 - 23:07 (22:24)
reboot	system boot	3.10.0-229.20.1.	Tue Dec 1 00:41 - 00:42 (00:00)
user1	pts/1	localhost	Tue Dec 1 01:29 - 01:29 (00:00)
user1	pts/2	:0	Mon Nov 30 20:07 - 01:03 (04:56)
user1	pts/1	:0	Mon Nov 30 12:31 - 20:43 (08:12)
user1	pts/0	:0	Mon Nov 30 12:06 - 20:40 (08:34)
user1	:0	:0	Mon Nov 30 12:05 - 01:30 (13:24)

- last -n 3 #顯示最近3筆紀錄
- last user1 #只顯示user1的登入紀錄
- last reboot #顯示重開機的紀錄

- 若要(永久)切換成root帳號，以下何者為非？
 1. su
 2. su root
 3. sudo
 4. sudo -i
- 要修改自己的密碼，最建議的方式為？
 1. 使用passwd指令
 2. 使用chpasswd指令
 3. 手動編輯 /etc/passwd
 4. 將密碼告知給root，請他幫你設定
- 以下敘述何者為非？
 1. 主要群組可以零至多個
 2. 帳號若無主要群組則無法登入系統
 3. root可以修改user1的主要群組設定
 4. 主要群組名稱預設與帳號名稱相同

Module 23. 群組管理

- 23-1: 群組相關檔案
- 23-2: 使用者專屬群組方案
- 23-3: 管理群組

- 群組設計之目的 
 - 為了方便管理系統、資源共享以及隱私安全性保護
 - 例如讓某個檔案只允許被某個群組的成員所存取
- 群組編號(GID)
 - 每個群組會對應一個不重複的編號(Group ID, 簡稱**GID**)

GID	群組身分	群主範例
0	root群組	root
1~999	系統服務群組	cdrom、daemon
1000~65535	一般群組	user1 (1000) user2 (1001)

- **主要群組**
 - 一個帳號只能擁有一個主要群組
 - 帳號若沒有主要群組，則該帳號無法登入
 - 新增帳號時，預設會指定一個與帳號同名的主要群組
 - 例如安裝帳號為user1 (uid=1000)，系統自動生出一個群組user1(gid=1000)為其主要群組
 - 用來判斷檔案/目錄的群組權限

- **次要群組**
 - 一個帳號可以擁有0至多個次要群組

群組相關的設定檔

/etc/group

- 記錄各個使用者是屬於那個群組

/etc/gshadow

- 記錄群組密碼相關資訊

/etc/gshadow

user1 : ! : :

/etc/group

user1 : x : 1000 :

- 一行代表一個群組，以:分隔，共4個欄位

1. 群組名稱
2. 群組密碼，顯示x表示已改存於/etc/gshadow
3. GID (Group ID)
4. 這個群組的會員

```
root:x:0:  
daemon:x:1:  
bin:x:2:  
sys:x:3:  
adm:x:4:user1  
tty:x:5:  
cdrom:x:24:user1  
...  
user1:x:1000:
```

預設任何人皆可
檢視此檔案內容

- 一行代表一個群組，以:分隔，共4個欄位

1. 群組名稱
2. 加密後的群組密碼
3. 群組管理員帳號
4. 這個群組的會員

預設只有root或shadow群組的帳號才可以檢視

```
root:*::  
daemon:*::  
bin:*::  
sys:*::  
adm:*::user1  
tty:*::  
cdrom:*::user1  
...  
user1:!::
```

新增/修改/移除群組

新增群組

- groupadd [-g gid] 群組名

修改群組

- groupmod [-n 新名稱] 群組名

移除群組

- groupdel 群組名

• 需root權限

- groupadd group1 #新增group1
- groupadd -g 1004 group2 #新增group2
- groupmod -g 1005 group2 #修改group2
- groupmod -n newgroup group2 #修改group2
- groupdel group1 #若仍有主要群組成員則無法刪除

Module 24. 軟體管理

- 24-1: 圖形化安裝
- 24-2: rpm與yum指令
- 24-3: apt與dpkg指令

- 依Linux核心而設計的軟體稱為套件，通常包含：
 - 可執行程式碼 (例如執行檔及函式庫)
 - 及其他額外資訊 (例如套件版本、相依性、說明文件)
- 安裝或更新套件的3種方式：

線上安裝

- 簡單方便
- 必須連結至伺服器
- 版本選擇不自由

執行安裝檔

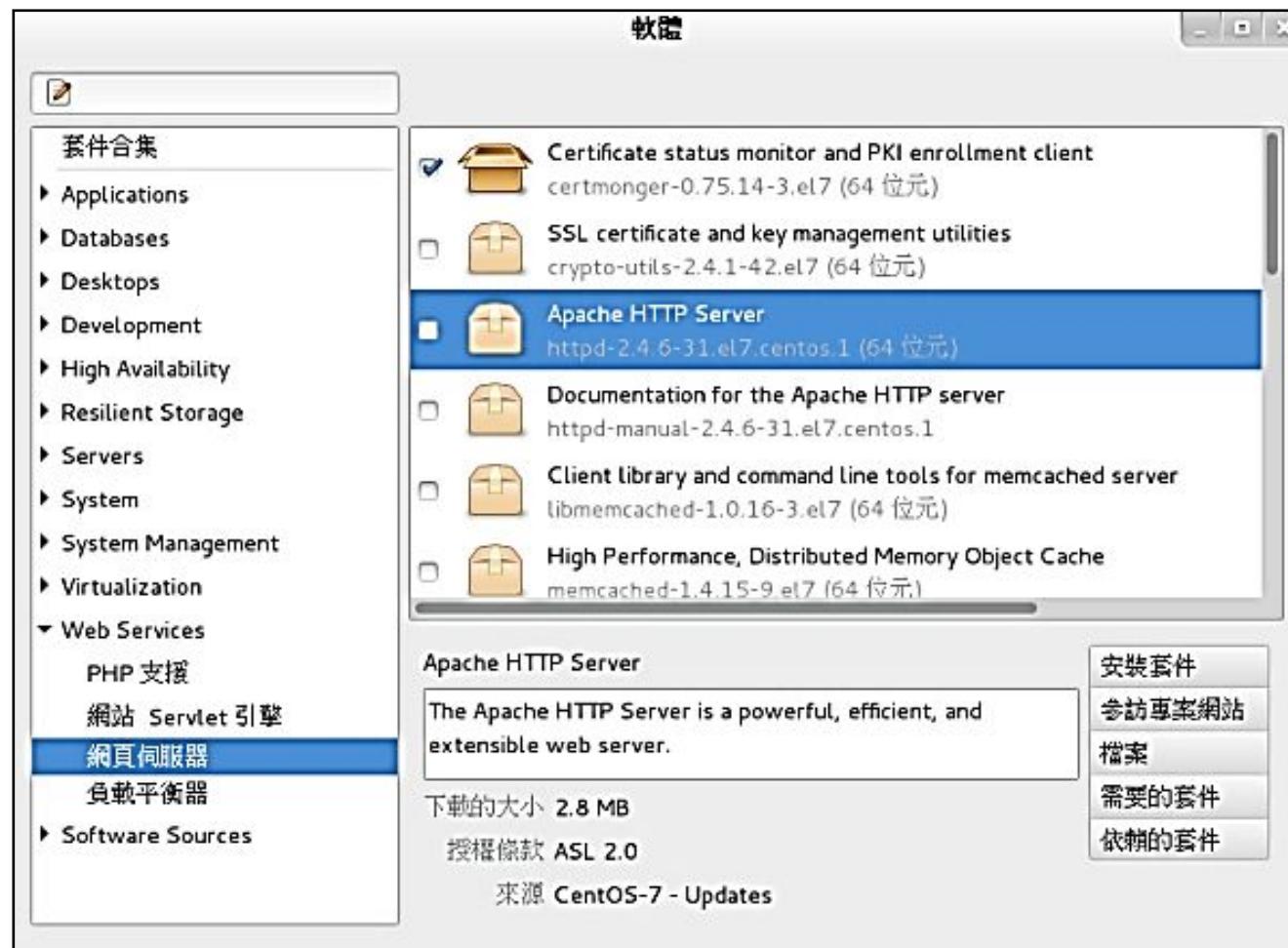
- 從網路或光碟取得安裝檔
- 再透過套件管理程式安裝

自行編譯TarBall

- 難度較高
- 版本選擇自由
- 可修改程式碼
- 可移植至其他平台

CentOS 圖形化套件管理工具

- CentOS 選單列 □ 系統工具 □ 軟體
- 操作畫面範例：



Linux套件管理機制:RPM與DPKG

RPM

(RedHat Package Manager)



RedHat / Fedora / CentOS / SuSE



安裝檔名格式: *.rpm



線上套件管理機制: YUM



使用的指令: rpm、yum

DPKG

(Debian package)



Debian / Ubuntu



安裝檔名格式: *.deb



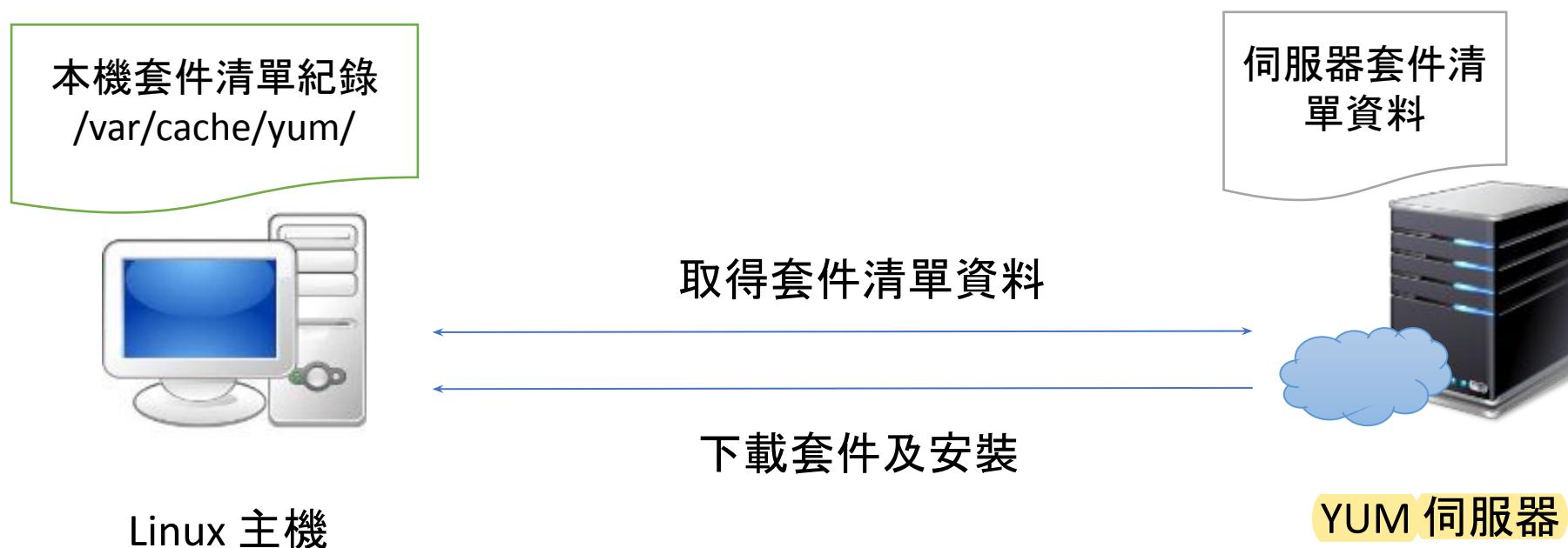
線上套件管理機制: APT



使用的指令: dpkg、apt

YUM線上套件管理機制

- 可進行線上安裝
- 安裝檔下載後會暫存在`/var/cache/yum/`
- 可自動處理套件的依賴關係



yum 指令用法

類別	用法	說明
安裝	yum install 套件名	安裝套件
	yum -y install 套件名	安裝時預設回答yes
	yum reinstall 套件名	重新安裝套件
	yumdownloader 套件名	只下載安裝檔
移除	yum remove 套件名	移除該套件
	yum clean	清除安裝套件時所下載的暫時檔案
更新	yum update	更新已安裝的所有套件
	yum update 套件名	更新該套件
檢視/查詢	yum list	列出所有的套件
	yum list installed	列出所有已安裝的套件
	yum list update	列出所有可更新的套件
	yum search "KEYWORD"	搜尋與KEYWORD有關的套件
	yum info 套件名	以套件名查詢相關資訊

- 搜尋能安裝的相關套件
 - yum search “subversion”
- 安裝subversion套件
 - sudo yum install subversion #安裝
 - yum info subversion #檢視版本資訊
- 執行說明
 - svn --help #執行subversion的指令(svn)

- 套件異動需使用root權限

類別	選項	意義	說明
安裝	rpm -i XXX1.rpm	install	以rpm安裝檔安裝套件
移除	rpm -e 套件1	erase	移除(解除安裝)該套件
更新	rpm -U XXX2.rpm	upgrade	以rpm安裝檔更新套件， 若未曾安裝過也仍會執行安裝動作
查詢	rpm -q 套件1	query	查詢套件1是否已安裝
	rpm -qa	query all	查詢目前系統上，所有已安裝的套件
	rpm -ql 套件1	query list	查詢套件的內容清單

- 以rpm安裝nmap
 - 首先必須下載安裝檔
 - 官網下載*.rpm <https://nmap.org/download.html>
 - 或是 wget https://nmap.org/dist/nmap-7.00-1.x86_64.rpm
 - 安裝
 - sudo rpm -i nmap-7.00-1.x86_64.rpm
 - nmap --version #查詢當前版本
- nmap使用範例
 - 以nmap掃描其官網的port、系統資訊
 - nmap -v -A scanme.nmap.org

APT線上套件管理機制

- DPKG系統的中階套件管理工具，可進行線上安裝
 - sudo apt-get install subversion #適用於ubuntu

類型	指令	說明
安裝	apt-get install 套件名	安裝套件
	apt-get install --reinstall 套件名	重新安裝套件
	apt-get download 套件名	只下載安裝檔
移除	apt-get remove 套件名	移除套件，但保留設定檔
	apt-get remove --purge 套件名	完整移除套件，不保留設定檔
	apt-get autoclean	移除無用套裝軟體檔案，但保留安裝檔
	apt-get clean	移除無用套裝軟體檔案，不保留安裝檔
更新	apt-get update	更新套件清單
	apt-get upgrade	逐一升級所有有新版的軟體套件
檢視/查詢	apt-cache search 字串	搜尋可用套件
	apt-cache -n search 字串	僅從套件名稱搜尋
	apt-cache show vim	查詢vim套件資訊

dpkg

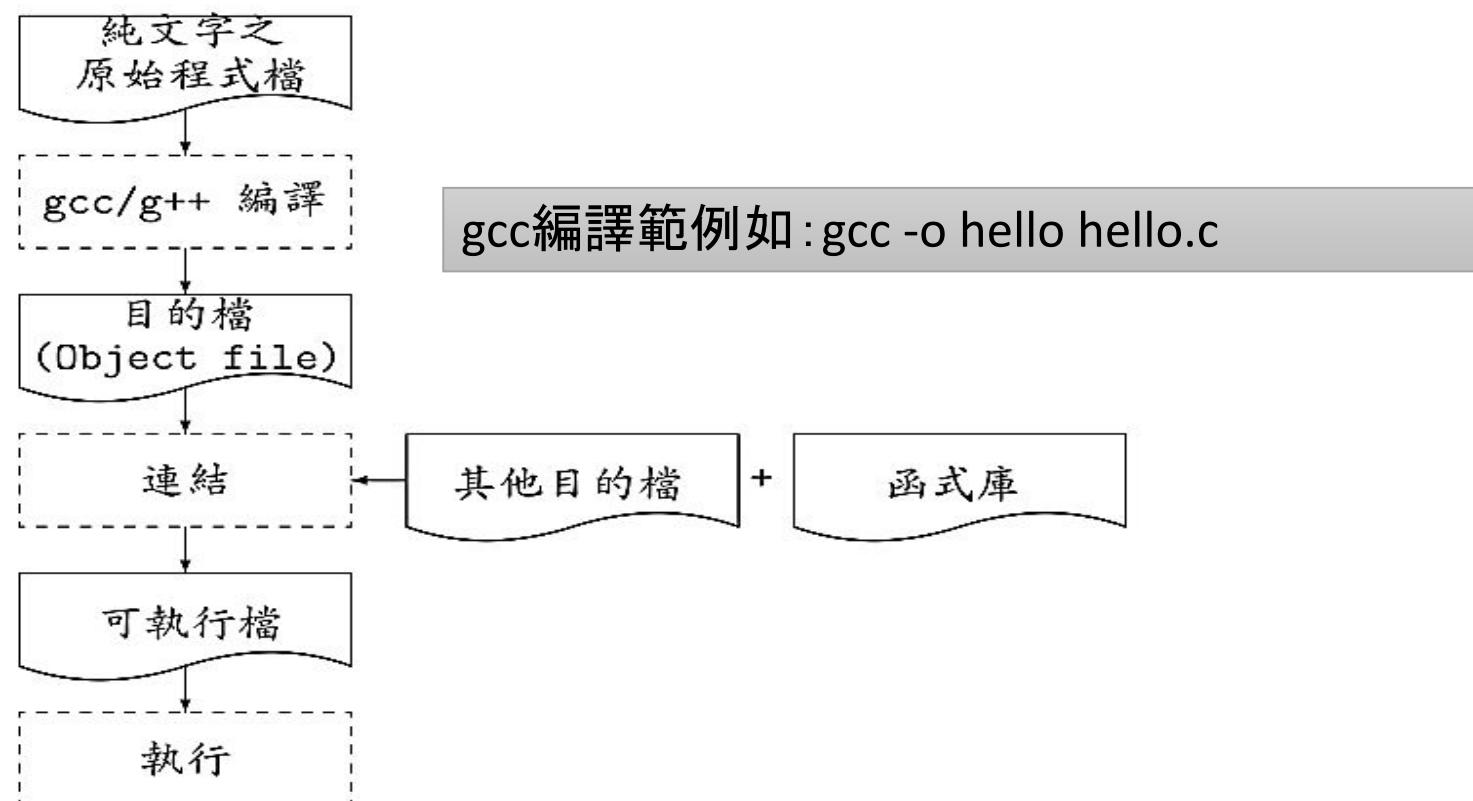
- 屬於DPKG系統的低階套件管理工具 (Ubuntu適用)
- 安裝檔格式為 *.deb
- 範例：
 - google搜尋並下載「subversion_1.9.5-1ubuntu1_i386.deb」
 - 安裝：sudo dpkg -i subversion_1.9.5-1ubuntu1_i386.deb

類型	指令	說明
安裝	dpkg -i XXX.deb	需自行下載安裝檔
檢視查詢	dpkg --info XXX.deb	查詢安裝檔資訊
	dpkg --list	顯示目前安裝的套件清單
	dpkg --listfiles vim	哪些跟vim有關的檔案被安裝了
	dpkg --status vim	查詢vim套件狀態, 版本等
	dpkg -p vim	顯示vim可用版本的詳細資訊
移除	dpkg -r vim	移除vim (套件的名稱)

