2020/5/19

虛擬機研究

系統程式-期中報告

翁瑋泓

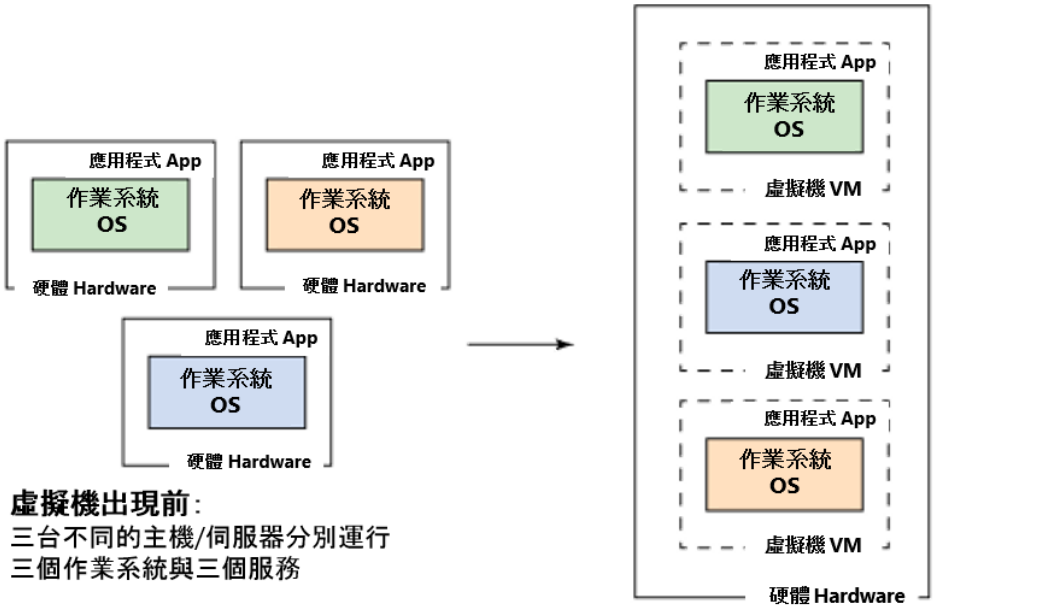
目錄

虛擬機 (Virtual Machine) 的基本原理

根據「切割硬體資源進行彈性分配」與「安全的系統測試環境」等需求的背景下，，IBM 開發出了「虛擬機」(Virtual Machine) 作為解方。

運用軟體技術、如 CPU 排程與虛擬記憶體，作業系統就能創造出一個多處理程式的假象：每個程式都具備自己的記憶體 (雖然是虛擬的)，且在屬於自己的處理器上面運作。

也就是藉由軟體虛擬技術，提供一個與底層硬體功能一模一樣的介面，這樣系統就能為每個程式都提供了一份底層硬體的 Copy，稱為虛擬機 (Virtual Machine)。簡單來講，就是用軟體技術模擬出一個硬體的環境。



虛擬機出現前:
三台不同的主機/伺服器分別運行
三個作業系統與三個服務

虛擬機出現後:
一台主機/伺服器能開三台虛擬機，在上面同時
運行三個作業系統與服務

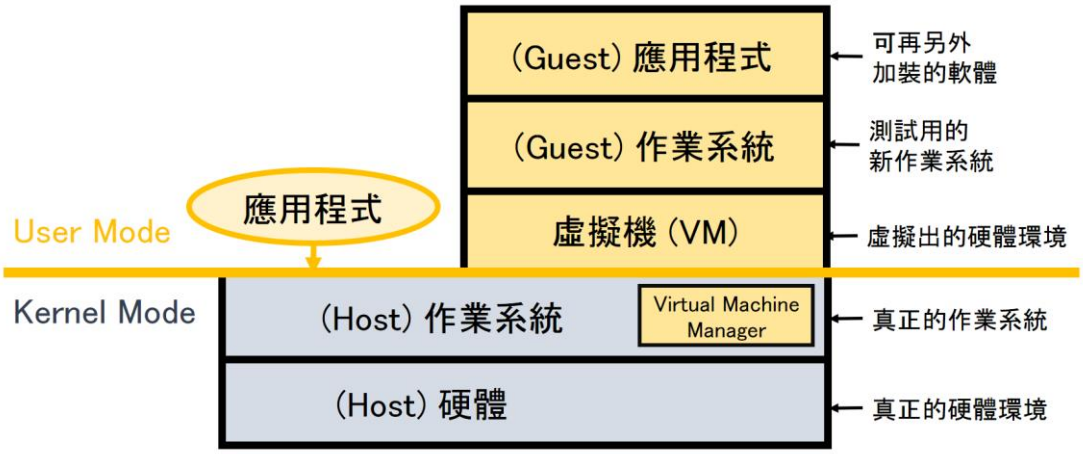
若用更好懂一點的比喻… 現在你有一台實體電腦，可以視為鳴人的本體 (硬體, Host Hardware)。

透過影分身之術 (虛擬化, Virtualization) 變出來分身 (虛擬機, Virtual Machine)，共享鳴人這個單一的實體資源、查克拉都是源於鳴人。

但把其中一個影分身打倒，並不會傷到鳴人本身。

虛擬機的基本名詞與類型

虛擬機的基本名詞



Host (鳴人的查克拉)

實際的作業系統和硬體環境。(Underlying hardware system or Operating System)

> Virtual Machine Manager (鳴人的影分身忍術)

簡稱 VMM、又叫 Hypervisor，用以建立，並管理、執行虛擬機的模組。

> VIRTUAL MACHINE (鳴人的影分身)

透過 VMM 創造出來、模擬底層硬體 (鳴人本人) 的軟體。

> Guest (鳴人的分身用來打架或做事)

運行在虛擬機上的作業系統或軟體程式。(Process provided with virtual copy of the host)

QEMU 介紹

QEMU（quick emulator）是一款由 Fabrice Bellard 等人編寫的免費的可執行硬體虛擬化的（hardware virtualization）開源代管虛擬機器（VMM）。

QEMU 是一個代管的虛擬機器鏡像，它通過動態的二進位轉換，類比 CPU，並且提供一組裝置模型，使它能夠執行多種未修改的客戶機 OS，可以通過與 KVM（kernel-based virtual machine 開源加速器）一起使用進而接近本地速度執行虛擬機器（接近真實電腦的速度）。

特性

QEMU 可以在執行所有程式的情況下儲存和恢復虛擬機器的狀態。

客戶作業系統（Guest Operating System）不需要修補就可以在 QEMU 中執行。

系統模組

QEMU 有多種模式[1]

User mod：又稱作「使用者模式」，在這種模組下，QEMU 執行針對不同指令編譯的單個 Linux 或 Darwin/macOS 程式。系統呼叫與 32/64 位元介面適應。在這種模式下，我們可以實現交叉編譯（cross-compilation）與交叉偵錯（cross- debugging）。