Module 25. 軟體編譯與建置

- 25-1: 開源軟體
- 25-2: 軟體編譯流程
- 25-3: 使用Tarball建置軟體

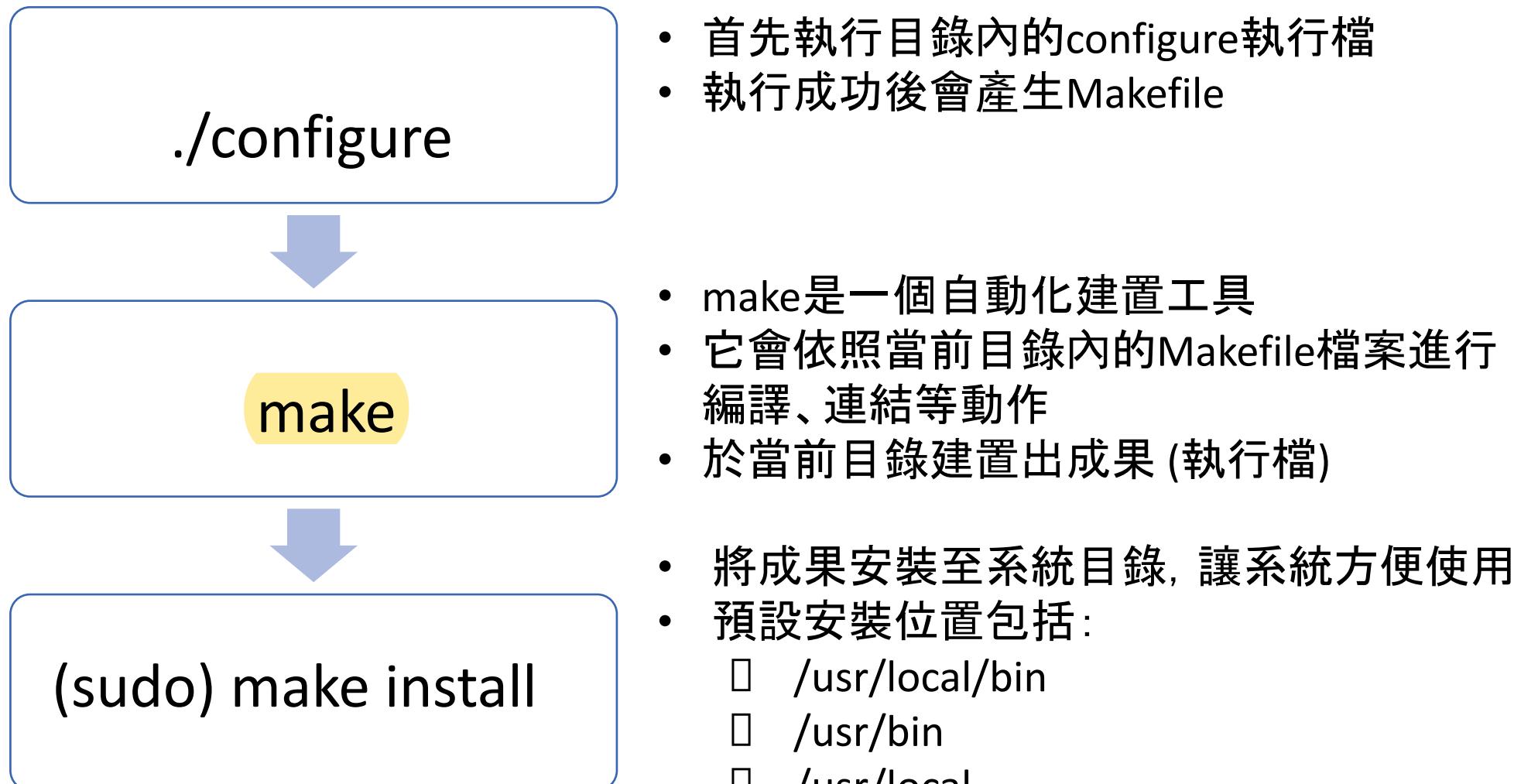
- Linux真正可執行的檔案是二進位檔案(binary file)
 - 例如cp、ls等
- 寫完一個C程式後需要有C的編譯器(例如gcc)將C原始碼轉換成機器可以執行的程式碼



- tarball 就是以 *.tar.gz 等型式壓縮之後的套件原始檔
- tarball組成內含：
 - 原始碼(例如 *.c , *.h 等)
 - 組態設定檔(configure)
 - 此為偵測用的程式
 - 目的在收集當前建置環境的資訊(作業系統及參數等)
 - 並檢查相關工具是否存在(如gcc等有沒有裝?)
 - 建置說明
 - 例如 INSTALL及 README
- 必須自行上網取得tarball後再手動建置

編譯 Tarball 的流程

- 進入解開來的tar ball目錄後，依序執行：



- 自行以Linux瀏覽器上網搜尋及下載 joe-4.4.tar.gz
- tar -zxvf joe-4.4.tar.gz #解開tarball
- cd joe-4.4/
- ./configure #組態設定
 - 遇到gcc未安裝錯誤
 - sudo yum install gcc
 - ./configure
- make #建置
- sudo make install #安裝至系統目錄
- joe #執行joe
 - 按Ctrl K後再按q 結束離開



如何指定安裝路徑？

- 建議於執行組態設定時指定路徑
- `./configure --help` #可查詢支援的選項
- `./configure --prefix=“自訂的安裝路徑”`
- 例如：
 - `./configure --prefix=/tmp/`
 - `make`
 - `make install`
- 但建議修改\$PATH以方便系統尋找執行檔
 - 設定範例：
 - `PATH=$PATH:/tmp/bin/`

如何清除編譯成果？

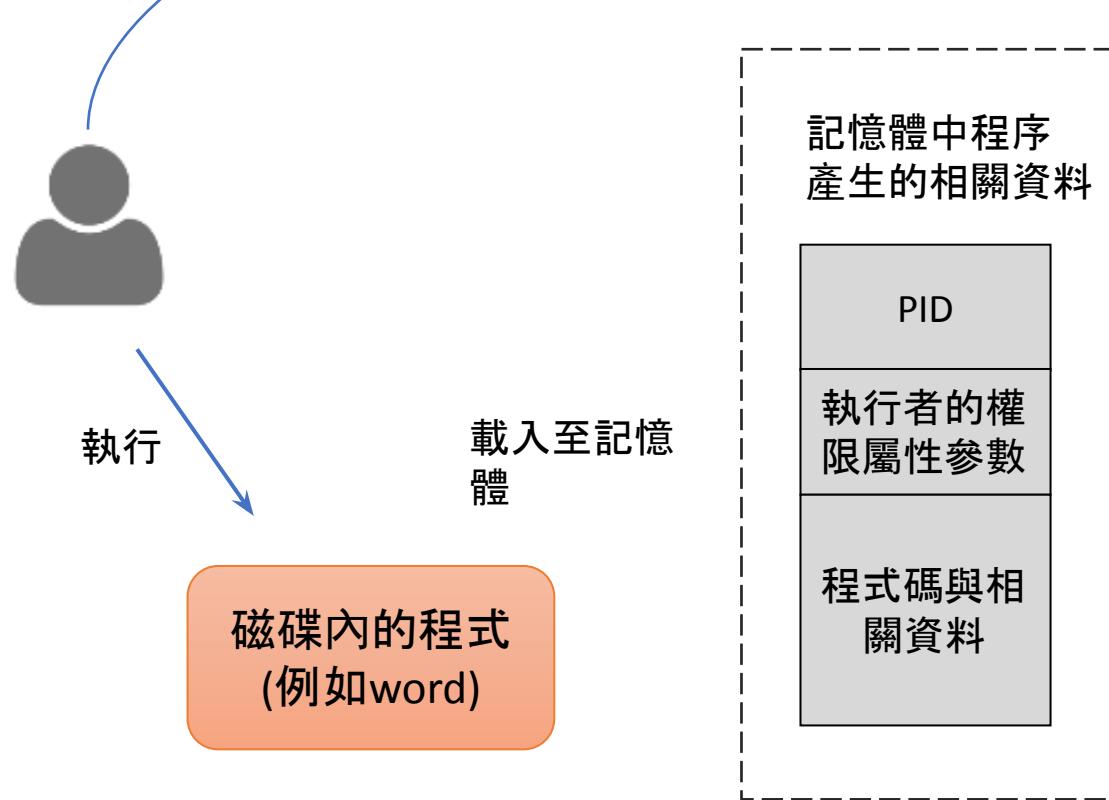
- 於Makefile所在目錄執行以下指令 (擇一執行即可)
 - 必須依賴Makefile內所撰寫的規則，才能執行相應動作
- make clean
 - 移除新建置的檔案
- make mrproper
 - 移除新建檔案
 - 移除設定檔
- make distclean
 - 移除新建檔案
 - 移除設定檔
 - 移除暫存檔、修補檔案等

Module 26. 程序管理

- 26-1: 程序架構
- 26-2: 檢視程序
- 26-3: 送信號給程序

程序(Process)

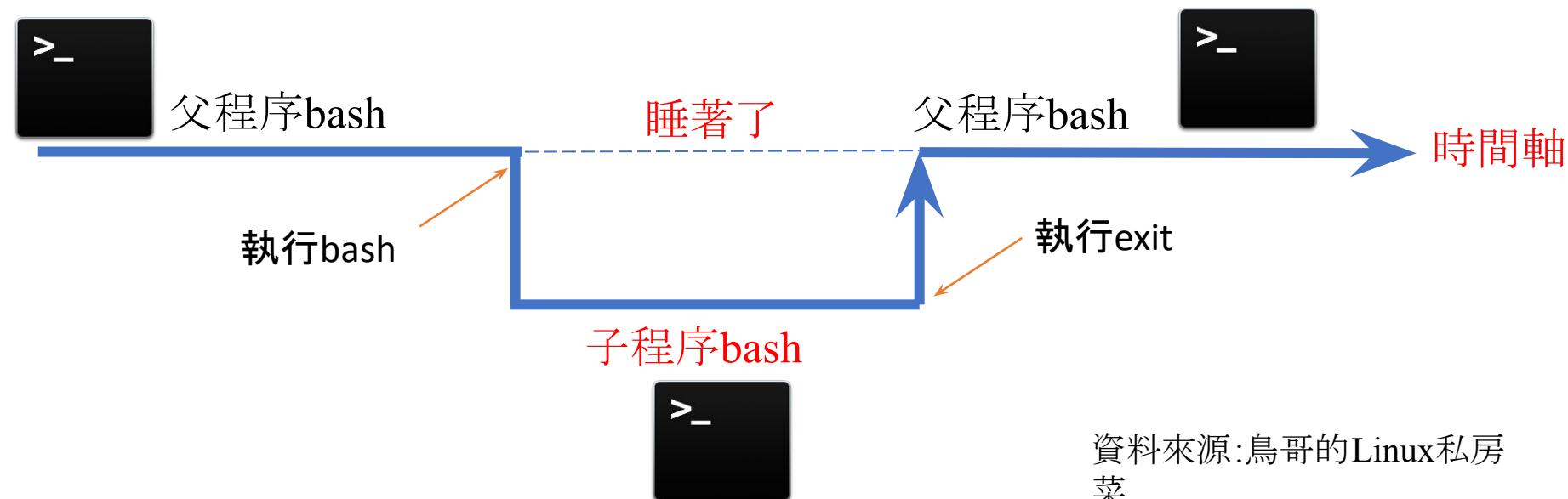
- 程式被觸發載入至記憶體中執行，即稱為程序
 - 同時也會載入執行者的權限屬性、程式碼與所需資料等
- 系統會指派對應的Process ID (**PID**)，用來管理程序



資料來源:鳥哥的Linux私房菜

父程序與子程序

- 「父程序」衍生出「子程序」
- 例如在bash內執行指令ls
 - 則父程序為bash, 子程序為ls
- 例如在bash內執行bash指令, 可新生一隻bash
 - 子程序的bash操作結束後, 可執行exit回到父程序bash



資料來源:鳥哥的Linux私房
菜

以樹狀結構檢視程序(pstree)

```
[iiiedu@localhost ~]$ pstree
systemd—ModemManager—2*[ { ModemManager}]
      | NetworkManager—dhclient
      |           3*[ { NetworkManager}]
      | 2*[ abrt-watch-log]
      | abrtd
      | accounts-daemon—2*[ { accounts-daemon}]
      | alsactl
      | anacron
      | 2*[ at-spi-bus-laun—dbus-daemon—{ dbus-daemon}]
      |           3*[ { at-spi-bus-laun}]
      | 2*[ at-spi2-registr—{ at-spi2-registr}]
      | atd
      | auditd—audispd—sedispatch
      |           { audispd}
      |           { auditd}
      | avahi-daemon—avahi-daemon
      | bluetoothd
      | chronyd
      | colord—2*[ { colord}]
      | crond
      | cupsd
      | 3*[ dbus-daemon—{ dbus-daemon}]
      | 2*[ dbus-launch]
      | 2*[ dconf-service—2*[ { dconf-service}]]
      | 2*[ evolution-addre—4*[ { evolution-addre}]]
      | evolution-calen—4*[ { evolution-calen}]
      | evolution-calen—5*[ { evolution-calen}]
      | 2*[ evolution-sourc—2*[ { evolution-sourc}]]
      | firewalld—{ firewalld}
      | fprintd
```

也就是目前系統上
程序的族譜

檢視系統中有哪些程序正在執行(ps)

- 僅列出自己的bash相關程序

- ps

PID	TTY	TIME	CMD
4276	pts/1	00:00:00	bash
4446	pts/1	00:00:00	ps

- ps -l #列出詳細資訊

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
0	S	1000	4276	4267	0	80	0	-	1564	wait	pts/1	00:00:00	bash
0	R	1000	4446	4451	0	80	0	-	1251	-	pts/1	00:00:00	ps

- 列出所有系統運作的程序

- ps aux #選項前沒有-

欄位	說明
PID	Process ID
PPID	Parent PID
TTY	登入的終端機代號
TIME	此程序所消耗的CPU時間
CMD	正在執行的程序名稱
NI	NICE值

ps aux | grep “關鍵字”

動態觀察程序的變化(top)

- top 可持續性觀察程序變化
 - 預設3秒更新一次。
 - top -d 10 #10秒更新一次



系統資訊

```
top - 00:47:33 up 12:42,  4 users,  load average: 0.20, 0.08, 0.06
Tasks: 152 total,   2 running, 150 sleeping,   0 stopped,   0 zombie
%Cpu(s): 33.1 us,  1.0 sy,  0.0 ni, 65.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 2049352 total,    78592 free, 1249412 used,   721348 buff/cache
KiB Swap:  839676 total,    832632 free,     7044 used.  556652 avail Mem
```

Process 資訊

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1903	iiiedu	20	0	1935800	585668	32092	S	26.2	28.6	15:02.03	gnome-shell
1254	root	20	0	307888	143644	6576	S	4.7	7.0	4:55.04	Xorg
2491	iiiedu	20	0	806300	25544	11504	S	2.0	1.2	0:31.30	gnome-terminal-
1942	iiiedu	20	0	461696	6592	3388	S	0.3	0.3	0:10.01	ibus-daemon
1	root	20	0	59748	5828	3160	S	0.0	0.3	0:03.20	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.53	ksoftirqd/0
5	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kworker/0:0H

top操作

- 執行top後，可進行以下的**互動式操作**

按鍵	功能
h	顯示說明文字
P	依據CPU使用時間排序 (預設)
M	依據記憶體使用量排序
T	依據執行時間排序
N	以PID由大至小排序
u 帳號	只列出該帳號的process
k [pid]	刪除process
d 秒數	更新的秒數
q	結束離開

- kill [-信號] PID #傳送終止信號給PID程序
 - 必須先查出PID
 - ps aux | grep “firefox”
 - 或是 pgrep firefox
 - kill -l #列出所有能使用的信號(Signal)

編號	信號	意義
2	SIGINT	Ctrl c
9	SIGKILL	強制刪除process
15 (預設)	SIGTERM	以正常的程序通知程式停止執行

- 範例
 - kill 12345
 - kill -9 12345



- 重新開啟一個新的終端機
- vim &
- ps
 - PID TTY TIME CMD
 - 4878 pts/1 00:00:00 bash
 - 4908 pts/1 00:00:00 vim #查出PID為4908
 - 4938 pts/1 00:00:00 ps
- kill -9 4908 #砍掉vim
- ps #查看vi是否被砍掉了
 - PID TTY TIME CMD
 - 4878 pts/1 00:00:00 bash
 - 4938 pts/1 00:00:00 ps

- killall 程序名稱
- 範例：
 - firefox & #背景執行第一個firefox
 - firefox & #背景執行第二個firefox
 - ps
 - PID TTY TIME CMD
 - 5513 pts/0 00:00:00 bash
 - 5610 pts/0 00:00:00 firefox
 - 5615 pts/0 00:00:00 firefox
 - 5639 pts/0 00:00:00 ps
 - killall -i -9 firefox #把firefox都砍掉
 - 信號 firefox(5610) ? (y/N) n
 - 信號 firefox(5615) ? (y/N) y
 - 已砍掉

依部分名稱終止程序(pkill)

- 類似killall可以指定程式名稱，但pkill會以正規表示法(regex)的方式比對
 - 也就是可以模糊搜尋後殺掉程序
 - 所以也更加危險
 - -e (或 --echo) : 顯示砍了哪個程序
- 範例：
 - pkill -e firef.+ #我想要砍掉firefox相關的
 - 其實寫成 pkill -e firef 就可以砍了

終止視窗程序(xkill)

- 當視窗程序當掉，不容易找到其PID或名稱，尤其是在當機的時候
 - **xkill 可用來關閉視窗，不需PID或程序名稱**

```
$ xkill
Select the window whose client you wish to kill with button 1....
# 用滑鼠在X window的視窗上按一下，就會砍掉該視窗
```



- 請任意舉例一個可用來刪除Process的指令?
- 請說明什麼是pid及ppid?
- fg是讓背景程序拉回前景執行，那bg指令的目的為?
 - 讓前景程序丟到背景執行
 - 讓前景程序丟到背景暫停
 - 讓背景程序在背景執行
 - 讓背景程序在背景暫停

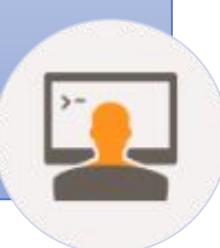
Module 27. 背景程序

- 27-1: 啟動背景程序
- 27-2: 檢視背景程序
- 27-3: 程序前背景切換

前景(Foreground)與背景(Background)

- 使用者在目前的終端機進行操作的程序
- 可直接讀取使用者的輸入

前景



Ctrl z (至背景暫停)

指令 & (至背景執行)

fg (至前景執行)

- 無法於終端機讓使用者進行操作的程序
- 不能等待使用者輸入，也無法以Ctrl c終止

背景



bg (背景執行)

範例: 把前景程序變成背景程序

- vim #於前景開啟vim
- 在vim裡按Ctrl z #把vim暫停並丟到背景去

[1]+ Stopped vim

並會返回到前景，可以輸入新指令了

[n]表示第n個工作

+ 代表最近一個被丟進背景的工作

- top #於前景開啟top
- 按Ctrl z #把top暫停並丟到背景去

[2]+ Stopped top

並會返回到前景，可以輸入新指令了

- sleep 1000 & #於背景執行sleep 1000
- 接下頁投影片

範例:查詢背景程序(jobs)

- **jobs #查詢背景程序狀態**

[1]- Stopped	vim
[2]+ Stopped	top
[3] Running	sleep 1000 &

- -l: 列出 job number、指令與PID
- -r: 列出正在背景執行(run)的程序
- -s: 列出正在背景暫停(stop)的程序

- **ps #若用ps會看到前景及背景的狀態**

PID	TTY	TIME	CMD
3565	pts/0	00:00:00	bash
3952	pts/0	00:00:00	vim
4135	pts/0	00:00:00	top
4163	pts/0	00:00:00	sleep
4186	pts/0	00:00:00	ps

範例：使用fg

- fg %1 #將job 1 (vim)切換至前景執行
 - 按下Ctrl z #再將vim丟回背景暫停
-
- fg %2 #將job 2 (top)切換至前景執行
 - 按下Ctrl z #再將top丟回背景暫停
-
- fg %3 #將job 3 (sleep 1000)切換至前景執行
 - 按下Ctrl z #再將sleep 1000丟回背景暫停

範例：使用bg

- jobs #查詢背景狀況

```
[1] Stopped vim
[2]- Stopped top
[3]+ Stopped sleep 1000 &
```

- bg %3 #讓job 3的狀態變成於背景執行中

```
[3]+ sleep 1000 &
```

- jobs

```
[1]- Stopped vim
[2]+ Stopped top
[3] Running sleep 1000 &
```

Module 28. 工作排程

- 28-1: 一次性工作排程
- 28-2: 例行性工作排程
- 28-3: 系統性工作排程

一次性的工作排程(at)

- at [-m] TIME #下達工作指令
 - -m: 將工作的輸出結果mail 紿下達指令的帳號
 - TIME指定執行時間
 - 定義可檢視/usr/share/doc/at-3.1.13/timespec, 或是man “at”

TIME格式	範例	說明
HH:MM	11:00	今天11點
HH:MM YYYY-MM-DD	04:00 2002-05-30	2002/5/30 4:00
HH[pm;am] [Month] [Day]	1pm May 30	5/30 下午1點

- atq #查看目前的工作排程
- atrm [工作編號] #刪除排程

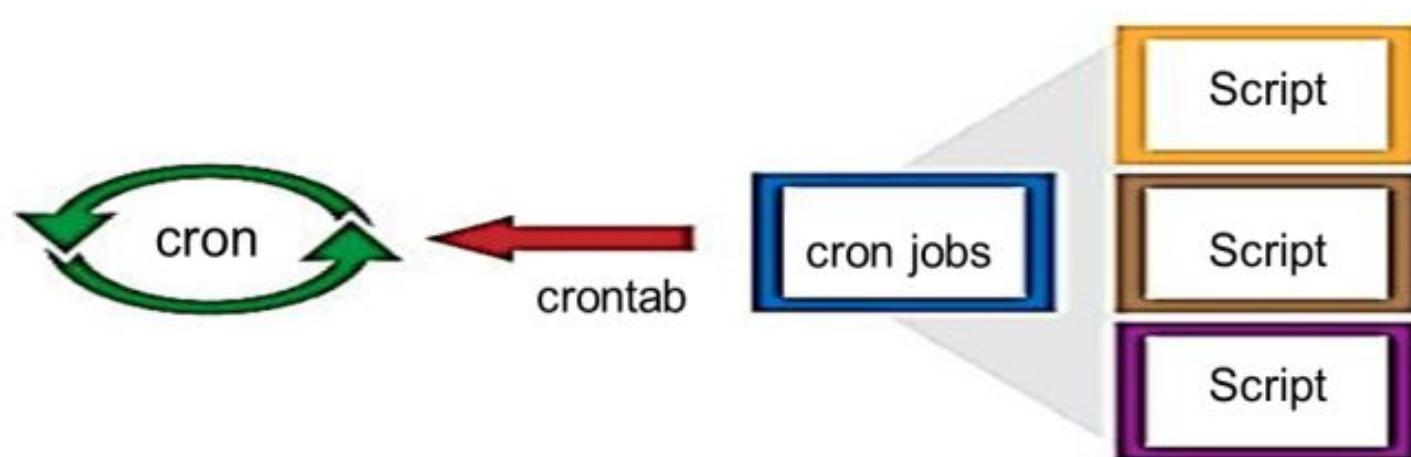
- 設定下午3點執行，印出兩個字串
 - 用-m所以執行結果會mail給user1

```
[ user1@localhost ~]$ at -m 3pm
at> echo helloworld
at> echo demo
at> <EOT>✉
job 1 at Tue Sep  3 15:00:00 2019
```

- atrm 1 #將job 1刪掉
- atq #檢視還剩下哪幾個工作

循環性的工作排程(crontab)

- crontab 可讓系統於指定時間(或週期)自動執行某些任務，例如：
 - 定期更新系統套件
 - 更新locate的索引資料庫 (updatedb)
 - 備份或清理檔案 (tar + gzip)
- 若工作任務較複雜，建議將多個指令寫成Script，設定時會較簡潔



crontab語法格式

分 時 日 月 星期 指令(或Script)

欄位	分	時	日	月	星期
範圍	0-59	0-23	1-31	1-12	0-7

特殊字符	說明
*	代表任何時刻
,	列舉時段，以逗號做分隔。例如1,3,5
-	代表一段時間範圍內。例如[9-11] 代表 9,10,11
/數字	亦即是『每隔單位間隔』的意思。例如*/3
@yearly	等同 0 0 1 1 *
@daily	等同 0 0 * * *
@hourly	等同 0 * * * *
@reboot	系統啟動時執行

crontab範例

crontrb範例	說明
0 22 * * 5 指令或腳本	每週五22點執行job1.sh
0 9,11 * * * 指令或腳本	於每日09:00及11:00 執行
*/5 * * * * 指令或腳本	每5分鐘執行
* * * * * 指令或腳本	每分鐘都執行
0 9-18 * * 1-5 指令或腳本	於每週一到週五的09:00至18:00 執行
@yearly 指令或腳本	於每年元旦執行該腳本
@reboot 指令或腳本	開機後執行該腳本

- 描述腳本位置時，建議使用**絕對路徑**
 - 例如 0 9 * * * bash /home/user1/job1.sh
- 無法使用「幾月幾號且為星期幾」

練習-設定工作排程

- 執行 `crontab -e` #編輯工作排程 
 - 預設會以 vi 開啟設定檔
 - 設定檔為 “/var/spool/cron/crontabs/使用者帳號”
- 寫入想執行的頻率與工作內容

#設定每2分鐘執行一次date指令，結果寫入log檔案

```
*/2 * * * * date >> /home/user1/mylog.txt
```

- 自行檢視執行結果
 - 查看 mylog.txt 內容

檢視或刪除工作排程

- 檢視工作排程清單
 - crontab -l
- 刪除所有的工作排程
 - crontab -r #會全部都刪除
- 若只要移除其中一項排程
 - 請執行 crontab -e 編輯即可



限定能使用crontab的帳號

- 預設系統上任何帳號皆可以使用crontab，若要進一步限定帳號，可編輯系統設定檔，有以下兩種：

/etc/cron.allow (白名單)

- 將可以使用 crontab 的帳號寫入其中
- 若不在這個檔案內的使用者則不可使用 crontab

/etc/cron.deny (黑名單)

- 將不可以使用 crontab 的帳號寫入其中
- 若未記錄到這個檔案當中的使用者，就可以使用 crontab

- 編輯 /etc/crontab 檔案 (需root)

分	時	日	月	星期	執行者身分	指令
---	---	---	---	----	-------	----

- vi /etc/crontab

SHELL=/bin/bash □ 使用哪種 shell 介面
PATH=/sbin:/bin:/usr/sbin:/usr/bin □ 設定執行檔搜尋路徑
MAILTO=root □ 若有額外STDOUT, 以 email 將資料寄給誰

Example of job definition:
.----- minute (0 - 59)
| .----- hour (0 - 23)
| | .----- day of month (1 - 31)
| | | .---- month (1 - 12) OR jan,feb,mar,apr ...
| | | | .--- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
| | | | |
* * * * * user-name command to be executed

0 9 * * * root updatedb #以root身分執行更新索引

- 也可將腳本置於下列目錄內，讓其自動執行
 - 可減少因系統關機而過期，導致工作沒有執行的狀況
 - 適合不要求精準的”執行時刻”，但”一定要執行”的任務
 - 若要檢視系統的週期設定，需使用root權限執行：
 - cat /etc/anacrontab

目錄	說明
/etc/cron.daily/	每天執行一次 未執行過排程則開機 5 分鐘後執行
/etc/cron.weekly/	每7天執行一次 未執行過排程則開機 25 分鐘後執行
/etc/cron.monthly/	每月執行一次 未執行過排程則開機 45 分鐘後執行

Module 29. 系統服務

- 29-1: systemctl 指令
- 29-2: 啟用(enable)/
停用(disable)服務
- 29-3: 啟動(start)/
停止(stop)服務

透過 systemctl 管理系統服務運行

- 語法 : `systemctl [動作] [服務名稱]` #需root權限
 - 以crond.service服務為例：

範例	說明
<code>systemctl stop crond.service</code>	停止該系統服務
<code>syetemctl start crond.service</code>	啟動該系統服務
<code>syetemctl restart crond.service</code>	重啟該系統服務
<code>syetemctl reload crond.service</code>	重新載入設定檔案, 不需停止服務
<code>syetemctl enable crond.service</code>	於下次開機時啟動服務
<code>syetemctl disable crond.service</code>	於下次開機時停止服務
<code>systemctl status crond.service</code>	檢視服務運行狀態

- 檢視服務運行狀態，不需要root權限
 - `systemctl status crond.service`

```
[iiiedu@localhost ~]$ systemctl status crond.service
crond.service - Command Scheduler
   Loaded: loaded (/usr/lib/systemd/system/crond.service; enabled)
   Active: active (running) since 五 2016-12-16 00:40:37 CST; 3 days ago
     Main PID: 1285 (crond)
        CGroup: /system.slice/crond.service
                  └─1285 /usr/sbin/crond -n
```

Module 30. 網路設定

- 30-1: 網路卡命名原則
- 30-2: 檢查網路設定
- 30-3: 變更網路設定

- 本章講述Linux相關網路設定指令及應用，若對網路原理尚不熟悉，建議可參考以下資源：
 - 維基百科：IP位址
 - <https://zh.wikipedia.org/wiki/IP地址>
 - 微軟：了解 TCP/IP 定址及子網路基本概念
 - <https://support.microsoft.com/zh-tw/kb/164015>
 - 鳥哥的 Linux 私房菜 第二章、基礎網路概念
 - http://linux.vbird.org/linux_server/0110network_basic.php
 - 鳥哥的 Linux 私房菜 第五章、Linux 常用網路指令
 - http://linux.vbird.org/linux_server/0140networkcommand.php

檢視或設定Linux主機名稱(hostname)

hostname

- 顯示主機名稱

hostnamectl set-hostname [NAME]

- 設定新主機名稱(需root)

- 檢視相關系統紀錄檔案

- 例如 cat /etc/hostname

/etc/hostname

- 記錄系統主機名稱

/etc/hosts

- 遠端主機名稱與IP的對應
- 127.0.0.1 localhost

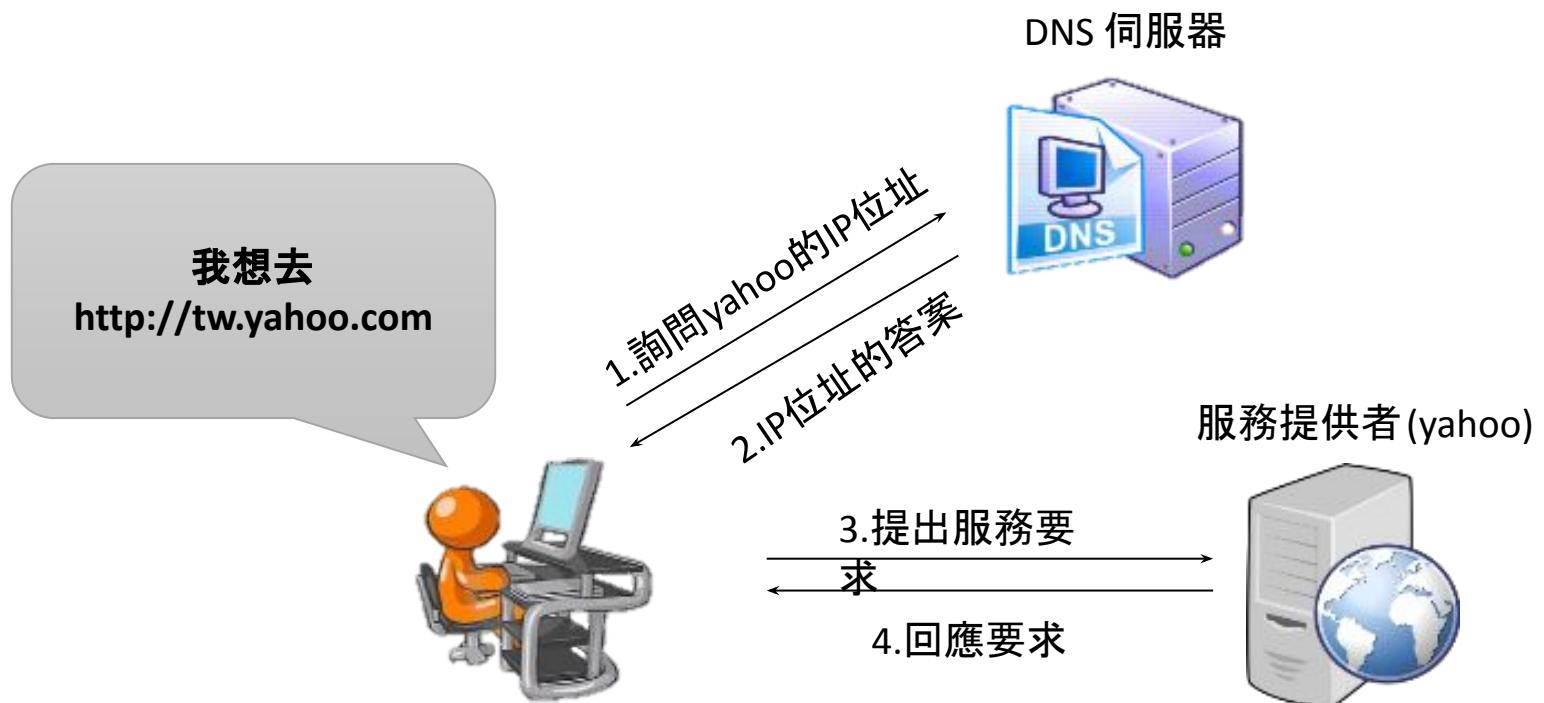


/etc/host.allow
/etc/host.deny

- 可設定哪些IP、主機、使用者帳號可否訪問或使用

- 向 DNS伺服器查詢Hostname對應的IP資訊

DNS供應商	IP
HiNet	168.95.1.1 及 168.95.192.1
Google public DNS	8.8.8.8 及 8.8.4.4 



- **/etc/resolv.conf**

- 此檔案可用來設定DNS用戶端要求名稱解析時，所定義的各項內容
- nameserver
 - 指定用戶端要求進行名稱解析的伺服器IP位址
 - 可指定多部DNS伺服器，使用者端會依序提出查詢要求

```
iiiedu@localhost:~  
檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)  
[ iiiedu@localhost ~]$ cat /etc/resolv.conf  
# Generated by NetworkManager  
nameserver 8.8.8.8
```

查詢主機名稱與IP位址的對應

host 查詢目標

- 查詢主機名稱與IP位址的對應

nslookup 查詢目標

- 查詢主機名稱與IP位址的對應

- host www.ncu.edu.tw #用主機查詢IP

```
www.ncu.edu.tw has address 140.115.17.151
www.ncu.edu.tw has address 140.115.17.152
www.ncu.edu.tw has address 140.115.17.36
www.ncu.edu.tw has address 140.115.17.37
www.ncu.edu.tw has address 140.115.17.38
```

- host 140.115.17.151 #用IP查詢主機

```
151.17.115.140.in-addr.arpa domain name pointer www-vm1.cc.ncu.edu.tw.
```

- nslookup www.ncu.edu.tw #用主機查詢IP
- nslookup 140.115.17.36 #用IP查詢主機

- Ethernet 

- 實體網路介面
- 經典命名方式為eth0、eth1等開頭的名稱
- 現在多改為使用enp0s3

- 本地迴路Loopback :lo

- 虛擬的網路介面，運用本地迴路機制，便可在主機上運行網路服務，期間不須安裝實體網路介面卡，也無須將該服務開放予主機所在網路
- 例如，在設定好本地安裝的網站後，可連線至 127.0.0.1(或http://localhost)來存取本地網站 

檢視網路介面的狀態(ifconfig)

網路
介面

位址

接收封包

傳送封包

本地
迴路

```
[iiiedu@localhost ~]$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe3a:dcd5 prefixlen 64 scopeid 0x20<link>
          ether 08:00:27:3a:dc:d5 txqueuelen 1000 (Ethernet)
            RX packets 43 bytes 6490 (6.3 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 71 bytes 7741 (7.5 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
          loop txqueuelen 0 (Local Loopback)
            RX packets 132 bytes 11556 (11.2 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 132 bytes 11556 (11.2 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

設定網路介面(ifconfig)

• ifconfig 網路卡介面 參數|位址 #需root

網路卡介面

- Ethernet : enp0s3或是eth0、eth1等
- Loopback : lo (virtual network interface)

參數 | 位址

- up / down : 啟動或關閉網路介面
- netmask : 子網路遮罩
- broadcast : 廣播位址
- 位址相關 : inet (TCP/IP), inet6 (IPv6)

• 以root執行：

指令範例	說明
ifconfig enp0s3 down	停用enp0s3網路介面
ifconfig enp0s3 192.168.0.1	設定IP為192.168.0.1
ifconfig enp0s3 192.168.0.1 netmask 255.255.0.0	設定IP及遮罩

編輯網路卡設定檔以設定IP

- 編輯設定檔 (需root權限)

- vi /etc/sysconfig/network-scripts/ifcfg-enp0s3

- 調整變數設定

- 關閉網路卡

- ifdown enp0s3

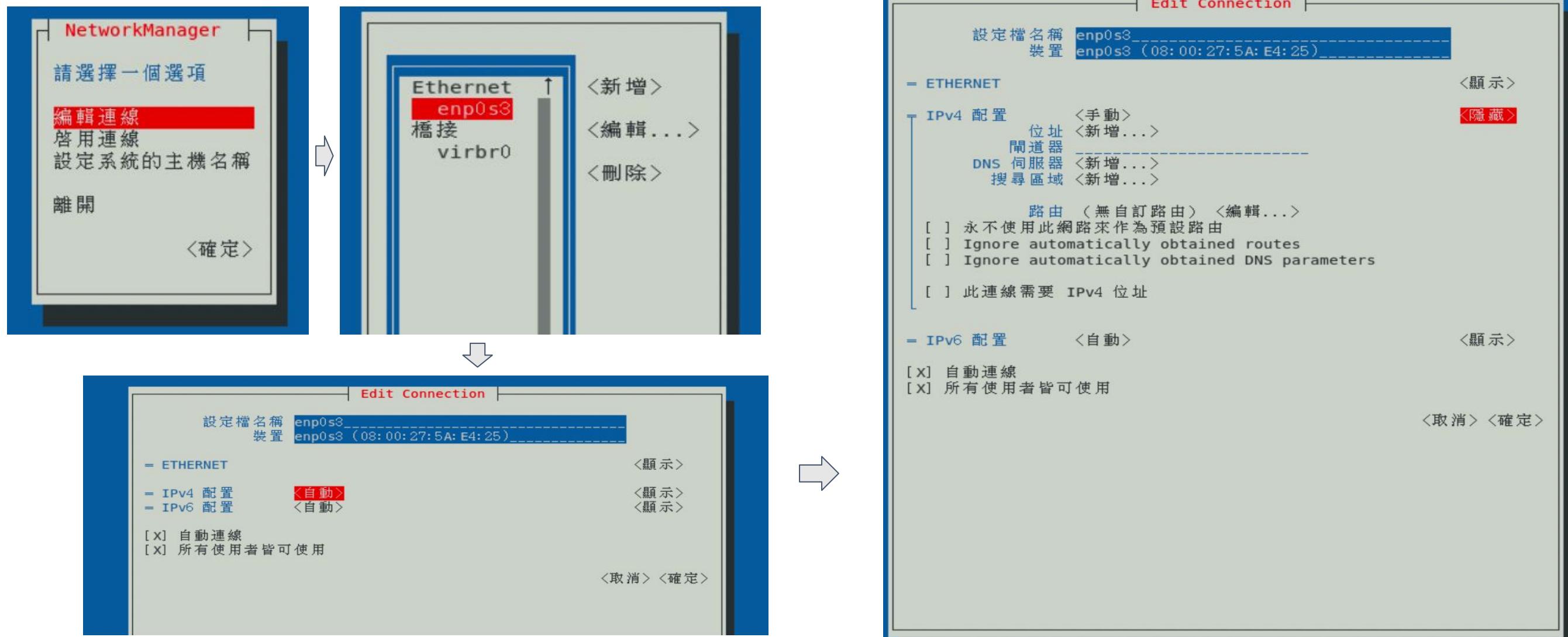
- 啟用網路卡

- ifup enp0s3