System mod：「系統模式」，在這種模式下，QEMU 類比一個完整的電腦系統，包括外圍裝置。它可以用於在一台電腦上提供多台虛擬電腦的虛擬主機。 QEMU 可以實現許多客戶機 OS 的啟動，比如 x86，MIPS，32-bit ARMv7，PowerPC 等等。

KVM Hosting：QEMU 在這時處理 KVM 鏡像的設定與遷移，並參加硬體的仿真，但是客戶端的執行則由 KVM 完成。

Xen Hosting：在這種代管下，客戶端的執行幾乎完全在 Xen 中完成，並且對 QEMU 封鎖。QEMU 只提供硬體仿真的支援。

加速器

KQEMU 是一個 Linux 核心模組，由 Fabrice Bellard 撰寫，它明顯加快了在具有相同 CPU 架構的平台上類比 x86 或 x86-64 程式的速度。這可以通過直接在主機 CPU 上執行用戶模式代碼（以及可選的某些核心代碼）以及僅對核心模式和真實模式代碼使用處理器與外設類比來實現。即使宿主機 CPU 不支援硬體輔助虛擬化，KQEMU 也可以從多個客戶作業系統執行代碼。QEMU 支援大容量記憶體，這使得它們與 KQEMU 不相容。較新的 QEMU 版本已完全取消對 KQEMU 的支援。

由於缺乏對 KQEMU 和 QVM86 的支援，基於核心的虛擬機器（KVM）已經基本成為基於 Linux 的硬體輔助虛擬化解決方案，與 QEMU 一起使用。

英特爾的硬體加速執行管理器（HAXM）是 KVM 在 Windows 和 MacOS 上基於 x86 的硬體輔助虛擬化的開源替代品。2013 年，英特爾使用 QEMU 來進行 Android 開發。

硬體輔助仿真

MIPS 相容的龍芯 3 處理器增加了 200 條新指令來幫助 QEMU 翻譯 x86 指令，這些新指令降低了在 MIPS 流水線中執行 x86 / CISC 風格指令的開銷。由於中國科學院對 QEMU 進行了進一步改進，龍芯 3 在 9 個基準測試中，執行 x86 二進位檔案的同時，執行本機二進位檔案的平均效能達到 70%。

並列仿真

使用 QEMU 的虛擬化解決方案能夠並列執行多個虛擬 CPU。 對於用戶模式仿真，QEMU 將仿真執行緒對映到宿主執行緒。 對於全系統仿真，QEMU 能夠為每個虛擬 CPU 執行一個主機執行緒。 前提是客戶端已經更新到可以支援並列系統仿真，目前可以支援的 CPU 是 ARM 和 Alpha。否則 QEMU 將使用單個執行緒以迴圈方式類比執行每個虛擬 CPU。

VirtualBox

VirtualBox 是一套免費開放原始碼的虛擬電腦軟體，讓你在原有的系統架構下再建立出一台或是多台的新電腦，且可以在虛擬電腦裡安裝不同的作業系統，或是進行軟體測試。最重要的是在虛擬電腦裡所進行的動作皆不會影響或干擾到原有的電腦，例如磁碟分割、格式化甚至是中毒等等。VirtualBox 可以在 Windows, Linux, Mac, Solaris 等多種環境下運作，而且提供包含中文在內的多國語系，使用者輕輕鬆鬆就能建構出虛擬電腦，而無須繁雜的技術手冊。

我的習慣是在 VirtualBox 上額外建立一台 Windows XP 與 Windows 7 的虛擬電腦，大部分多用來進行軟體測試或是寫作使用，而主電腦裡只會安裝我平常最常使用到的軟體，這樣一來就算安裝到不安全的程式也不會影響到我日常的使用，只需要將虛擬電腦還原或者是刪除後重灌即可解決問題。若你想練習或玩玩 Linux、FreeBSD 的話，其實也無須捨棄原有的作業系統，直接安裝在 VirtualBox 裡使用就可以了。

主要特點

-支持 64 位客戶端操作系統，即使主機使用 32 位 CPU

-支持 SATA 硬盤 NCQ 技術

-虛擬硬盤快照

-無縫視窗模式（須安裝客戶端驅動）

-能夠在主機端與客戶端共享剪貼簿（須安裝客戶端驅動）

-在主機端與客戶端間建立分享文件夾（須安裝客戶端驅動）

-內建遠端桌面服務器，實現單機多用戶 - 支持 VMware VMDK 磁盤檔及 Virtual PC VHD 磁盤檔格式

-3D 虛擬化技術支持 OpenGL（2.1 版後支持）、Direct3D（3.0 版後支持）、WDDM（4.1 版後支持）

-最多虛擬 32 顆 CPU（3.0 版後支持）

-支持 VT-x 與 AMD-V 硬件虛擬化技術

-iSCSI 支持

-USB 與 USB2.0 支持

安裝 VirtualBox

在 VirtualBox 網站下載主機操作系統對應的二進製文件。 VirtualBox 可以安裝在 32 位和 64 位操作系統上。在 32 位主機操作系統上運行 64 位的虛擬機是可以的，但必須在主機的 BIOS 中啟用硬件虛擬化特性。

運行二進制安裝文件將開啟一個簡單的安裝嚮導，允許用戶定制 VirtualBox 特性，選擇任意快捷方式並指定安裝目錄。 USB 設備驅動以及 VirtualBox host-only 網絡適配器將一起安裝。

創建虛擬機

在 VirtualBox 中創建虛擬機相當簡單，很多設置可以按照用戶個人的喜好進行配置。一旦安裝了客戶操作系統並選擇了資源和網卡設置，就可以嘗試在小環境或開發環境中使用 VirtualBox 了。