```
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=static
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
NAME=enp0s3
UUID=fc289d60-31ad-4d3c-a668-14d7445efa41
DEVICE=enp0s3
ONBOOT=yes
IPADDR=10.3.111.207
GATEWAY=10.3.111.254
NETWORK=10.3.111.0
NETMASK=255.255.255.0
DNS1=8.8.8.8 DNS2=9.9.9.9
```

設定網路介面(nmtui)

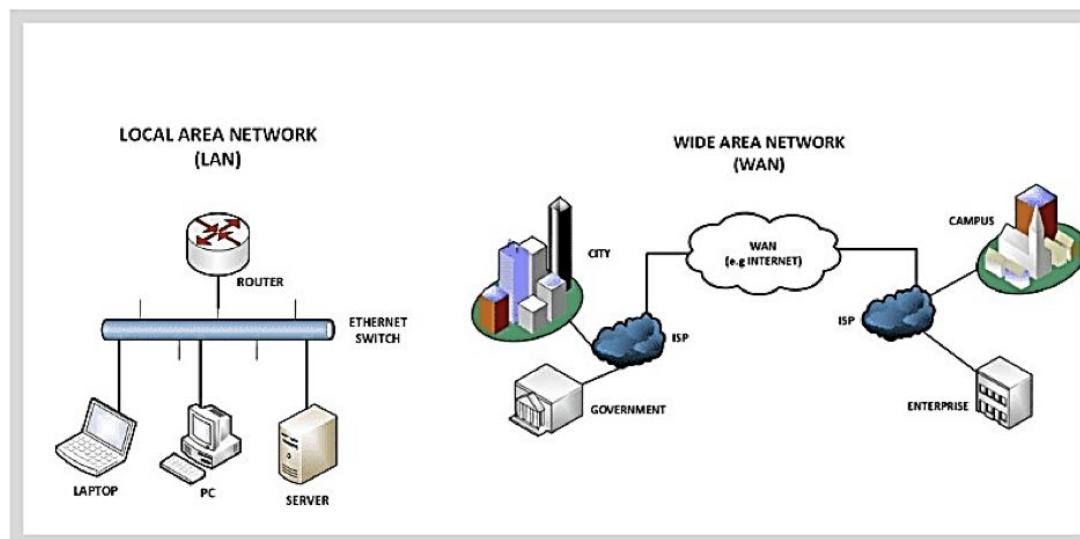
- 在終端機輸入指令nmtui後，可以直接受編輯網路



Module 31. 網路架構與連結埠

- 31-1: 區域網路與網際網路
- 31-2: 連接埠、NAT 網路
- 31-3: Virtual box 的網路架構

- 區域網路(Local Area Network, LAN)
 - 連結在限定範圍(如辦公室)之內的資訊設備，彼此傳遞資訊的網路
- 廣域網路(Wide Area Network, WAN)
 - 連結廣闊的地理區域範圍的資訊設備，例如全台各地戶政機關網路
- 網際網路(Internet) 
 - 連結範圍橫跨全世界的超大型廣域網路，讓全球任何一台能上網的電腦彼此溝通



<https://purple.ai/blogs/whats-the-difference-between-a-lan-and-a-wan/>

連結埠(TCP/IP Port)

- TCP/IP協議中的Port, 主要用來識別在主機上的網路應用類型
 - Port號的範圍從0到65535 ☺
 - 比如瀏覽網頁服務時, Client端會連到HTTP Server主機的80 Port



<https://microchipdeveloper.com/tcpip:tcp-ip-ports>

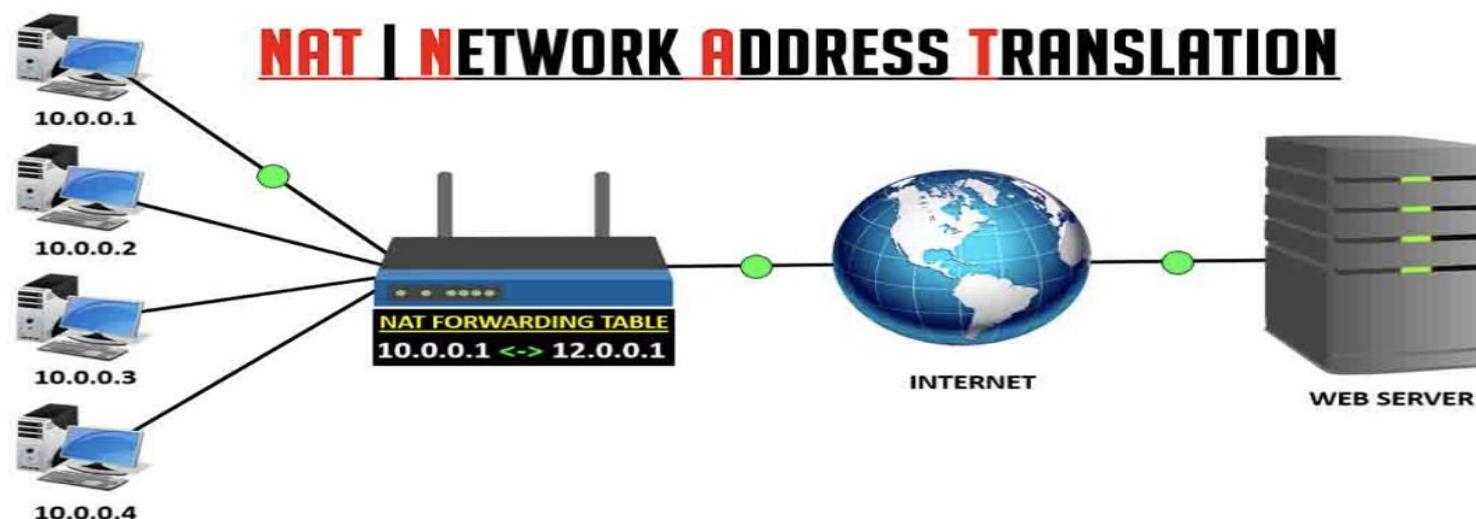
- less /etc/services #檢視埠, protocol 網路服務
- 常見的網路服務埠(通常小於1024):

PORT	服務	說明
21	FTP	檔案傳輸協定的control channel
22	SSH	較為安全的遠端連線
23	Telnet	遠端連線
25	SMTP	郵件傳遞協定
53	DNS	Domain Name Server
80	WWW	網頁連線 
110	POP3	郵件收信協定
443	HTTPS	有加密的WWW
3306	MySQL	MySQL資料庫



● Network Address Translation(NAT)

- 是一種在IP封包通過網路設備時重寫來源IP或目的的IP技術
- 可用來節省使用Public IP的數量
 - 通過NAT轉換，接入只網路可以使用Private IP
 - 對外連接時由路由器(router)對應Private IP (例如10.0.0.1)與Public IP (例如12.0.0.1) 的關係，修改傳輸的IP包上的位址





- NAT(預設)
 - 依附在Host的網路之下，只要Host能上網，Guest就能上網
 - 各Guest各自使用自己的NAT router



- NAT 網路
- 各Guest共同使用一台NAT router



- 橋接介面卡 (Bridged)
 - Guest與Host的同網域，但各自獨立
 - 同網域內的主機可直接跟Guest連線

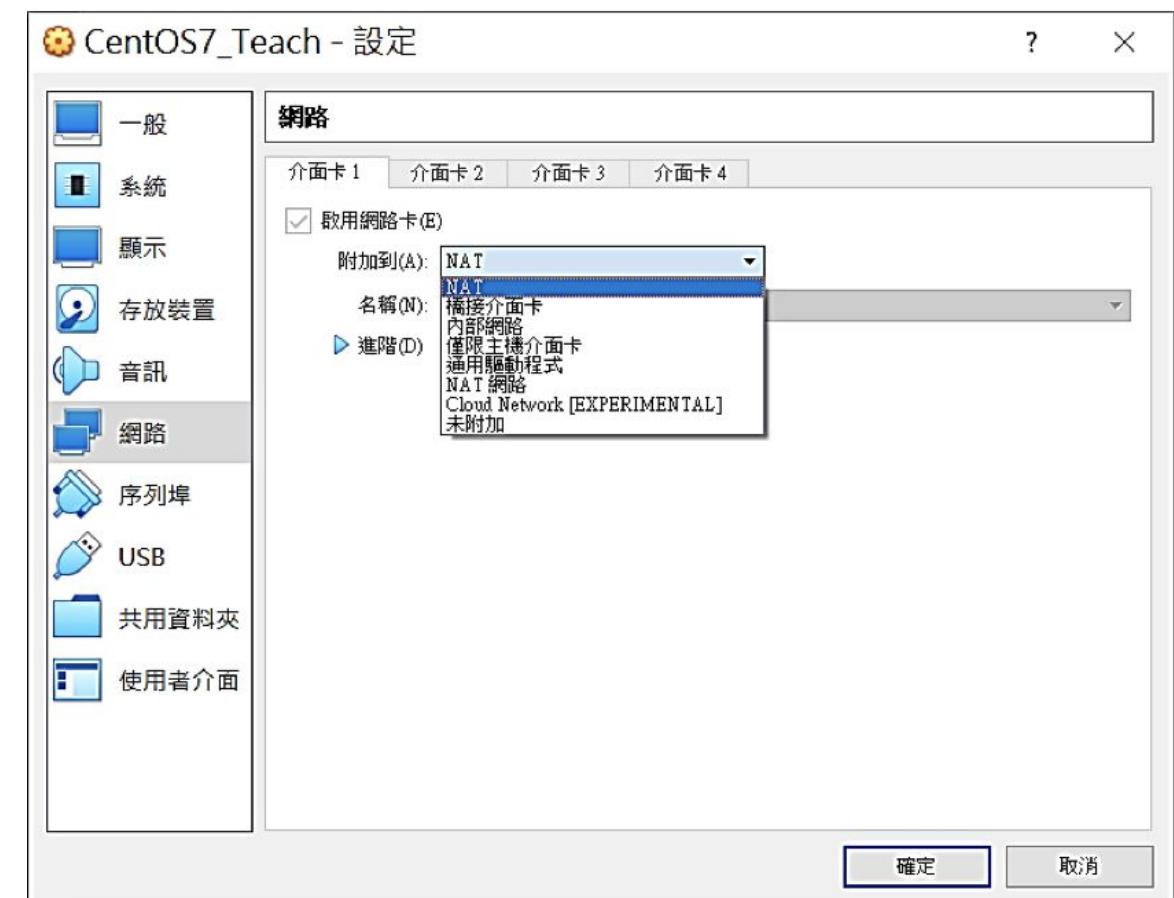


- 內部網路
 - 只允許Guest之間互相連接
 - 無法連到Host，也沒法連到外部網路



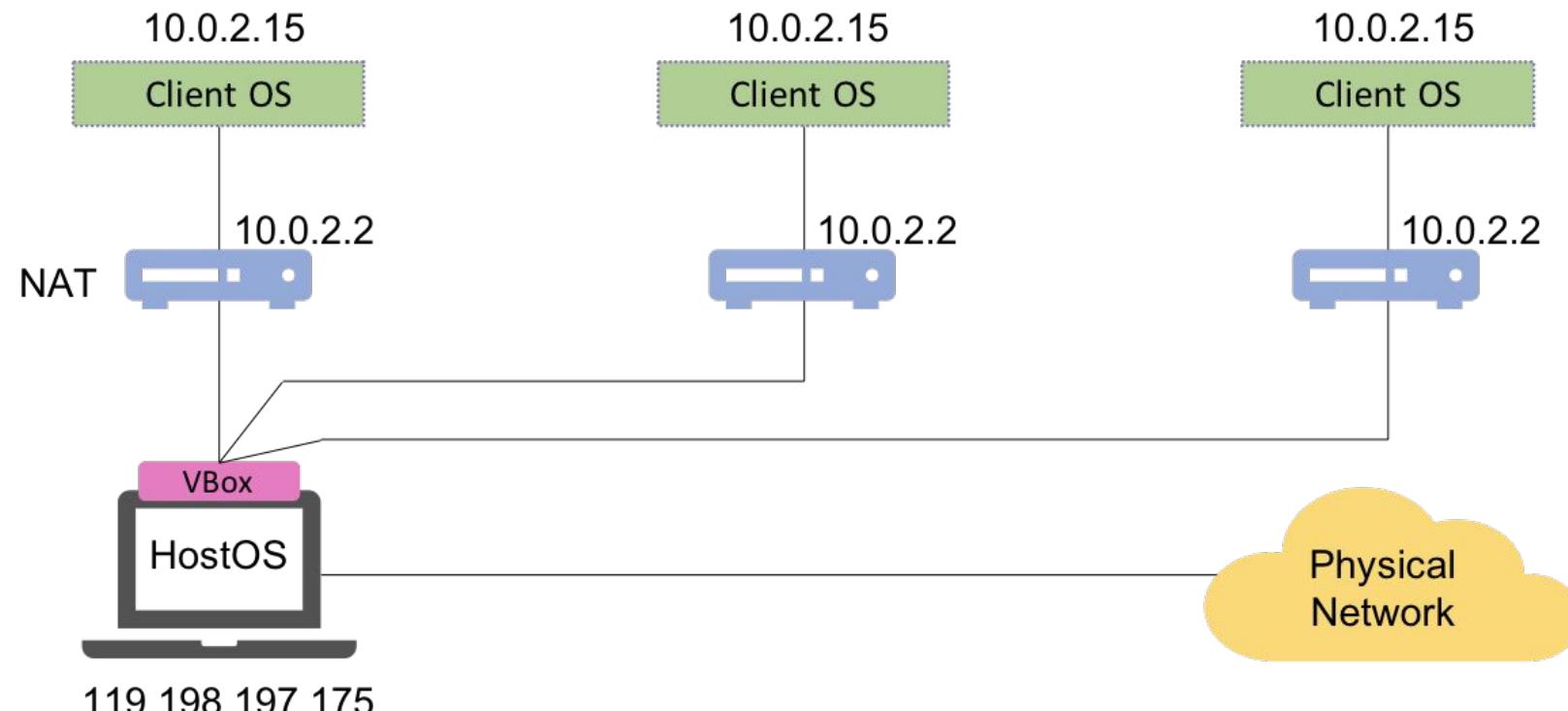
- 僅限主機
 - Host可以跟一群Guest形成一個網路
 - 無法直接連到外部網路

VirtualBox 預設的網路模式為NAT



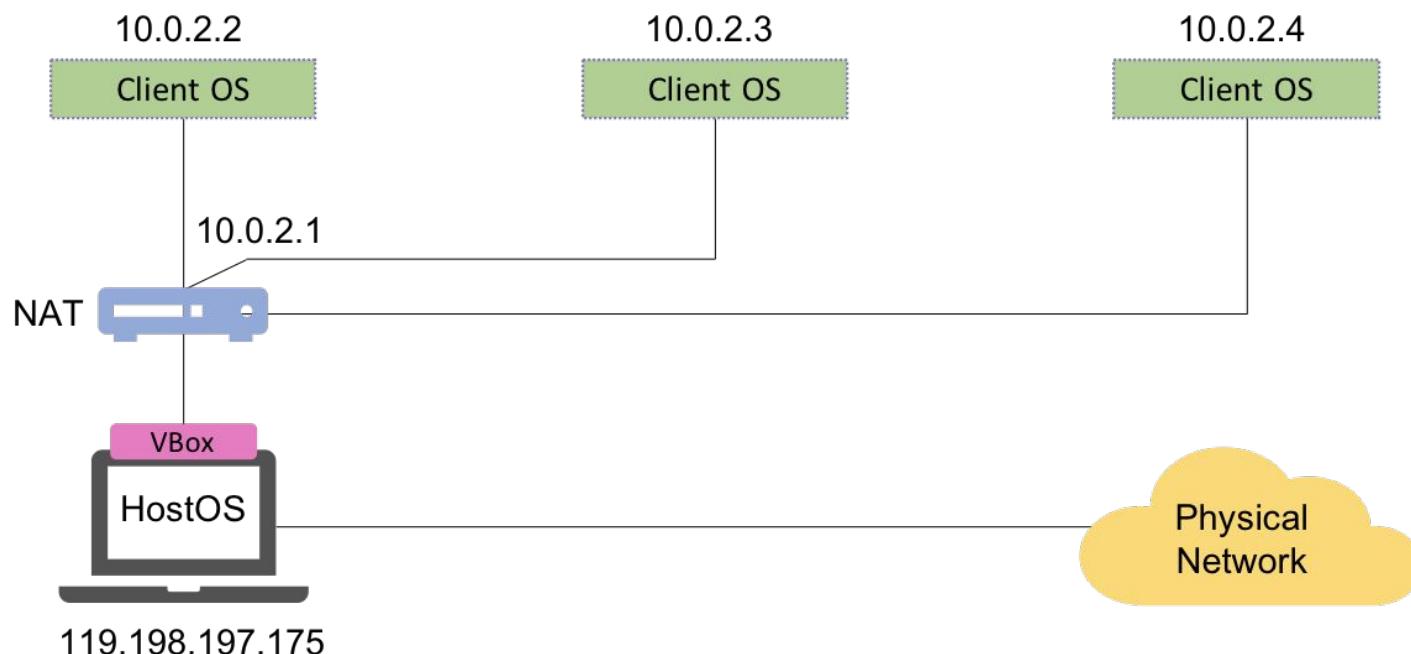
範例:VirtualBox NAT模式

- 預設配發給虛擬主機的虛擬IP為10.0.2.15
- 預設閘道(Gateway) IP為10.0.2.2
- 預設遮罩(Mask)為255.255.255.0



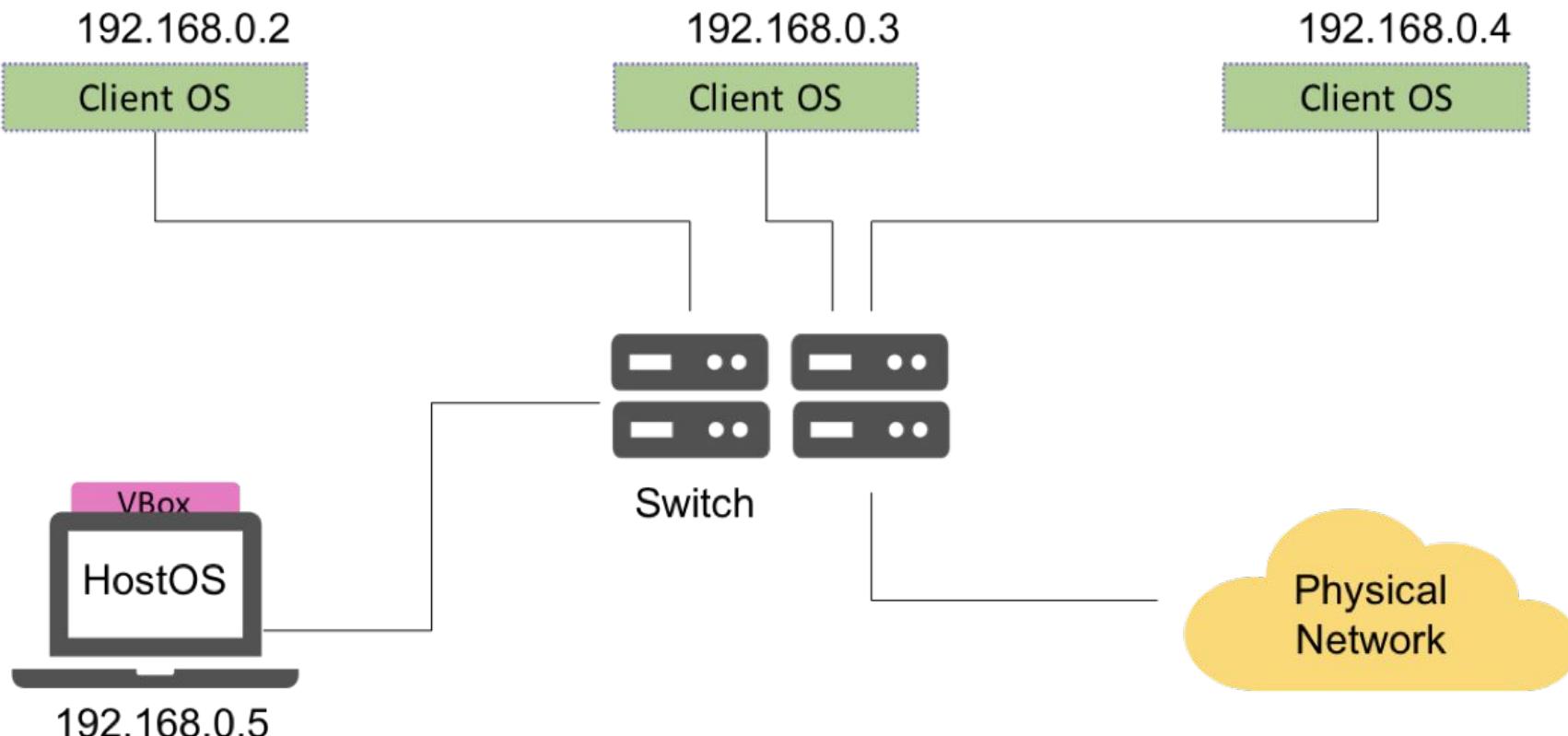
範例:VirtualBox NAT網路模式

- 各Guest共同使用一台NAT router (需手動建立)
 - 預設閘道(Gateway) IP為10.0.2.1
 - 預設遮罩(Mask)為255.255.255.0
 - 自動取得IP
- 使用情境類似家用無線基地台



範例:Bridged Networking網路模式

- 在網路環境中將Guest視同與Host同等地位的主機
- 一般會比照Host的閘道、遮罩、DNS等網路設定，但只有IP不能相同



© toto.zhang@gmail.com

296

Module 32. 遠端連線

- 32-1：網路相關工具指令
- 32-2：SSH 連線
- 32-3：使用 SSH Key 登入

顯示連線是否建立(ping)

- 傳送網路封包至對方主機，測試是否可連線

ping www.google.com #以Ctrl-c中斷測試
• ping -c 3 www.google.com #只測試三次

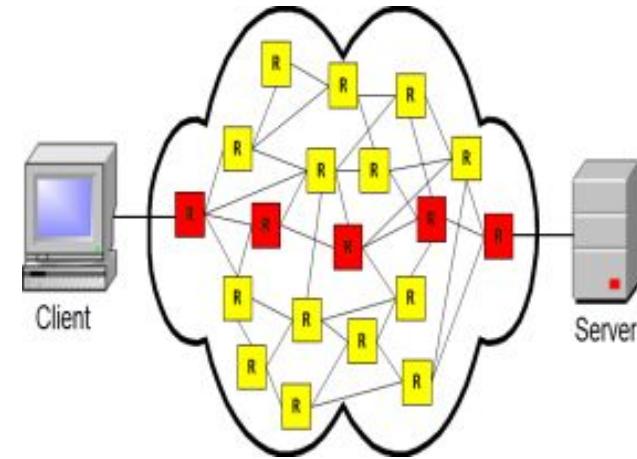


```
user1@Vbox:~$ ping -c 3 www.google.com
PING www.google.com (210.61.221.178) 56(84) bytes of data.
64 bytes from 210-61-221-178.HINET-IP.hinet.net (210.61.221.178): icmp_req=1 ttl=128 time=2.29 ms
64 bytes from 210-61-221-178.HINET-IP.hinet.net (210.61.221.178): icmp_req=2 ttl=128 time=2.06 ms
64 bytes from 210-61-221-178.HINET-IP.hinet.net (210.61.221.178): icmp_req=3 ttl=128 time=2.16 ms
--- www.google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 2.061/2.175/2.296/0.103 ms
```

- 註:TTL (Time To Live, 存活時間)
 - 指一個封包在通過一個網路時，最長可以逗留的時間
 - 封包經過一個路由器，ttl減1，ttl為0時，主機便取消封包

顯示封包歷程(traceroute)

- 本機 -> 遠端主機
 - 顯示經過節點及每一段的連線速度，以了解網路擁塞情形
 - 為避免偏頗，每個連線會送3個封包
 - traceroute tw.yahoo.com



```
traceroute to tw-tpc-fo.fyap.b.yahoo.com (203.84.202.164), 30 hops max, 38 byte packets
 1 ip-66-33-193-3 (66.33.193.3) 0.674 ms 0.604 ms 0.477 ms
 2 ip-66-33-201-113 (66.33.201.113) 0.470 ms 0.660 ms 0.459 ms
 3 border11.te8-4.newdream-8.lax.pnap.net (216.52.220.145) 0.740 ms 0.658 ms 0.476 ms
 4 core2.dol-20g-bbnet1.lax.pnap.net (216.52.255.2) 0.499 ms 0.685 ms 0.707 ms
 5 dpr1-so-3-1-0.losangelesequinix.savvis.net (204.70.194.174) 0.639 ms 0.653 ms 0.714 ms
 6 crl-pos-0-2-5-0.lay.savvis.net (204.70.203.46) 0.731 ms 0.635 ms 0.724 ms
 7 208.172.44.34 (208.172.44.34) 12.229 ms 12.280 ms 12.085 ms
 8 * * *
 9 * * *
```

顯示*表示封包該路由器忙碌中或是被防火牆擋住

檢視本機的網路連線狀況(netstat)

- 例如可用來查詢目前有誰連結到此主機



選項	說明
-a	顯示所有網路埠
-n	以數字顯示位址, 速度較快
-t	TCP
-u	UDP
-l	listening
-p	PID (用root才看的到)

指令範例	說明
netstat -na less	以數字顯示所有網路通訊
netstat -na grep "ESTABLISHED"	顯示所有已經建立的連線
netstat -na grep :80	顯示所有port 80的連線
netstat -tulnp less	檢視目前已經啟動的網路服務
netstat -s less	顯示網路流量統計資訊
sudo netstat -p less	顯示PID (需root權限)

netstat範例

• netstat | less

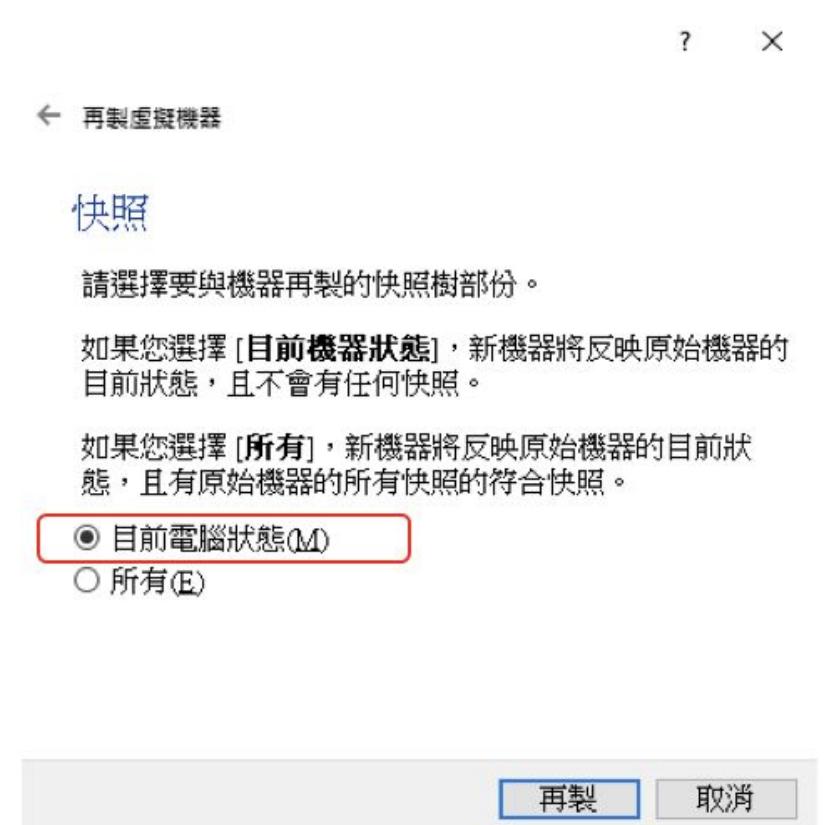
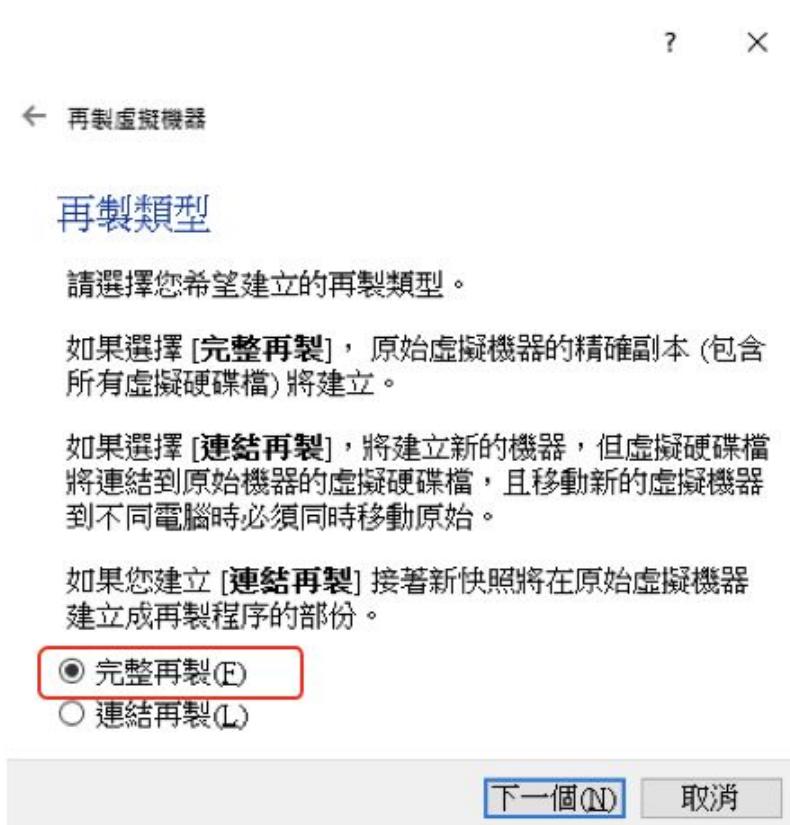
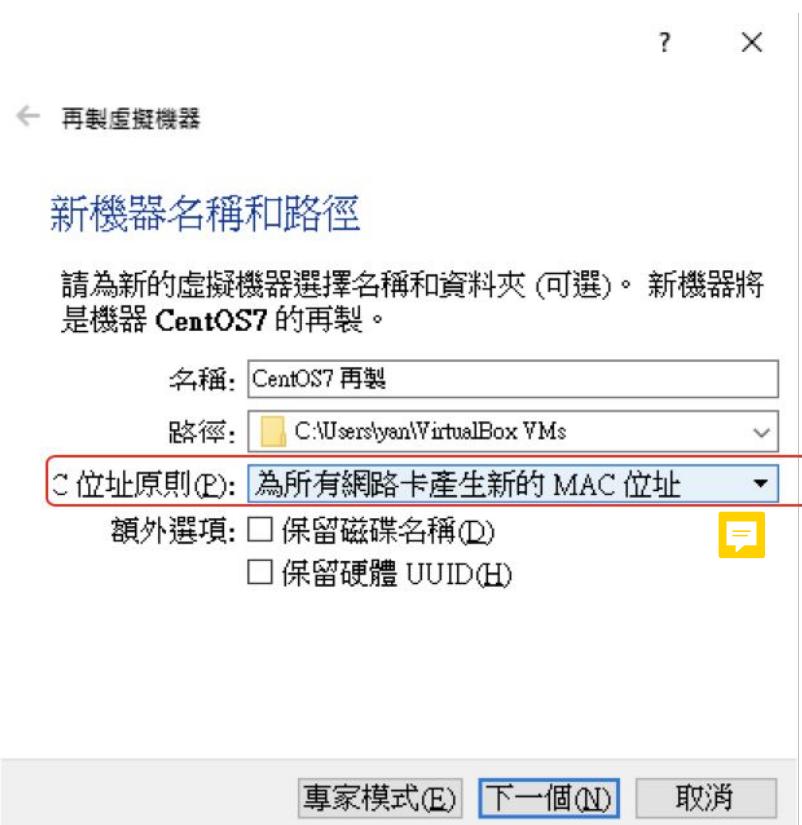
- 遠端主機(10.2.0.68)連線到本機(192.168.120.204)
- 使用了SSH協定
 - Client端隨機取一個大於1024以上的port進行連線
 - 只有root可以啟動小於1024 以下的port

Active Internet connections (w/o servers)						
Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
tcp	0	268	192.168.120.204:ssh	10.2.0.68:62420	ESTABLISHED	7737/sshd: user1
Active UNIX domain sockets (w/o servers)						
Proto	RefCnt	Flags	Type	State	I-Node	Path
unix	2	[]	DGRAM		1491	@/org/kernel/udev/udevd
unix	4	[]	DGRAM		7337	/dev/log
unix	3	[]	STREAM	CONNECTED	7287	
.... 以下省略....						

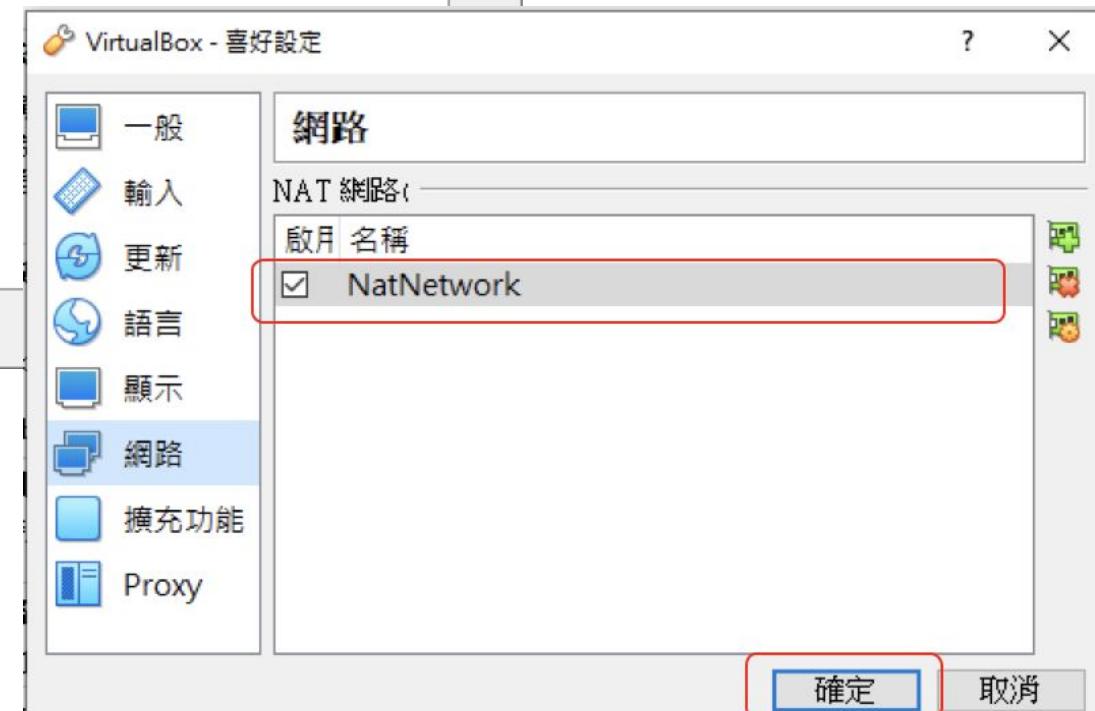
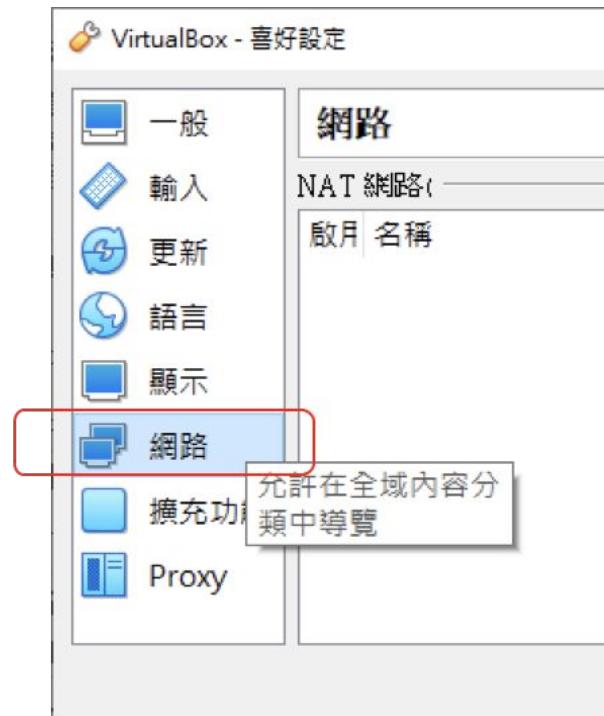
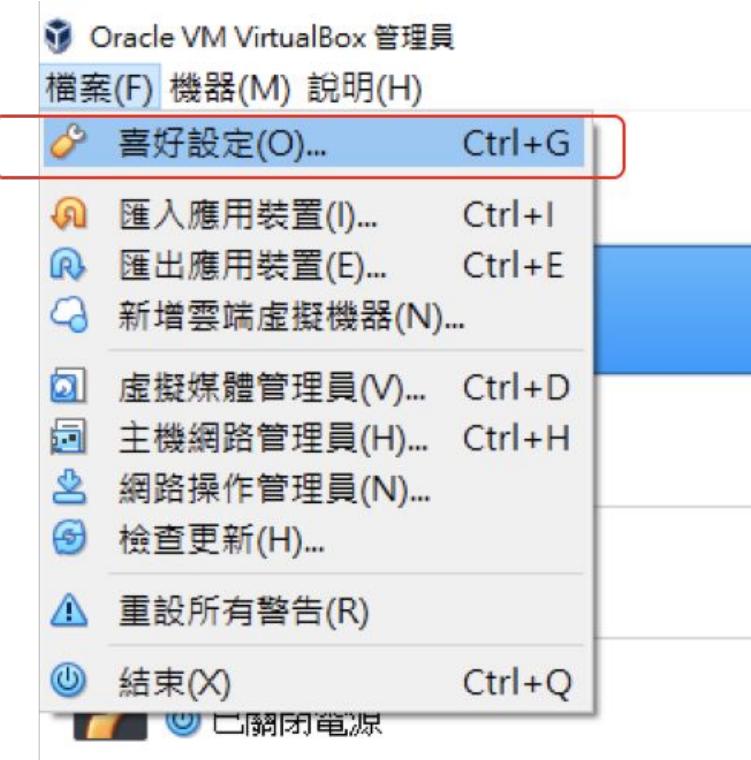
- SSH是一種加密的網路協定，常用來進行遠端主機的連線操作
- Linux主機已安裝openssh-server套件(CentOS7已內建)
 - sudo yum install openssh-server #若要自行安裝的話
- 連線指令
 - ssh 帳號@遠端主機IP #需要遠端主機的帳號及密碼
- Windows使用者該如何連線？
 - 可安裝pietty或putty等連線軟體



- 我們要準備兩台Linux主機
 - 利用VirtualBox 再製(Clone)功能，新增一台Linux
 - 位址原則:為所有網路卡產生新的MAC位址



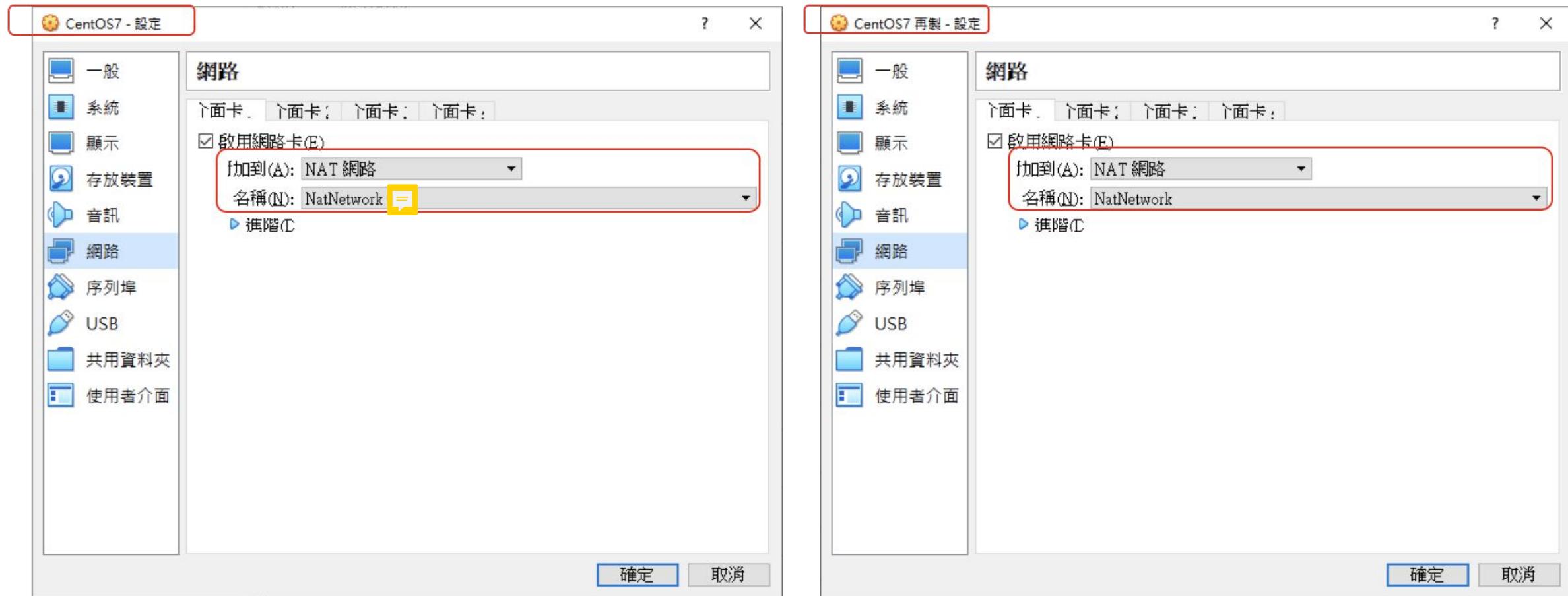
- 調整VirtualBox 喜好設定 -> 網路右方+圖示 -> 勾選NatNetwork -> 確定



NAT網路-實驗環境準備(3/3)

- 分別調整兩台Linux的網路設定模式，從「NAT」改為「NAT網路」

- CentOS7 -> 設定 -> 網路 -> NAT網路，名稱 NatNetwork -> 確定。完成後開機。
- CentOS7再製 -> 設定 -> 網路 -> NAT網路，名稱 NatNetwork -> 確定。完成後開機。

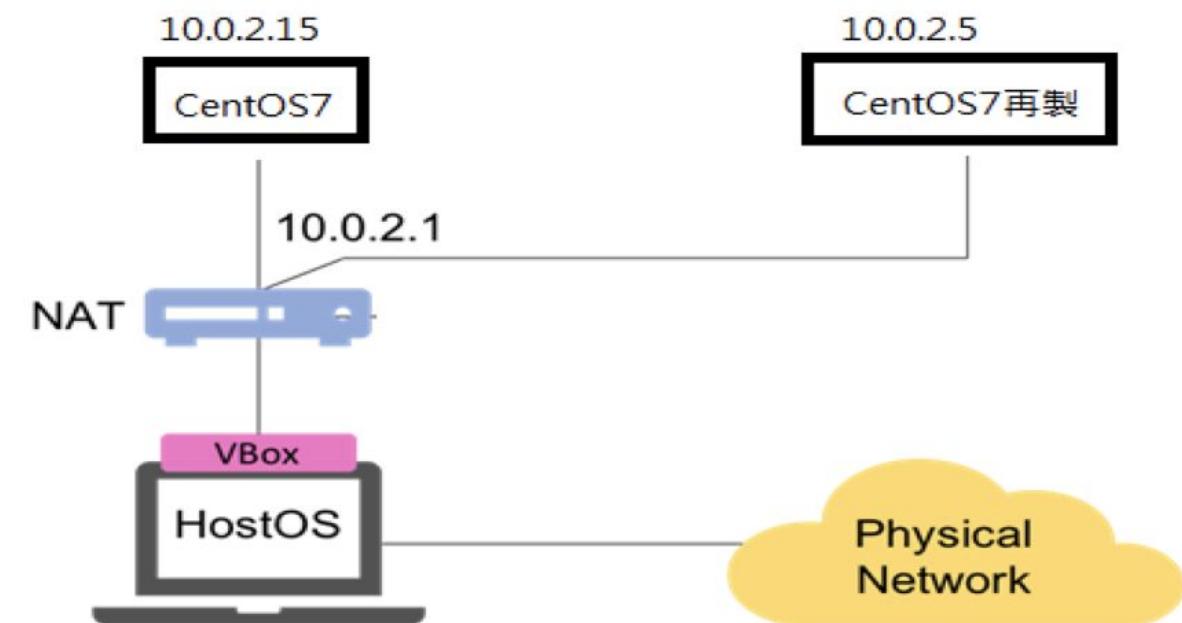


NAT網路實驗環境架構圖

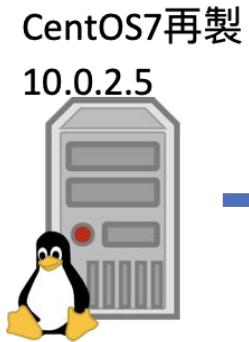
- 請分別查出兩台Linux取得的IP，例如下圖