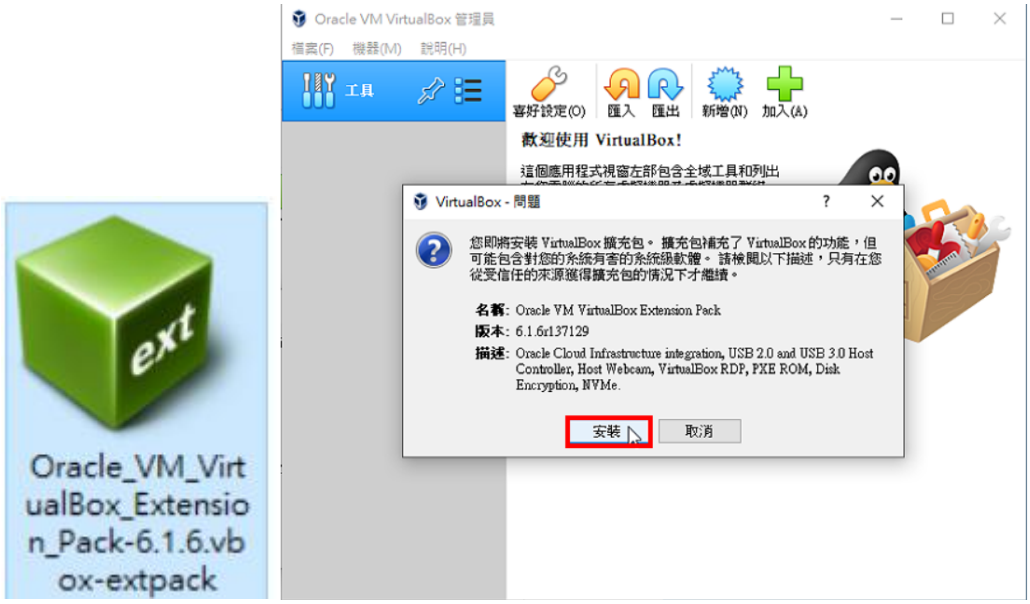
總體來說，在 VirtualBox 中創建虛擬機分三步：

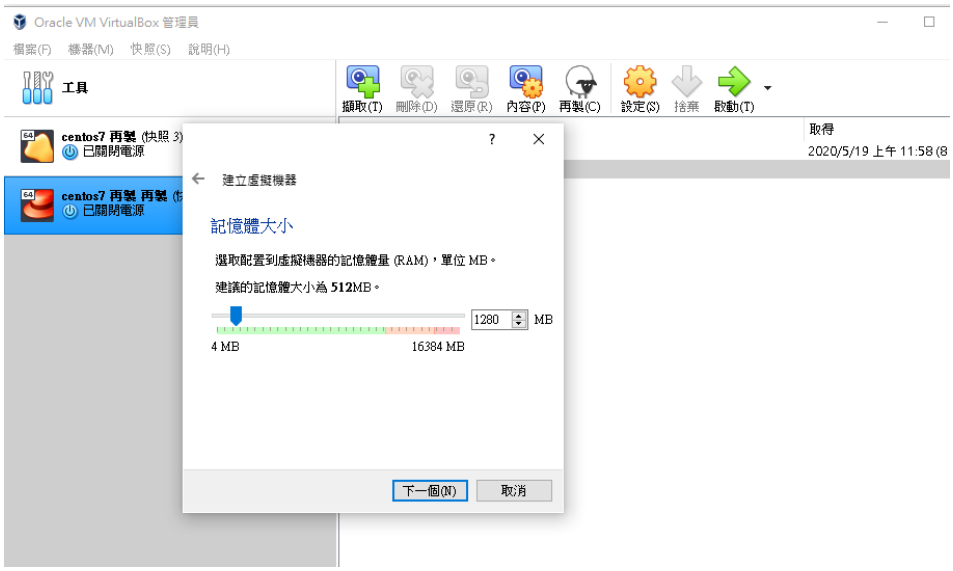
1. 為創建 VirtualBox 虛擬機做準備

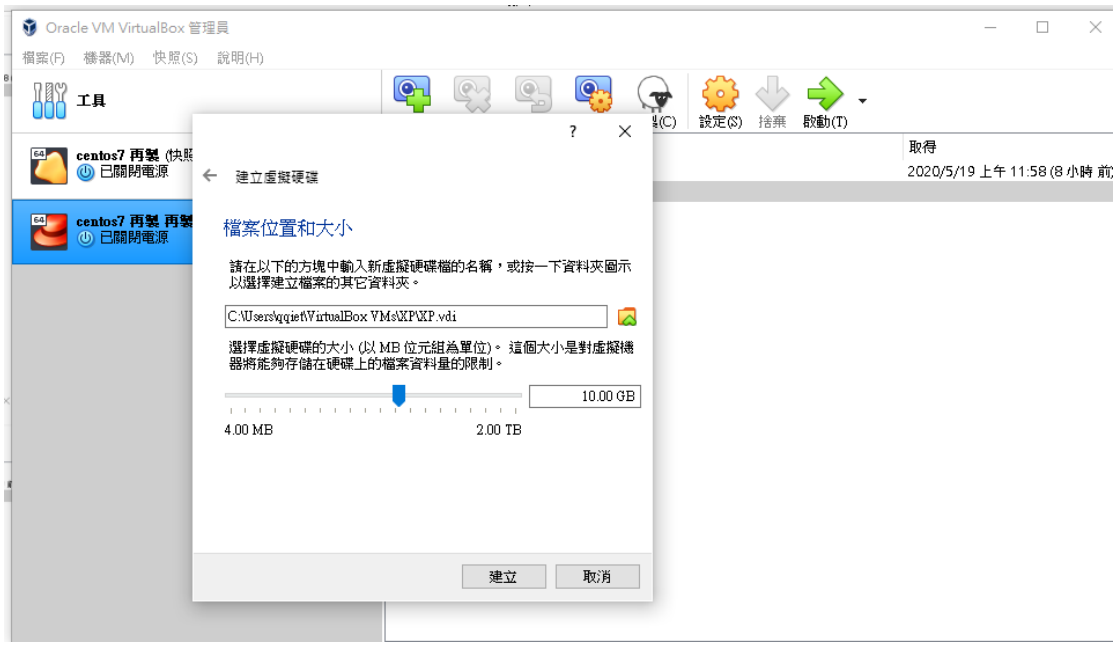
首先，命名虛擬機並選擇將要運行的客戶操作系統類型。此時選擇的客戶 OS 會影響之後嚮導中出現的默認設置。

接下來，配置計劃分配給每個虛擬機的內存大小。 VirtualBox 不支持內存過量使用，所以不能給一個虛擬機分配超過主機內存大小的內存值。

最後一個步驟是創建虛擬磁盤並指定虛擬機磁盤文件的類型和大。在 Oracle VM VirtualBox 中，你可以選擇動態擴展的磁盤或者固定大小的磁盤。動態磁盤起始值較小，隨著客戶操作系統寫入數據到磁盤而逐漸增加。對於固定磁盤類型來說，所有的磁盤空間在虛擬機創建階段一次性分配。之後也可以給虛擬機增加磁盤，或者使用 VBoxManage 命令行工具增加磁盤大小。


Oracle VM VirtualBox 管理員
檔案(F) 機器(M) 說明(H)
工具
喜好設定(O) 匯入 匯出 新增(N) 加入(A)
歡迎使用 VirtualBox!
這個應用程式視窗左部包含全域工具和列出
Oracle_VM_VirtualBox_Extension_Pack-6.1.6.vbox-extpack
VirtualBox - 問題
您即將安裝 VirtualBox 擴充包。擴充包補充了 VirtualBox 的功能，但可能包含對您的系統有害的系統級軟體。請檢閱以下描述，只有在您從受信任的來源獲得擴充包的情況下才繼續。
名稱: Oracle VM VirtualBox Extension Pack
版本: 6.1.6r137129
描述: Oracle Cloud Infrastructure integration, USB 2.0 and USB 3.0 Host Controller, Host Webcam, VirtualBox RDP, PXE ROM, Disk Encryption, NVMe.
安裝
取消