```
[user1@centos7 ~]$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
          inet6 fe80::cc29:ab51:6ed9:5cb8 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:5a:e4:25 txqueuelen 1000 (Ethernet)
              RX packets 678 bytes 814297 (795.2 KiB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 493 bytes 36421 (35.5 KiB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- 確認都取得IP後，可以開始SSH連線了



- ssh 對方開給你的帳號@對方的IP
- 從CentOS7 再製 連線到 CentOS7



ssh user1@10.0.2.15



新主機第一次連線時會有安全性警告，問你信
不信任這台主機，此時輸入 yes 即可連線

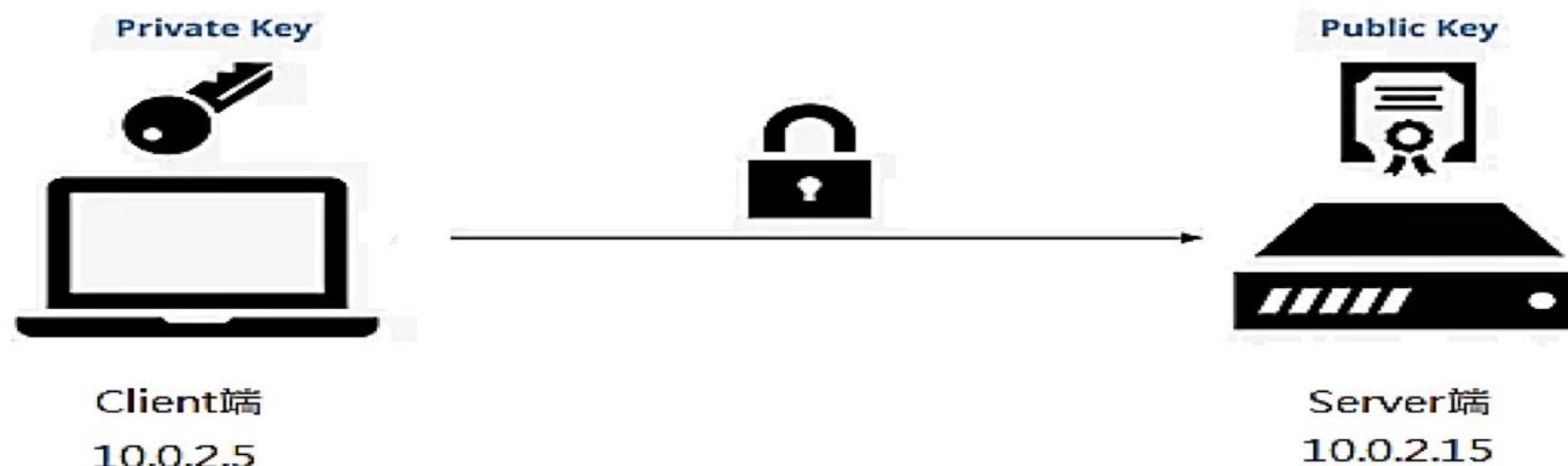
```
user1@localhost:~  
檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)  
[user1@localhost ~]$ ssh user1@10.0.2.15  
The authenticity of host '10.0.2.15 (10.0.2.15)' can't be established.  
ECDSA key fingerprint is 8e:96:39:54:06:3a:44:3d:86:b2:53:f6:92:d4:36:60.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '10.0.2.15' (ECDSA) to the list of known hosts.  
user1@10.0.2.15's password:  
Last login: Mon Jun 28 14:58:55 2021  
[user1@localhost ~]$ ls  
下載 公共 圖片 影片 文件 桌面 模板 音樂  
[user1@localhost ~]$ mkdir dir99  
[user1@localhost ~]$ ls  
dir99 下載 公共 圖片 影片 文件 桌面 模板 音樂  
[user1@localhost ~]$ exit  
logout  
Connection to 10.0.2.15 closed.  
[user1@localhost ~]$
```

1.請問: dir99是新增在哪一台主機上?

答案： 10.0.2.15

2.若要登出可以執行exit

- 先於Client端產生Key-Pair (私鑰+公鑰)
 - 彼此互相搭配且唯一
- 將公鑰上傳到Server端
- 連線時, Client端憑著私鑰當作一種信物, 讓Server端驗證通過後, 就可以直接放行而不用密碼



SSH登入免輸入密碼設定步驟

1. ssh-keygen #產生Key-Pair, 過程中直接一直按 enter 即可
2. ssh-copy-id user1@10.0.2.15 #將公鑰傳到Server
3. 以後可直接以user1免密碼登入Server ☺ ☺

```
user1@localhost:~  
檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)  
[user1@localhost ~]$ ssh-keygen  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/user1/.ssh/id_rsa):  
Created directory '/home/user1/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/user1/.ssh/id_rsa.  
Your public key has been saved in /home/user1/.ssh/id_rsa.pub.  
The key fingerprint is:  
7e:d1:61:af:56:73:86:8e:2d:2e:53:b2:94:e1:c0:a8 user1@localhost.localdomain  
The key's randomart image is:  
++[ RSA 2048]---+  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
+-----+[user1@localhost ~]$
```

```
[user1@localhost ~]$ ssh-copy-id user1@10.0.2.15  
The authenticity of host '10.0.2.15 (10.0.2.15)' can't be established.  
ECDSA key fingerprint is 8e:96:39:54:06:3a:44:3d:86:b2:53:f6:92:d4:36:60.  
Are you sure you want to continue connecting (yes/no)? yes  
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter  
out any that are already installed  
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt  
ed now it is to install the new keys  
user1@10.0.2.15's password:  
  
Number of key(s) added: 1  
  
Now try logging into the machine, with: "ssh 'user1@10.0.2.15'"  
and check to make sure that only the key(s) you wanted were added.  
  
[user1@localhost ~]$ ssh user1@10.0.2.15  
Last login: Mon Jun 28 13:53:39 2021  
[user1@localhost ~]$
```

Module 33. 檔案傳輸

33-1 : wget

33-2 : scp與sftp

33-3: filezilla

檔案下載工具(wget)



- 支援HTTP, HTTPS, FTP 等網路協定

- wget http://www.google.com #抓取google首頁
- wget -m https://www.kernel.org/ #鏡像, 也就是抓取整個網站
- wget -r -A “*.mp3” http://address #抓取附檔名為mp3的檔案
- wget ftp://user:password@addr/files/ #連線至FTP抓檔案

選項	說明
-r	遞迴下載, 把文件中所有的連結都下載回來
-N	只下載有更新的文件
-m	鏡像, 相當同時使用-r和-N
-P	指定存放路徑
-c	設定續傳功能
-np	不下載目標目錄的parent或其他目錄
-A	接受的檔案樣式



wget 說明手冊: <https://www.gnu.org/software/wget/manual/>

311

- sftp 對方開給你的帳號@對方主機位址
 - get、put、exit
 - 檔案會下載至本機的當前目錄

CentOS7再製
10.0.2.5CentOS7
10.0.2.15

```
[user1@localhost tmp]$ sftp user2@192.168.0.112
user2@192.168.0.112's password:
Connected to 192.168.0.112.
```

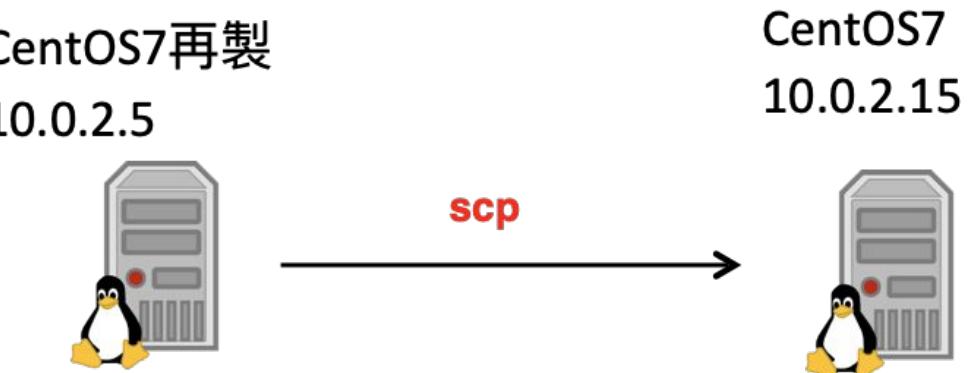
```
sftp> ls
test.txt 下載 公共 圖片 文件 桌面
```

```
sftp> get test.txt #下載檔案
Fetching /home/user1/test.txt to test.txt
/home/user1/test.txt 100% 3486 3.4KB/s 00:00
```

```
sftp> exit      #結束sftp
[user1@localhost tmp]$
```

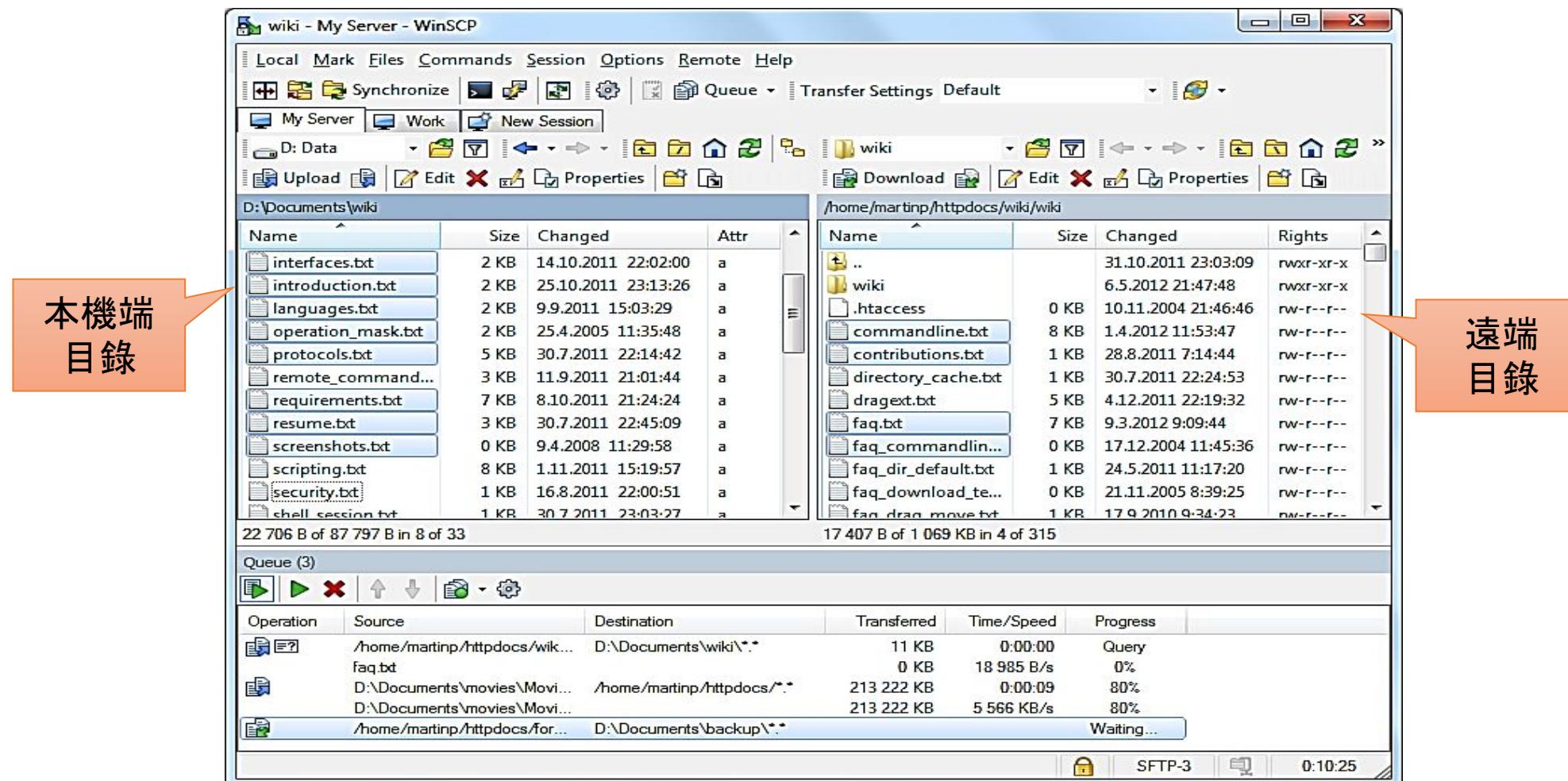


- 若是撰寫Script, 或是已知道遠端主機上的檔案位置, 則可改用scp指令
- 範例：
 - 下載text2.txt至本地端的/tmp/
 - scp user2@192.168.0.112:/home/user2/test2.txt /tmp
 - 上傳本地端test2.txt至遠端/home/user2
 - scp test2.txt user2@192.168.0.112:/home/user2/



Windows 與 Linux 之間的檔案傳輸

- Windows 可用 filezilla 或 winscp 傳輸 Linux 主機內的檔案
 - <https://filezilla-project.org/>
 - <http://winscp.net/>



- 連線至遠端Linux主機進行系統維運，建議的指令為？
 - telnet
 - **ssh**
 - bash
 - scp
- 要抓取遠端Linux主機內的檔案到本地電腦，建議的指令為？
 - telnet
 - wget
 - **sftp**
 - ssh