Oracle VM VirtualBox 管理員
檔案(F) 機器(M) 快照(S) 說明(H)
工具
摄取(T) 刪除(D) 還原(R) 內容(P) 再製(C) 設定(S) 捨棄 啟動(T)
centos7 再製 (快照 3)
已關閉電源
centos7 再製 再製
已關閉電源
取得
2020/5/19 上午 11:58 (8
建立虛擬機器
記憶體大小
選取配置到虛擬機器的記憶體量 (RAM)，單位 MB。
建議的記憶體大小為 512MB。
1280 MB
4 MB
16384 MB
下一個(N)
取消


Oracle VM VirtualBox 管理員
檔案(F) 機器(M) 快照(S) 說明(H)
工具
設定(S) 捨棄 啟動(T)
centos7 再製 (快照
已關閉電源
centos7 再製 再製
已關閉電源
取得
2020/5/19 上午 11:58 (8 小時 前)
建立虛擬硬碟
檔案位置和大小
請在以下的方塊中輸入新虛擬硬碟檔的名稱，或按一下資料夾圖示以選擇建立檔案的其它資料夾。
C:\Users\qqiet\VirtualBox VMs\XP\XP.vdi
選擇虛擬硬碟的大小 (以 MB 位元組為單位)。這個大小是對虛擬機器將能夠存儲在硬碟上的檔案資料量的限制。
10.00 GB
4.00 MB
2.00 TB
建立
取消

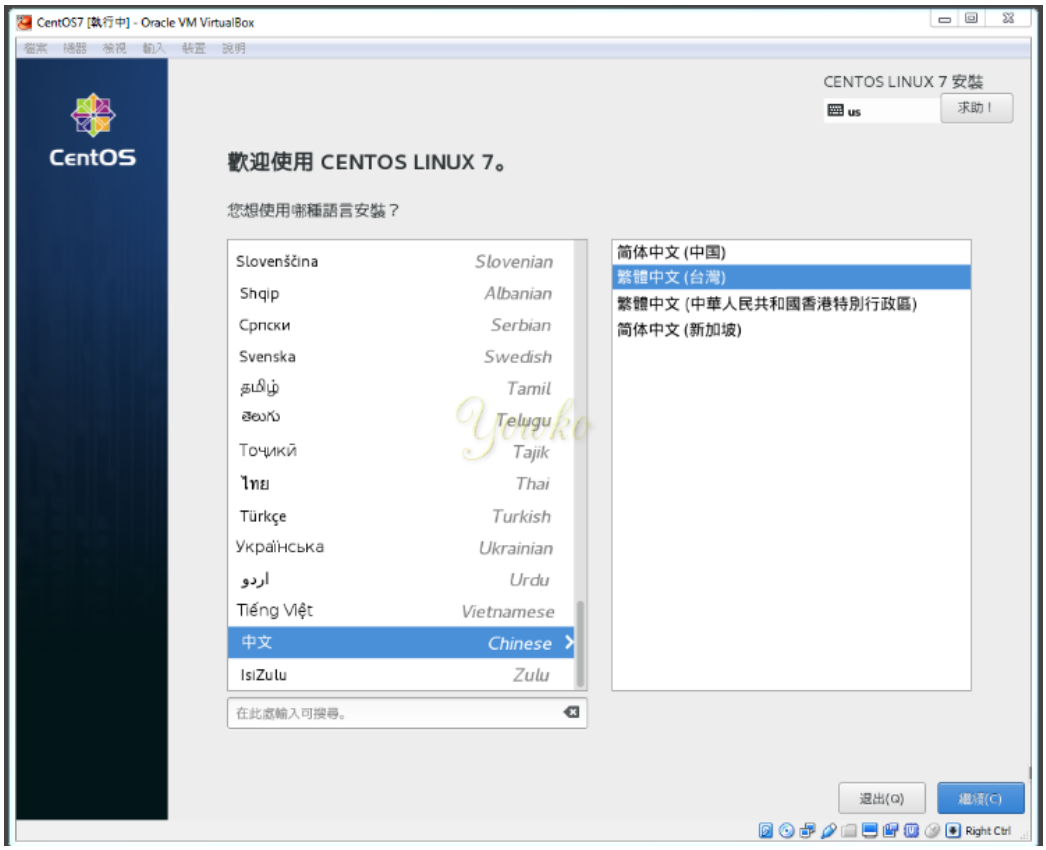
2. 安裝 VirtualBox 客戶操作系統

一旦完成了 VirtualBox 虛擬機創建嚮導，就可以開始安裝客戶操作系統了。為了掛載客戶操作系統光盤，選擇虛擬機，單擊設置，開始編輯虛擬機硬件配置。 （1）選擇左邊面板中“存儲”選項. （2）選擇“存儲”選項下的 CD/DVD 圖形。 （3）選擇屬性視圖下帶箭頭的 CD/DVD 圖形配置虛擬的 CD/DVD 驅動器。

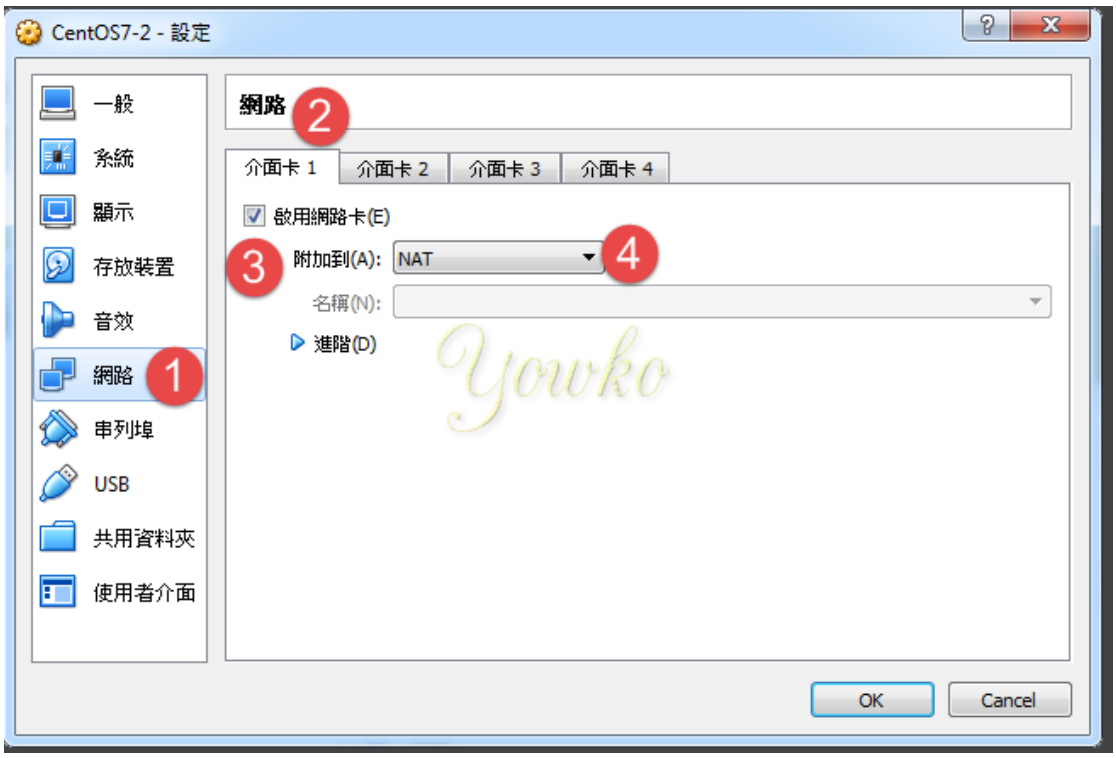
客戶操作系統安裝完成後，應該安裝增強功能包，增強功能包包括一些便於集成主機和虛擬機的驅動程序。為安裝增強功能包，需要打開虛擬機控制台窗口，在頂部菜單中選擇設備，然後選擇安裝增強功能包，啟動一個簡單的安裝嚮導。增強功能包安裝完成後，會重啟虛擬機。

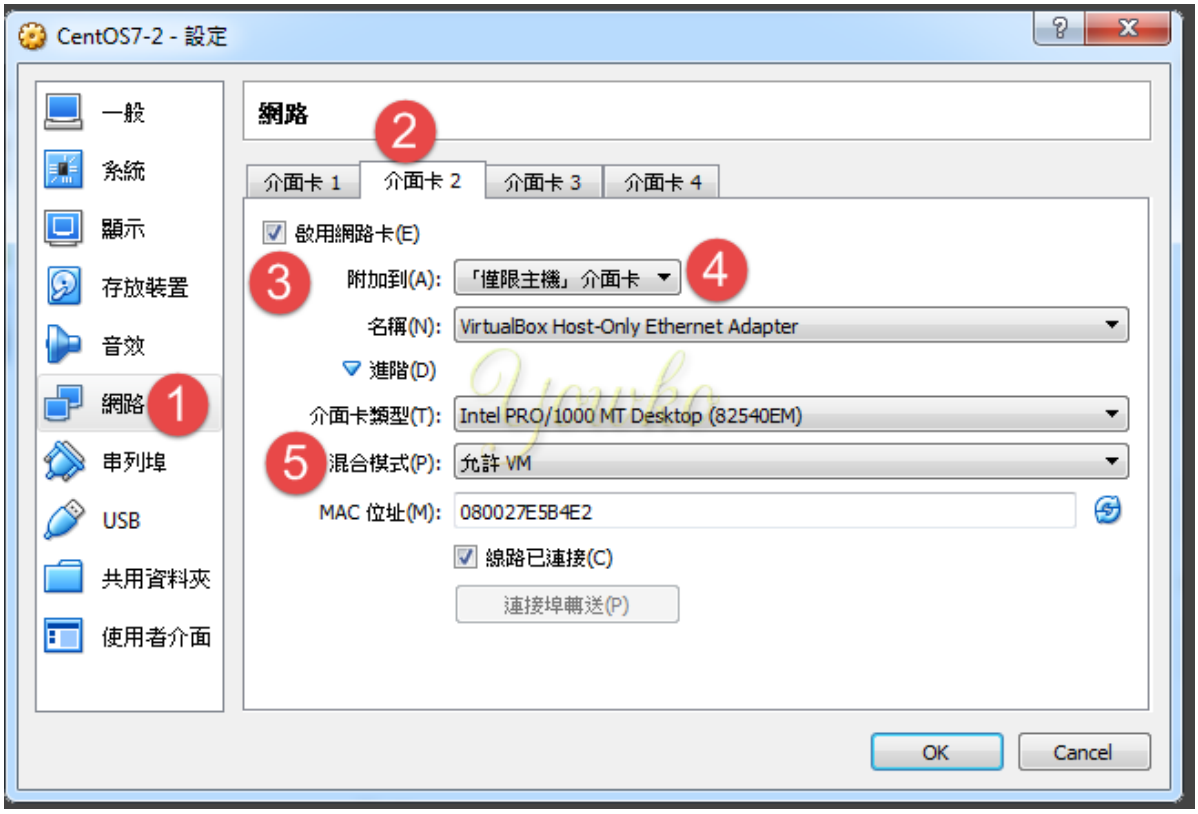
3. 在 VirtualBox 中設置虛擬機

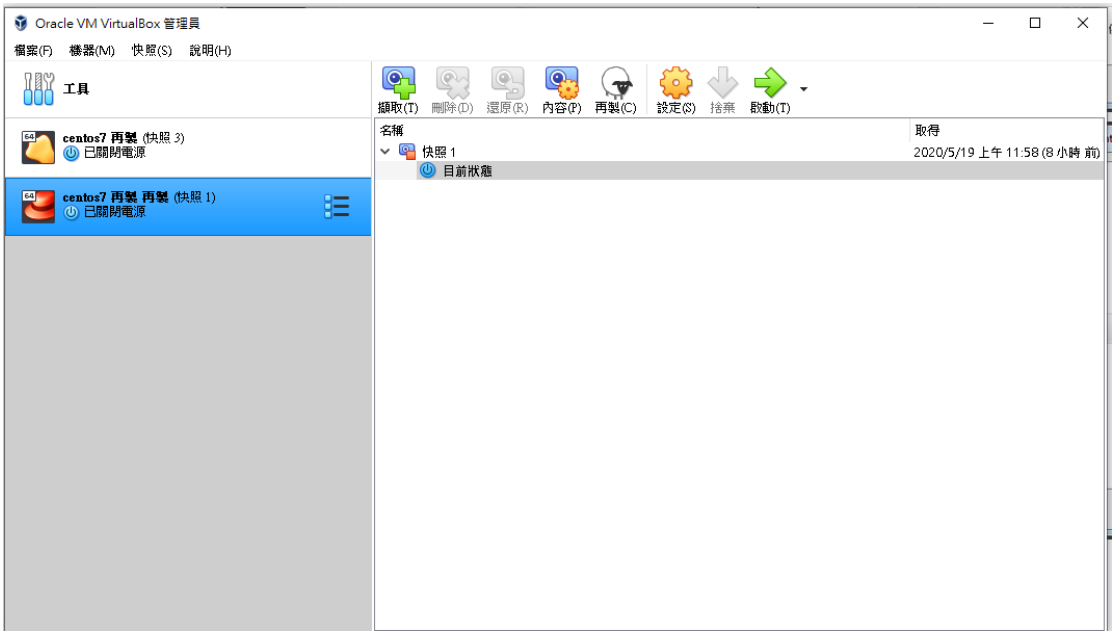
虛擬機關閉時，可以編輯虛擬機設置並更改硬件。VirtualBox 具有高級硬件設置特性，啟用了 IO APIC、PAE/NX 和嵌套分頁。用戶還可以修改虛擬 CPU 的數量—最多支持 32 個—不用管主機具有的物理 CPU 的核數。最後配置虛擬網卡。VirtualBox 允許在一個虛擬機上配置至多四塊虛擬網卡。默認的是 AMD PCnet-FAST III，大多數操作系統都支持 AMD PCnet-FAST III。也可以選擇 AMD PCnet-PCI II 和 Intel PRO-1000 系列的網卡，以及支持直接存取主機網卡的準虛擬化網卡。