Module 34. bash 變數

- 34-1: 環境變數與區域變數
- 34-2: 檢查變數指令
- 34-3: 重要環境變數

- 變數是以一個固定字串來代表不固定的內容
- 變數名稱只能用英文字母、數字與底線
- 第一個字元不能是數字
- 自行定義變數慣例用小寫

- 檢視所有環境變數值
 - export
- 印出單一變數值
 - echo \${變數名稱} 
 - 或是 echo \${變數名稱}

```
[user1@localhost ~]$ echo $HISTSIZE  
1000  
[user1@localhost ~]$ echo ${HISTSIZE}  
1000
```

- 若變數不存在，則不會印出任何值(也沒有錯誤訊息)

```
[user1@localhost ~]$ echo $qwerasdf  
[user1@localhost ~]$ █
```

- var=value #**設定變數=變數值**
- 變數預設視為字串形式
- 等號兩邊不能接空白字元

```
[user1@localhost ~]$ 123foo=test
bash: 123foo=test: 找不到指令...
[user1@localhost ~]$ foo =a
bash: foo: 找不到指令...
[user1@localhost ~]$ foo= a
bash: a: 找不到指令...
[user1@localhost ~]$ foo=123
```

- 變數值內若需要空白字元，可使用單引號 ‘ 或雙引號 “ 標註範圍

```
[user1@localhost ~]$ name=CoCo Lee
bash: Lee: 找不到指令...
[user1@localhost ~]$ name="CoCo Lee"
[user1@localhost ~]$ █
```

單引號與雙引號

- 以下範例，使用單引號會印出原始字串而非變數值

```
[user1@localhost ~]$ echo $HISTSIZE  
1000  
[user1@localhost ~]$ echo ${HISTSIZE}  
1000  
[user1@localhost ~]$ echo "$HISTSIZE"  
1000  
[user1@localhost ~]$ echo '$HISTSIZE'  
$HISTSIZE
```

- 以下範例，使用雙引號才能保留原始空白

```
[user1@localhost ~]$ name="CoCo Lee"  
[user1@localhost ~]$ echo $name  
CoCo Lee  
[user1@localhost ~]$ echo ${name}  
CoCo Lee  
[user1@localhost ~]$ echo "$name"  
CoCo Lee  
[user1@localhost ~]$ echo '$name'
```

- 特殊符號可使用跳脫字元 (或使用單雙引號)

- \'表示無特殊意義的單引號
- \"表示無特殊意義的雙引號
- \\ 表示無特殊意義的右斜線



- 使用範例

- day=X\'mas
- day="X'mas"

- A=100 #將變數A設定為字串"100"
- B=A #將變數B設定為字串"A"
- C=\$A #將變數C設定為變數A的值
- echo \$A
 - 100
- echo \$B
 - A
- echo \$C
 - 100

```
[user1@localhost ~]$ A=100
[user1@localhost ~]$ B=A
[user1@localhost ~]$ C=$A
[user1@localhost ~]$ echo $A
100
[user1@localhost ~]$ echo $B
A
[user1@localhost ~]$ echo $C
100
```

- **unset var** #移除變數
- 範例
 - car="benz"
 - unset car
 - echo \$car -> 會顯示空白
- 練習：
 - 已知A="100"且 B="A"
 - 若執行unset \$B
 - echo \$A 結果為？
 - echo \$B 結果為？

```
[i i i edu@localhost ~]$ A="100"
[i i i edu@localhost ~]$ B="A"
[i i i edu@localhost ~]$ unset $B
[i i i edu@localhost ~]$ echo $A
[i i i edu@localhost ~]$ echo $B
A
```

變數設定練習

- song=Tom's pen
 - > 按Ctrl-c跳出
- song="Tom's pen"
- echo \$song
 - Tom's pen
- song="Tom's pen'"
 - > 按Ctrl-c跳出
- song=Tom\'s\ pen
- echo \$song
 - Tom's pen
- foo="cil"
- song="Tom's pen\$foo"
- echo \$song
 - Tom's pencil

- 可利用變數提高操作方便性
 - 例如若常需要去某個目錄或檔案時，可替該路徑設定變數
 - work=/tmp/myproject #設定work變數
 - cd \$work #切換至/tmp/myproject
- 如何擴充變數值內容?
 - 若想在原有變數值內，新增字串
 - 可先用\$來取回原變數值，例如：
 - PATH=\$PATH:/tmp/bin #可成功新增
 - Name="\$Name"newword #可成功新增
 - Name=\${Name}newword #可成功新增
 - Name=\$Namenewword #新增失敗，因為變數不存在

讀取使用者輸入，當作變數值

- `read 變數名稱 #讀取來自鍵盤輸入的變數值`
 - `-p` : 向使用者印出提示字串
- 範例
 - `read -p "input your name:" name`
 - `echo $name`

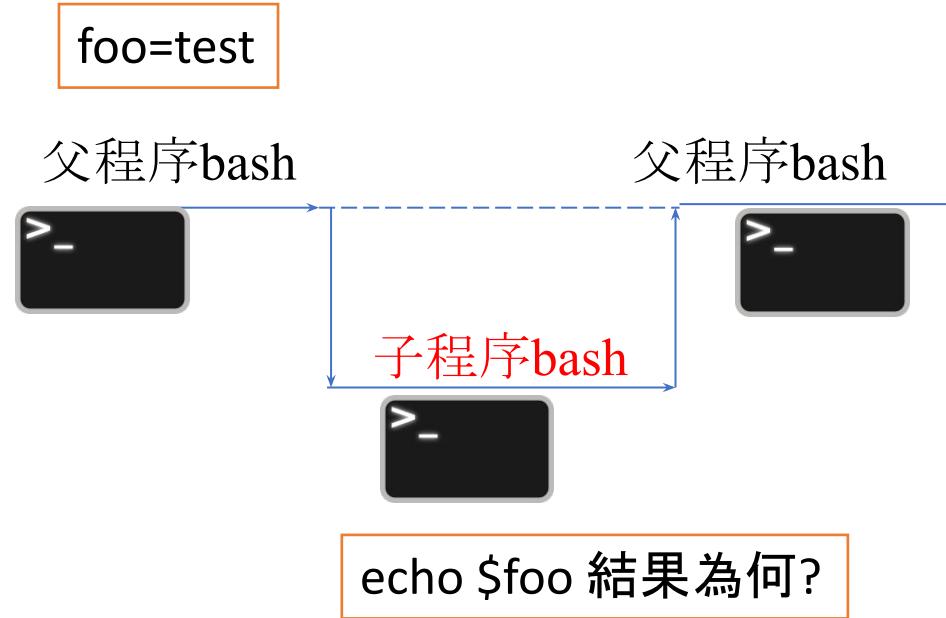
```
[i i i edu@localhost ~]$ read -p "input your name:" name
input your name:jojo
[i i i edu@localhost ~]$ echo $name
jojo
```

變數生效的範圍

變數類型	生效範圍	使用方式
指令列自訂變數	在當前Shell有效	var=value
環境變數	在當前Shell以及子shell皆有效	<ul style="list-style-type: none">• export var• export var=value

Shell 變數 – export/unset

- foo=test
- echo \$foo
 - test
- bash (進入子程序)
- echo \$foo
 - (沒東西了)
- exit (回到父bash)
- export foo (設成環境變數)
- bash (進入子程序)
- echo \$foo
 - test
- unset foo
- exit (回到父bash)
- echo \$foo (不會影響到父bash變數的使用)
- test



- 系統環境變數全大寫

變數	意義
HOME	使用者的家目錄的絕對路徑
PATH	執行檔搜尋路徑
LOGNAME	使用者的登入名稱
SHELL	目前使用哪個shell
LS_COLORS	設定Shell顏色
PWD	記錄使用者目前所在的目錄
UID	目前的使用者uid
HISTFILE	指令歷史紀錄的檔案位置

Module 35. 別名、系統設定檔

- 34-1：別名
- 34-2：系統重要設定檔
- 34-3：軟式連結(link -s)

- 於終端機執行「help」

```
~$ help
GNU bash, version 5.0.17(1)-release (x86_64-pc-linux-gnu)
These shell commands are defined internally. Type `help' to see this list.
Type `help name' to find out more about the function `name'.
Use `info bash' to find out more about the shell in general.
Use `man -k' or `info' to find out more about commands not in this list.
```

A star (*) next to a name means that the command is disabled.

```
job_spec [&]
(( expression ))
. filename [arguments]
:
[ arg... ]
[[ expression ]]
alias [-p] [name[=value] ... ]
bg [job_spec ...]
bind [-lpsvPSVX] [-m keymap] [-f filename] [-q name] [-u name] [-r keyseq] [-x keyse>
break [n]
builtin [shell-builtin [arg ...]]
caller [expr]
case WORD in [PATTERN [| PATTERN]...) COMMANDS ;;; esac
cd [-L|[-P [-e]] [-@]] [dir]
command [-pVv] command [arg ...]
compgen [-abcdefgjksuv] [-o option] [-A action] [-G globpat] [-W wordlist] [-F func>
complete [-abcdefgjksuv] [-pr] [-DEI] [-o option] [-A action] [-G globpat] [-W wordl>
compopt [-o]+o option] [-DEI] [name ...]
continue [n]
coproc [NAME] command [redirections]
```

指令別名 (alias / unalias)

alias 別名

- 設定別名

unalias 別名

- 移除別名

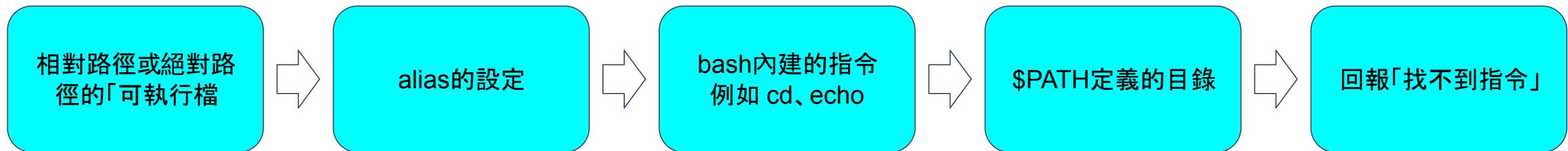
- 「別名」可用來用方便記憶的字串取代長指令

- alias cl="clear" #設定cl等同clear
- alias temp="cd /tmp; ls" #設定temp等同cd /tmp;ls
- unalias cl #刪除cl的別名
- alias #查看目前所有的別名

```
alias egrep='egrep --color=auto'  
alias fgrep='fgrep --color=auto'  
alias grep='grep --color=auto'  
alias l='ls -CF'  
alias la='ls -A'  
alias ll='ls -alF'  
alias ls='ls --color=auto'
```

alias運作原理

- 當使用者輸入指令時，系統會依序分析所輸入的內容是否為：



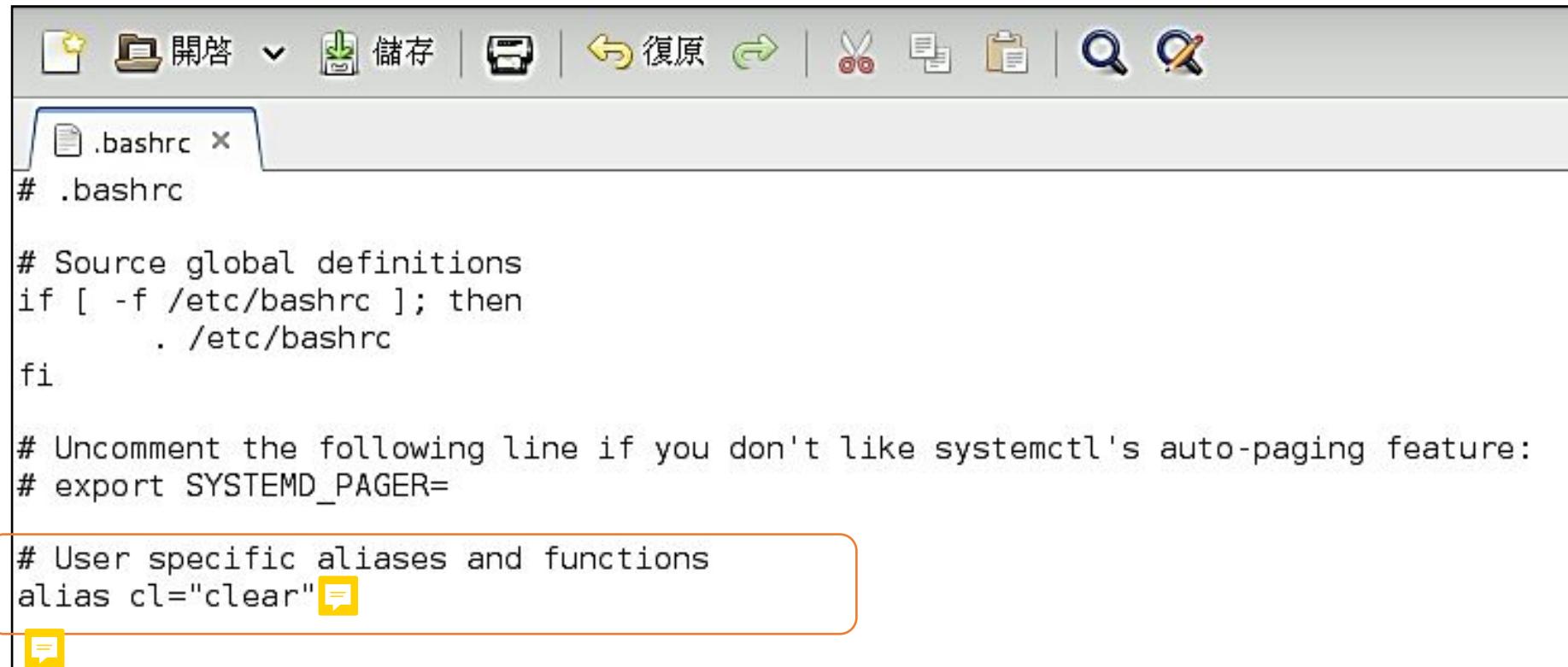
- 若要執行當前目錄下的執行檔，必須使用./
- 例如 ./testexe

- PATH是一個環境變數，用來記錄系統執行檔(指令)的所在目錄清單
- 各個目錄路徑以：區隔，可執行 echo \$PATH 檢視內容
- 例如 /usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin
- 依序在各目錄內搜尋符合名稱的檔案
- 一旦找到後就停止

```
[user1@localhost ~]$ echo $PATH
/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin:/home/user1/.local/
bin:/home/user1/bin
```

- 於命令列設定的變數或是別名，無法「長久維持」，一旦重開機(甚至是切換終端機)後就復原了，所以必須改使用「系統設定檔」來達成目的
- Linux內有很多種設定檔，在系統重啟或是帳號登入時會重新讀取其中內容，所以可以達到「長久維持」的效果
- 常用的設定檔如：
 -  `~/.bashrc` → 每個帳號家目錄內都有一份，可自行修改，只影響到自己
 - `/etc/profile` → 必須使用root權限才能修改，會影響整個系統及所有帳號

- 編輯`~/.bashrc`檔案 (影響範圍只限自己帳號)
 - gedit `~/.bashrc`
- 編輯完成後，要重新套用設定檔才會生效
 - `source ~/.bashrc`



```
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

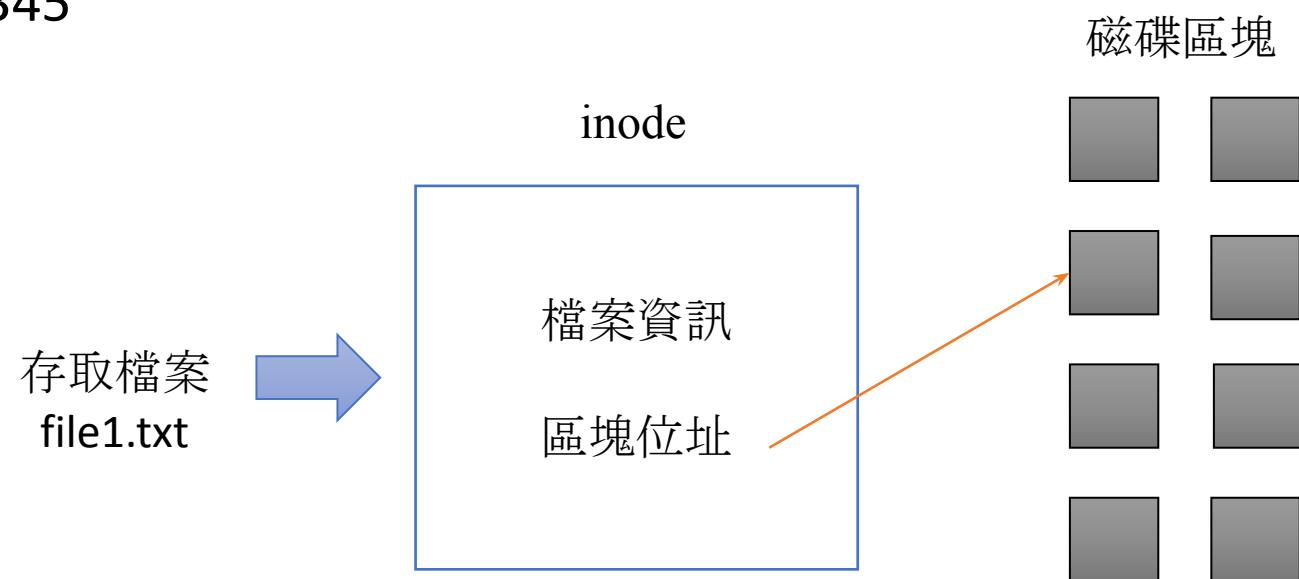
# User specific aliases and functions
alias cl="clear"
```

註：若要影響全系統，需以root身分編輯`/etc/profile`

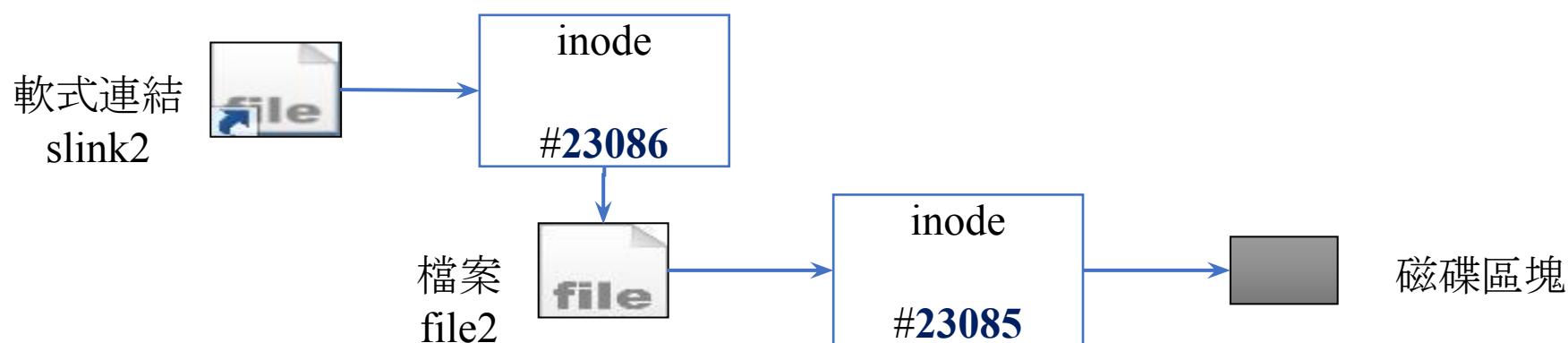
source 指令

- source指令可以用來套用系統設定檔或是執行腳本
- 例如當修改~/.bashrc檔案後，修改的內容不會自動生效，生效方法有二
 - 重新開機或者帳號重新登入
 - 手動指定套用設定檔：
 - source ~/.bashrc