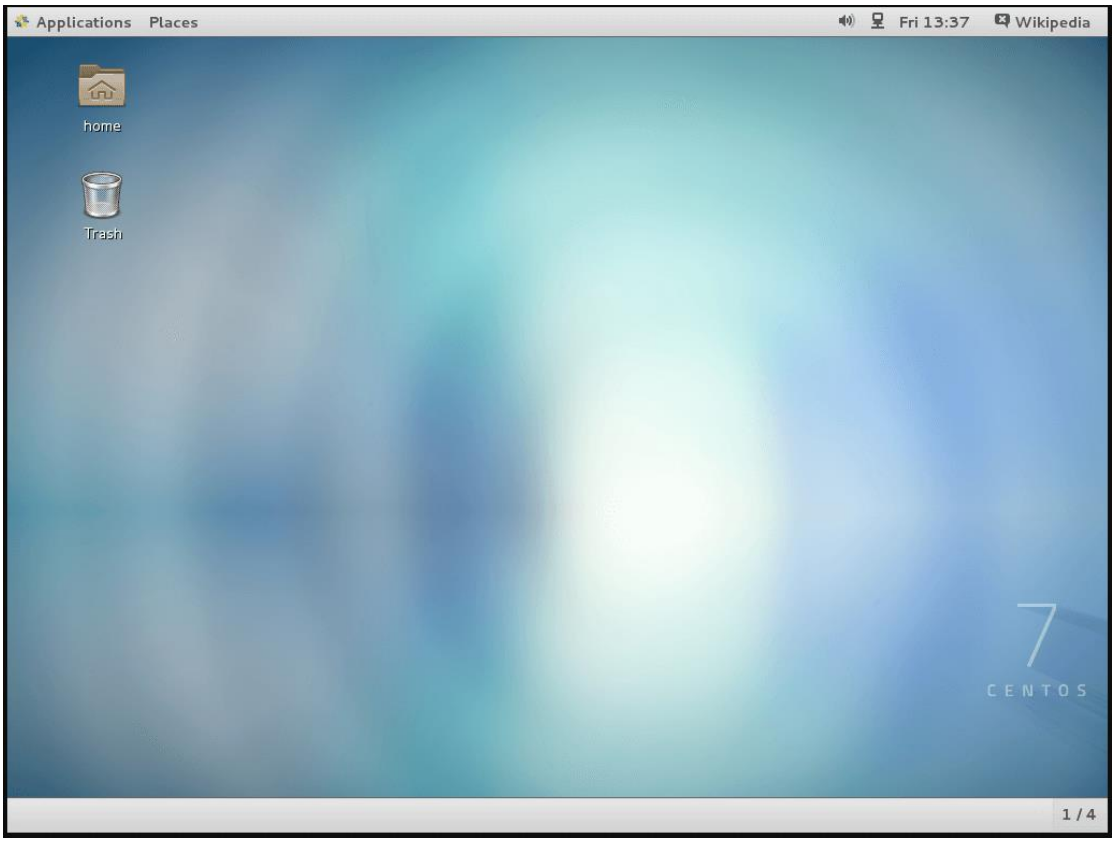

CentOS7 [執行中] - Oracle VM VirtualBox
CentOS
CENTOS LINUX 7 安裝
us
求助！
歡迎使用 CENTOS LINUX 7。
您想使用哪種語言安裝？
Slovenščina Slovenian
Shqip Albanian
Српски Serbian
Svenska Swedish
தமிழ் Tamil
తెలుగు Telugu
Тоҷикӣ Tajik
ไทย Thai
Türkçe Turkish
Українська Ukrainian
اردو Urdu
Tiếng Việt Vietnamese
中文 Chinese
IsiZulu Zulu
在此處輸入可搜尋。
简体中文 (中国)
繁體中文 (台灣)
繁體中文 (中華人民共和國香港特別行政區)
简体中文 (新加坡)
退出(Q)
繼續(C)
Right Ctrl


Oracle VM VirtualBox 管理員
檔案(F) 機器(M) 說明(H)
新增(N) 設定值(S) 捨棄 啟動(T)
詳細資料(D) 快照(S)
CentOS7
電源關閉
一般
名稱: CentOS7
作業系統: Red Hat (64-bit)
系統
基礎記憶體: 1024 MB
處理器: 4
開機順序: 軟碟, 光碟機, 硬碟
加速: VT-x/AMD-V, 巢式分頁, PAE/NX, KVM 半虛擬化
預覽
CentOS7
顯示


CentOS7-2 - 設定
一般
系統
顯示
存放裝置
音效
網路
串列埠
USB
共用資料夾
使用者介面
網路
介面卡 1
介面卡 2
介面卡 3
介面卡 4
啟用網路卡(E)
附加到(A):
NAT
名稱(N):
進階(D)
OK
Cancel
1
2
3
4


CentOS7-2 - 設定
一般
系統
顯示
存放裝置
音效
網路
串列埠
USB
共用資料夾
使用者介面
網路
介面卡 1
介面卡 2
介面卡 3
介面卡 4
啟用網路卡(E)
附加到(A):
「僅限主機」介面卡
名稱(N):
VirtualBox Host-Only Ethernet Adapter
進階(D)
介面卡類型(T):
Intel PRO/1000 MT Desktop (82540EM)
混合模式(P):
允許 VM
MAC 位址(M):
080027E5B4E2
線路已連接(C)
連接埠轉送(P)
OK
Cancel
1
2
3
4
5


Oracle VM VirtualBox 管理員
檔案(F) 機器(M) 快照(S) 說明(H)
工具
擷取(T) 刪除(D) 還原(R) 內容(P) 再製(C) 設定(S) 捨棄 啟動(T)
centos7 再製 (快照 3)
已關閉電源
centos7 再製 再製 (快照 1)
已關閉電源
名稱
取得
快照 1
2020/5/19 上午 11:58 (8 小時 前)
目前狀態


Applications Places
Fri 13:37
Wikipedia
home
Trash
7
CENTOS
1 / 4

模擬環境

軟件模擬

能夠安裝多個客戶端操作系統，每個客戶端系統皆可獨立開啟、暫停與停止。主端操作系統與客戶端操作系統皆能相互通訊，多個操作系統同時運行的環境，也彼此能夠同時使用網絡。

硬件模擬

VirtualBox 支持 Intel VT-x 與 AMD AMD-V 硬件虛擬化技術。

硬盤被模擬在一個稱為虛擬磁盤映像檔（Virtual Disk Images）的特殊容器，此格式不相容於其它虛擬機平台運行，通常作為一個系統檔存放在主機端操作系統（副檔名.vdi）。 VirtualBox 能夠連結 iSCSI，且能在虛擬硬盤上運作，此外 VirtualBox 可以讀寫 VMware VMDK 檔與 VirtualPC VHD 檔。

ISO 映像檔可以被掛載成 CD/DVD 裝置，例如下載的 Linux 發行版 DVD 映像檔可以直接使用在 VirtualBox，而不需燒錄在光碟片上，亦可直接在虛擬機上掛載實體光驅。

默認上 VirtualBox 提供了一個支援 VESA 相容的虛擬顯卡，與一個供 Windows、Linux、Solaris、OS/2 客戶端系統額外的驅動程式（guest addition），可以提供更好的效能與功能，如當虛擬機的視窗被縮放時，會動態的調整分辨率。在 4.1 更支援 WDDM 相容的虛擬顯卡，令 Windows Vista 及 Windows 7 可以使用 Windows Aero。

在聲卡方面，VirtualBox 虛擬一個 Intel ICH AC97 聲卡與 SoundBlaster 16 聲霸卡。

在以太網接口卡方面，VirtualBox 虛擬了數張網絡卡：AMD PCnet PCI II、AMD PCnet-Fast III、Intel Pro/1000 MT Desktop、Intel Pro/1000 MT Server、Intel Pro/1000 T Server。

網絡設置

VirtualBox 提供了多種網絡接入模式，他們各有優缺點，用戶可以根據自己的需要進行選擇。

1、NAT 模式：最簡單的實現虛擬機上網的方式，無需配置，默認選擇即可接入網絡。虛擬機訪問網絡的所有數據都是由主機提供的，訪問速度較慢，和主機之間不能互相訪問。

2、Bridged Adapter 模式：即網橋模式，可以為虛擬機模擬出一個獨立的網卡，有獨立的 IP 地址，所有網絡功能和主機一樣，並且能夠互相訪問，實現文件的傳遞和共享。 （注： Windows 7 系統選擇網橋模式時，需要手動安裝 VirtualBox 的橋接服務驅動。在本地連接的屬性選項中，選擇“Microsoft 網絡客戶端”點擊安裝，網絡功能類型選擇“服務”點擊添加，選擇從磁盤安裝，找到驅動路徑“Oracle\VirtualBox\drivers\network\netflt”，選擇 VBoxNetFlt_m 文件安裝完成。）

3、Internal 模式：即內網模式，虛擬機與外網完全斷開，只實現虛擬機於虛擬機之間的內部網絡模式，和主機之間不能互相訪問，就相當於虛擬機之間架設了一個獨立的局域網。

4、Host-only Adapter 模式：即主機模式，是所有接入模式中最複雜的一種，需要有比較紮實的網絡基礎知識才行。前面幾種模式所實現的功能，通過虛擬機及網卡的設置都可以被實現。

虛擬機參數

1、虛擬機名稱和系統類型：為將要創建的虛擬機命名，要求是唯一的標識，用來區分該虛擬機硬件配置、操作系統、軟件等數據。並選擇將要安裝的操作系統類型和版本，以便 VirtualBox 自動配置合適的硬件環境。

2、內存：指定虛擬機可用內存大小，系統會自動分配，也可自行設置。

3、虛擬硬盤：選擇一個虛擬硬盤作為主硬盤，也可以新建一個，第一次創建，默認即可。如果是選擇新建，將進入硬盤類型選擇界面（VDI：VirtualBox 的格式，VMDK：VM 虛擬機的格式，VHD：微軟 VirtualPC 虛擬機的格式，HDD：Parallels 虛擬機的格式），默認選擇 VDI 即可。另外，幾種格式都可以相互轉換，網上有相應的轉換軟件。

4、硬盘存儲類型：分為動態擴展和固定大小兩種，其中動態擴展類型最初只需佔用非常小的物理硬盘空間，然後根據虛擬機的實際需求動態分配，固定大小類型就是建立時就分配指定的大小給虛擬機使用。後者在性能上有一定優勢，但建立時間較長；

5、摘要：顯示虛擬機的各項數據情況，確定後完成虛擬機的創建。

認識 Docker

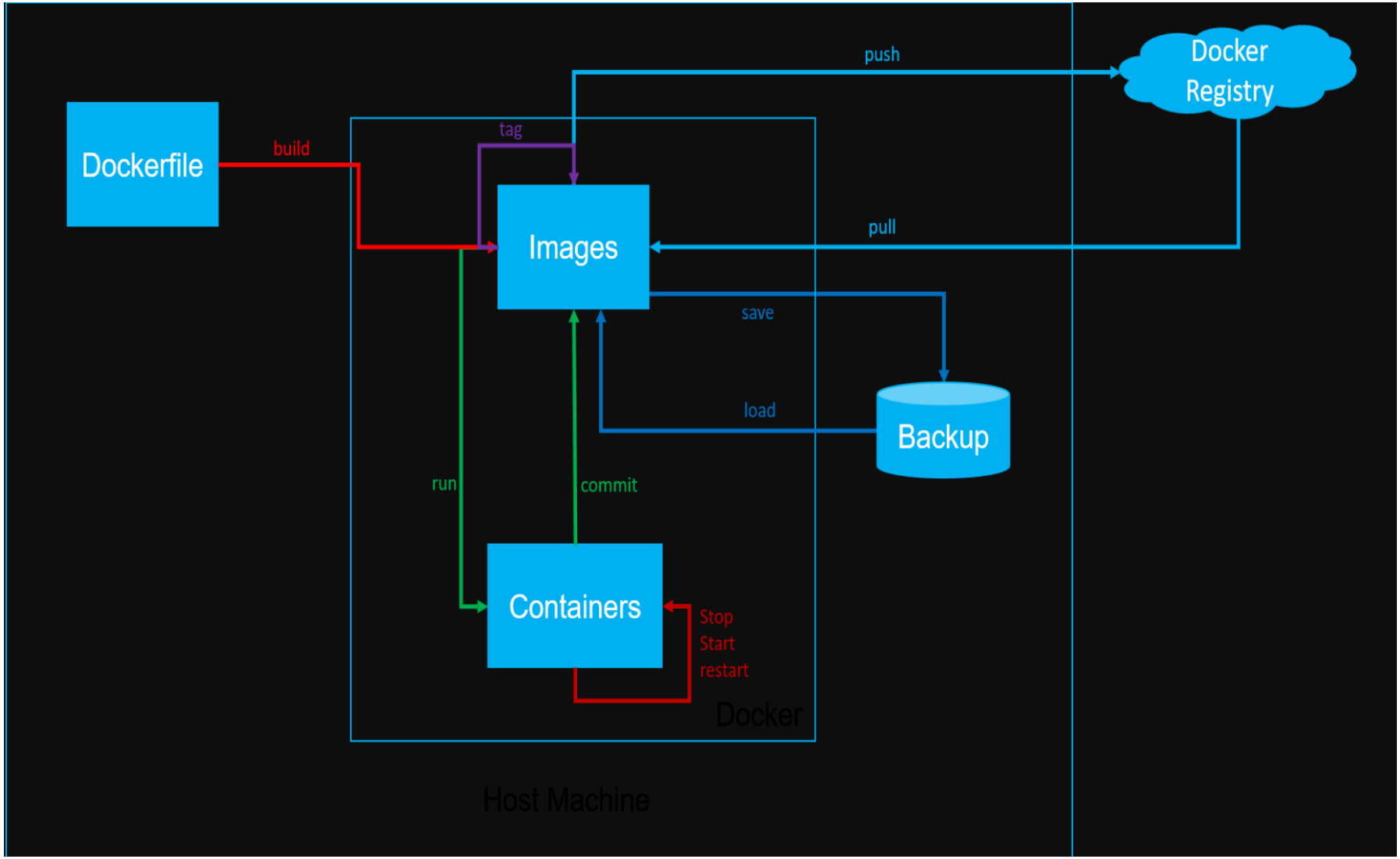
Docker 是一個應用平台，它是一種全新的應用程式的運行方式，一切都發生在一個封閉的、精簡的單元裡，此單元我們稱之為容器(containers)。