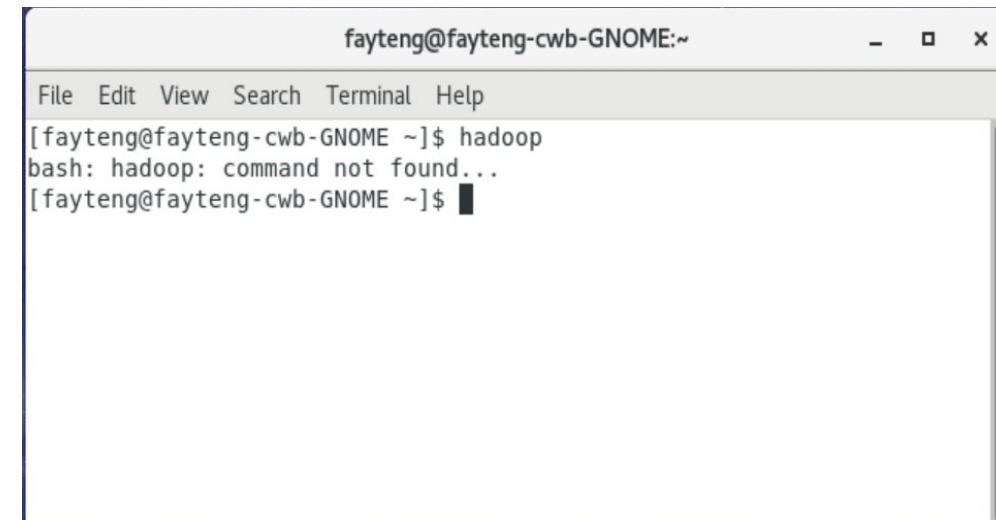
- 每個檔案皆有對應的inode用來描述：
 - 檔案資訊 (權限屬性、大小、更動時間等)
 - 檔案儲存的區塊位置 (檔案內容)
- 查詢 inode
 - ls -i file1.txt
 - 12345



- `ln -s 檔案(或目錄) 連結名稱`
 - -s 表示 soft link (或叫 symbolic link)
 - 若實際檔案(或目錄)被刪除，則符號連結就失效
 - `ln -s file2 slink2 #建立`
 - `rm file2 #刪除file2後，slink2就失效了`
 - `rm slink2 #刪除`



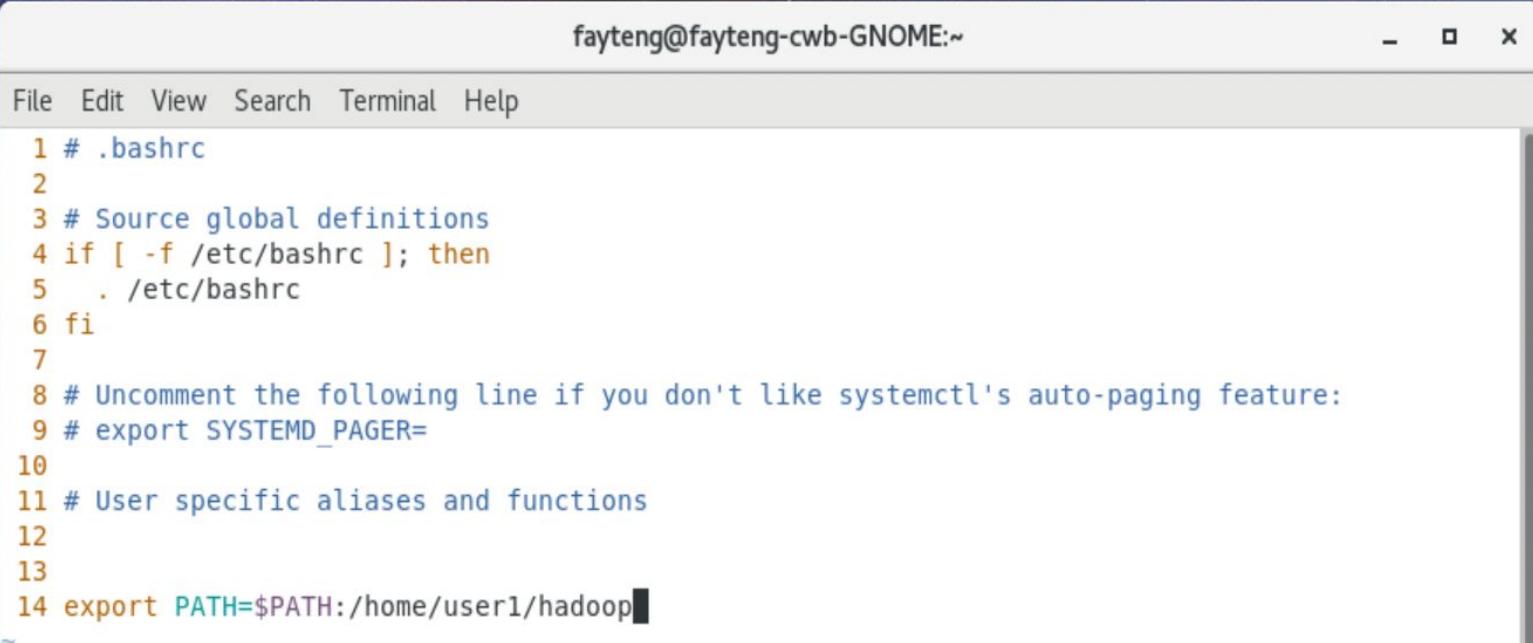
- 假設家目錄下有一個資料夾hadoop裡面有一個執行檔hadoop, 如何讓他變成可執行指令?
 - 首先先建立一個hadoop資料夾和執行檔
 - mkdir ~/hadoop
 - touch ~/hadoop/hadoop
 - 再來讓檔案可執行
 - chmod +x ~/hadoop/hadoop
 - 此時打hadoop會發現, 指令找不到



fayteng@fayteng-cwb-GNOME:~ - □ ×
File Edit View Search Terminal Help
[fayteng@fayteng-cwb-GNOME ~]\$ hadoop
bash: hadoop: command not found...
[fayteng@fayteng-cwb-GNOME ~]\$ █

方法一：(修改設定檔)

- vi ~/.bashrc
- 在下方加入一行 export PATH=\$PATH:/home/user1/hadoop



```
fayteng@fayteng-cwb-GNOME:~
```

```
File Edit View Search Terminal Help
```

```
1 # .bashrc
2
3 # Source global definitions
4 if [ -f /etc/bashrc ]; then
5   . /etc/bashrc
6 fi
7
8 # Uncomment the following line if you don't like systemctl's auto-paging feature:
9 # export SYSTEMD_PAGER=
10
11 # User specific aliases and functions
12
13
14 export PATH=$PATH:/home/user1/hadoop
```

- source ~/.bashrc

方法二：(建立軟連結)

- 在PATH裡面的其中一個目錄建立一個軟連結
 - `sudo ln -s /home/user1/hadoop/hadoop /usr/bin/hadoop`

Module 36. shell script

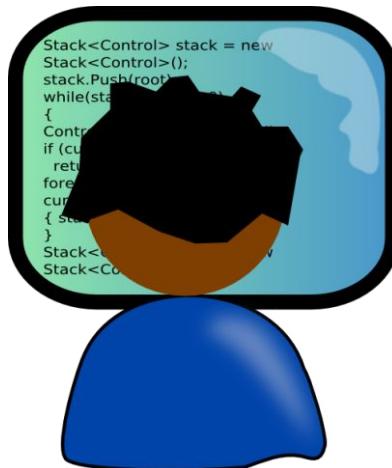
- 36-1: shebang定義
- 36-2: 權限設定
- 36-3: 執行方式

指令稿 (Shell Script)

- 可將多個指令寫在一個文字檔案內，作為執行腳本
 - 自動化，節省指令輸入的時間
- 常被命名為*.sh 以供(人)識別
- 執行原理：
 - 是一種直譯式語言
 - 由上到下，由左到右執行
 - 空白行會被忽略



- **#!/bin/bash** (#!稱為shebang)
 - 放置於Script的第一行
 - 宣告將用/bin/bash程式來解譯這個指令稿
- 其他以 # 開頭的行則表示為註解



程式設計師最討厭的兩件事：
1. 寫註解
2. 別人程式碼內沒有註解

指令稿的執行方式

- 使用bash指令
 - bash test.sh 

在子程序內的變數於執行完成後會結束
，而不會傳回到父程序



bash test.sh
或 **./ test.sh**



- 視為執行檔案
 - chmod a+x test.sh
 - ./test.sh

指令稿的執行方式 (續)

- 使用source指令

- source test.sh

在同一個shell內完成



父程序bash 

source test.sh
或 . test.sh

- 使用 . 符號

- . test.sh

指令稿執行練習

- gedit test.sh

```
#!/bin/bash
echo "Input your name:"
read username #輸入資料會儲存至變數username
echo $username #印出變數值
```

由於Windows (\r\n)跟Linux (\n)
的換行符號不一致，故建議：
在Linux平台上寫Linux腳本

- 以子bash執行
 - bash test.sh #設定的username變數不會影響原系統
- 改以source指令執行
 - source test.sh #設定的username變數會影響原系統

用於指令稿中的特殊變數

變數	意義
\$#	參數的個數
\$0	當前的腳本名稱 (或指令的名稱)
\$1,\$2,...,\$9	分別代表傳進來的第幾個參數



```
/path/to/scriptname opt1 opt2 opt3 opt4 ...
$0           $1   $2   $3   $4   ...
```

- 於指令稿 test.sh 加入以下指令：

```
echo $0
echo $1
echo $2
```

- bash test.sh a b

結果顯示：
test.sh
a
b

- 執行指令後，系統會回傳一個「執行結果狀態碼」
 - 若指令正常結束，狀態碼為「0」
 - 若指令錯誤或異常結束，狀態碼為「大於0的整數值」
- 以 \$? 查出狀態碼
 - 此特殊變數可以列出「上一個指令」執行結果的狀態碼
- 範例：

```
[iiiedu@localhost ~]$ echo hello
hello
[iiiedu@localhost ~]$ echo $?
0
[iiiedu@localhost ~]$ echooooo
bash: echooooo: 找不到指令...
[iiiedu@localhost ~]$ echo $? 反馈
127
```

條件判斷式

- 簡單條件判斷式

```
if [ 條件判斷式 ]; then  
    當條件判斷式成立時, 進行的工作內容;  
fi
```

- 多重條件判斷式

```
if [ 條件判斷式 ]; then  
    當條件判斷式成立時, 進行的工作內容;  
else  
    當條件判斷式不成立時, 進行的工作內容;  
fi
```

在中括號的兩端需要
有空白字元來分隔

- 更複雜的:

```
if [ 條件判斷式一 ]; then  
    當條件判斷式一成立時, 進行的工作內容;  
elif [ 條件判斷式二 ]; then  
    當條件判斷式二成立時, 進行的工作內容;  
else  
    當一與二都不成立時, 進行的工作內容;  
fi
```

檔案類型及權限的判斷

- 可利用test測試，或是放到if判斷內

- test -f /home/user1/file1
- test -d /home/user1/dir1
- test -x /home/user1/file1
- if [-r /home/user1/file1]; then

符號	說明
-f	檢查檔案是否存在
-d	檢查目錄是否存在
-e	檢查檔案或目錄是否存在
-r	檢查該檔案、目錄是否可讀
-w	檢查該檔案、目錄是否可寫
-x	檢查該檔案、目錄是否可執行
-L	檢查連結檔是否存在

```
[iiiedu@localhost ~]$ test -d /tmp
[iiiedu@localhost ~]$ echo $?
0
[iiiedu@localhost ~]$ test -f /tmp
[iiiedu@localhost ~]$ echo $?
1
```

宣告變數的類型(declare)

- 變數預設為字串型態
- 使用declare可宣告變數類型
- -i : 變數為整數(integer)
 - declare -i num1
- -a : 變數為陣列(array)
- -x : 等同export, 將變數設定為環境變數；
- -r : 將變數設定成為 唯讀(read only), 不可再被更改內容, 也不能unset

```
[~]$ sum=10+20+30
[~]$ echo $sum
10+20+30
```

```
[~]$ declare -i sum2
[~]$ sum2=10+20+30
[~]$ echo $sum2
60
```

- 整數變數可以進行數值運算
 - 但預設最多僅能到達整數形態，意即沒有小數點
 - 也就是說 $1/3$ 的計算結果為 0
- 整數變數可比較數值大小
 - 例如比較n1及n2的大小
 - if [n1 -eq n2]; then

符號	說明
-eq	$n1=n2$ (equal)
-ne	$n1 \neq n2$ (not equal)
-gt	$n1 > n2$ (greater than)
-lt	$n1 < n2$ (less than)
-ge	$n1 \geq n2$ (greater than or equal)
-le	$n1 \leq n2$ (less than or equal)

- bash提供一維陣列，通常會與迴圈或者其他判斷式交互使用
- 陣列的設定方式為
 - var[index]=content
- 範例
 - declare -a car
 - car[1] = "ford"
 - car[2] = "benz"
 - car[3] = "bmw"
 - echo \${car[1]}

```
[~]$ declare -a car
[~]$ car[1] = "ford"
[~]$ car[2] = "benz"
[~]$ car[3] = "bmw"
[~]$ echo ${car[1]}
ford
[~]$ echo ${car[2]}
benz
[~]$ echo ${car[3]}
bmw
```

命令執行的判斷依據 (; 及 && 及 ||)

- 指令1 ; 指令2
 - 分號用來區隔前後兩個指令，依序執行
 - 不論指令1是否執行成功，都會繼續執行指令2
 - 例如 cd /tmp ; rm *.txt
- 指令1 && 指令2
 - && (表示and)
 - 若指令1正確執行(回傳值為0)，才會執行指令2
 - 若指令1執行錯誤(回傳錯誤代碼)就終止
- 指令1 || 指令2
 - || (表示or)
 - 若指令1正確執行(回傳值為0)，則指令2不執行
 - 若指令1執行錯誤(回傳錯誤代碼)，則執行指令2

範例：以 ls 測試 /tmp/myfile 是否存在？存在則顯示 "exist"，否則顯示 "not exist"

```
ls /tmp/myfile && echo "exist" || echo "not exist"
```

測試字串

選項	說明
-n 字串	測試字串長度是否大於0
-z 字串	是否為空字串
字串1 == 字串2	左右兩邊字串(變數值)相等
字串1 != 字串2	左右兩邊字串(變數值)不相等

== 與 != 符號的左右
兩邊要有空白字元

```
[i i i edu@localhost ~]$ name=abc
[i i i edu@localhost ~]$ test -n $name
[i i i edu@localhost ~]$ echo $?
0
[i i i edu@localhost ~]$ test -z $name
[i i i edu@localhost ~]$ echo $?
1
[i i i edu@localhost ~]$ test $name == "abc"
[i i i edu@localhost ~]$ echo $?
0
[i i i edu@localhost ~]$ test $name == "123"
[i i i edu@localhost ~]$ echo $?
1
```

- 寫一個指令稿 mytest.sh
- 執行 bash mytest.sh “hello”

```
#!/bin/bash
if [ "$1" == "hello" ]; then
    echo "world"
elif [ "$1" == "" ]; then
    echo "Please input Parameter"
else
    echo "wrong parameter"
fi
```

- 在中括號的兩端需要有空白字元來分隔
- 為避免空白字元造成錯誤，變數建議用雙引號標起來，
- 常數最好也用雙引號標起來
- \$1 表示傳進來的第一個參數

- 測試帳號是否存在
 - 寫一個指令稿 : mytest2.sh
 - 執行 bash mytest2.sh

```
#!/bin/bash
read -p "Input account to test :" account #讀取使用者輸入
isExist=$(id "$account") #將id執行結果存在isExist

#若帳號存在, 表示id執行結果會被記錄下來
#若帳號不存在, 表示stdout會無資料(只有stderr)
if [ "$isExist" == "" ]; then
    echo "not exist"
else
    echo "account exist"
fi
```

case流程控制

• 語法

```
case string in
    樣式1 ) 指令敘述1
    ;;
    樣式2 ) 指令敘述2
    ;;
    以此類推
    * ) 指令敘述 4;;
esac
```

兩個分號 (;;) 表示該指令段落結束

• 範例

```
#!/bin/bash
language='Java'
case $language in
    Java*) echo "It is Java"
    ;;
    Python*) echo "It is Python"
    ;;
    C*) echo " It is C"
    ;;
    *) echo " None"
esac
```

前面所有樣式皆不符合時，
才會顯示None

迴圈 for...do...done

• 腳本範例：

for迴圈通常用於循序處理資料，
讓包在段落內的指令可以重複執行

迴圈
段落

```
for var in Tom Mary John  
do  
    echo $var  
done
```

依序印出Tom、Mary、John

```
filelist=$(ls /tmp)  
for name in $filelist  
do  
    echo "File name is ${name}"  
done
```

依序印出每個檔名

```
for ((i=0;i<10;i++)) #分別寫出起始式、判斷式、遞增式  
do  
    echo "$i" #印出i的值  
done
```

依序印出 0 ~ 9

while...do...done

- 當判斷式成立時進行迴圈，直到條件不成立才停止

- 語法：

```
while [ 判斷式 ]
do
    程式段落
done
```

- 範例

- 印出Hi直到Ctrl C結束

```
while true
do
    echo "Hi"
done
```

- 印出 0 ~ 9

- 用 **\$((計算式))** 進行數值運算
 - 支援+, -, *, /, % 的整數運算

```
i=0
while [ $i != 10 ]
do
    echo $i      #印出i的值
    i=$((i+1))  #讓 i 每次都增加 1
done
```

while...do...done

- 範例

- 讀檔

```
while read input #將讀取的資料存到input變數
do
    echo $input
done < /etc/passwd #這裡使用了資料流導入符號
```

- 自我練習:

- 修改以上腳本，可顯示第一個參數所指定的檔案
 - 例如: bash readfile.sh /etc/passwd

until...do...done

- 判斷式條件成立時就終止迴圈，否則持續進行迴圈
- 語法：

```
until [ 判斷式 ]
do
    程式段落
done
```

- 腳本範例：印出 0 ~ 9

```
i=0
until [ $i -ge 10 ]
do
    echo $i      #印出i的值
    i=$((i+1))   #讓 i 每次都增加 1
done
```

迴圈範例：發送垃圾信

- 灌爆某個信箱

```
for((i=1;i<10;i++))  
do  
    echo "hello" | mail -s "test" xxx@gmail.com  
done
```



- 同時寄給名單上的多人

```
maillist=$(cat /home/iiiedu/maillist.txt)  
for email in $maillist  
do  
    echo Sending to $email  
    echo "hello" | mail -s "Wow!" $email  
done
```

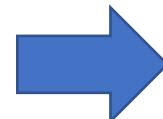
maillist.txt

```
user1@gmail.com  
user2@gmail.com  
user3@gmail.com  
user4@gmail.com  
....
```

- 函式基本架構如下：
 - 函式名稱(function 關鍵字為選擇性)
 - 傳入參數
 - 函式內操作
 - 回傳值

```
function function_name () {  
    # Do some thing....  
    [回傳值]  
}
```

```
#!/bin/bash  
echo "call function example:"  
  
function myFun() {  
    echo "${0} hello ${1}"  
}  
  
myFun "world"
```



```
[root@localhost ~]# bash test3.sh  
call function example:  
test3.sh hello world
```

有個工程師某日在下班前接到老婆的電話...
老婆說：「**下班順便帶10個包子回家，如果看到賣西瓜的，就買1個。**」
晚上工程師回到家，老婆看了就問：「怎麼只買了1個包子？」
工程師：「**因為看到賣西瓜的。**」

- 請撰寫一個shell script，符合以上情境。
- 執行範例如下：

```
[user1@localhost ~]$ bash buy.sh
Do you see waltermelon seller (y / n)? n
I buy 10 buns
[user1@localhost ~]$ bash buy.sh
Do you see waltermelon seller (y / n)? y
I buy 1 bun
```