容器是一種非常經濟的應用程式運行方式，它們可以在數秒之內啟動，且不會對應用程式的記憶體和運算經濟需求帶來額外的負擔。

你可以透過 Docker 執行 Node.js 的程式之外，也可以啟動另外一個容器運行已存在好幾年的 asp.net MVC 的網頁程式。

Docker、Dockerfile 與 Container 等關係

要了解並學習 Docker 前，我們先來看一下映像檔(Images)、登入所(registries)、容器、以及 Dockerfile 等關係，並透過我所製作的下圖來理解 Docker 的運作方式。


Dockerfile
build
tag
push
Docker
Registry
Images
pull
save
load
Backup
run
commit
Containers
Stop
Start
restart
Docker
Host Machine

Containers

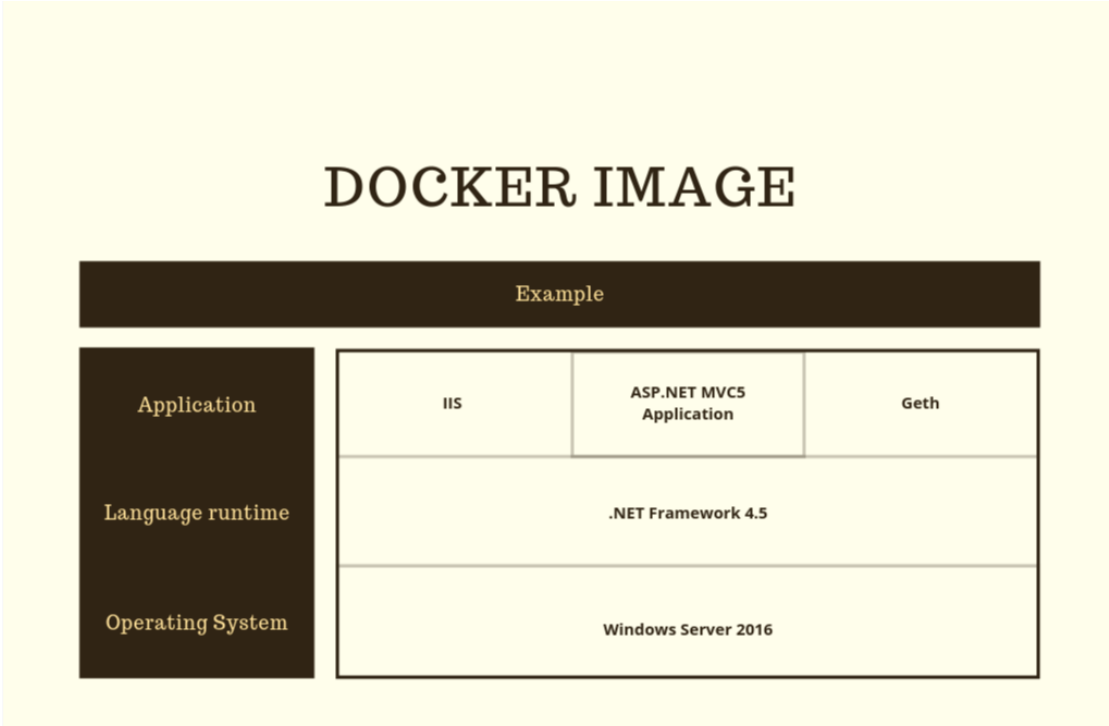
容器是一個源自映像檔的應用程式執行個體。當運行一個容器時，Docker 會根據 Image 的內容來做該做的事情。我們透過下達

```
>docker container run
```

來啟動應用程式。

Images

一個 Docker 映像檔，其實就是一個完整的應用程式封裝。一個 Images 裡面包含了應用程式以及它運行所需的所有相關成分，包括底層作業系統，語言執行平台以及應用程式。如圖所示：



我們可以透過下達

>docker image build

就可以建立客製化的映像檔。我們須提供 Dockerfile 本身以及任何相關資訊如需封裝在映像檔內的資源(例如 WebApp)來建構此映像檔。

Docker 映像檔如同應用程式在某個版本狀態時的檔案系統快照。

Dockerfile

Dokcerfile 即指令稿，它可以用以客製化整個 Image 要用什麼 OS，要準備好什麼樣的環境，透過使用docker build的指令就可以建構起 Docker Image，未來要使用此 image 就直接 Run 此 Image。

以下一個 **Dockerfile** 的內容範例是來自於 [ASP.NET Core 2.2 的範例](#)：

```
FROM mcr.microsoft.com/dotnet/core/sdk:2.2 AS build
WORKDIR /app

# copy csproj and restore as distinct layers
COPY *.sln .
COPY aspnetapp/*.csproj ./aspnetapp/
RUN dotnet restore

# copy everything else and build app
COPY aspnetapp/. ./aspnetapp/
WORKDIR /app/aspnetapp
RUN dotnet publish -c Release -o out


FROM mcr.microsoft.com/dotnet/core/aspnet:2.2 AS runtime
WORKDIR /app
COPY --from=build /app/aspnetapp/out ./
ENTRYPOINT ["dotnet", "aspnetapp.dll"]
```

Dockerfile hosted with ❤ by GitHub view raw

registries

登錄我們公有或自有的映像檔的地方。只要你有權使用該映像檔，

就可以透過指令

> docker image push

上傳映像檔，或用

> docker image pull

下載映像檔。

感謝觀看