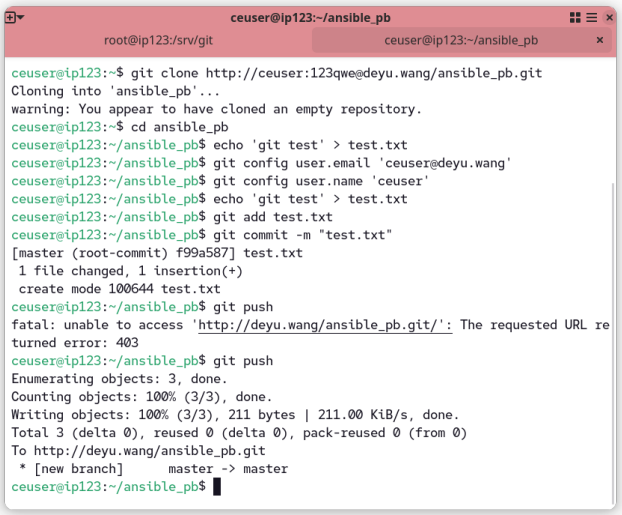
Ansible – Linux 自動化部署 (RHCE10)

Department of Computer Science and Information Engineering
Chaoyang University of Technology
Taichung, Taiwan, Republic of China

Instructor: De-Yu Wang (王德譽)
E-mail: dywang@mail.cyut.edu.tw
Phone: (04)23323000 ext 4538
Office: E738



Thursday 11th June, 2026

- Instructor: De-Yu Wang
    1. Email: dywang@csie.cyut.edu.tw
    2. Homepage: http://dywang.csie.cyut.edu.tw
    3. 本文件 pdf 檔下載
    4. Phone: (04)23323000 ext 4538
    5. Office: E738
- 參考資料
    1. RedHat Documentation
    2. Ansible User Guide
    3. Ansible 中文權威指南
    4. 現代 IT 人一定要知道的 Ansible 自動化組態技巧

Contents

1 實機練習系統 **1**

1.1 動機 1
1.2 系統設計 1
1.3 練習系統製作-VirtualBox 2
1.4 練習系統製作-Raw Image 5
1.5 NAT 設計 6
1.6 虛擬機管理界面 7
1.7 虛擬機管理界面 V2 8
1.8 系統管理與評分 9
1.9 系統註冊–上課前先確定三件事 11
1.10 虛擬機設定 12
1.11 系統伺服器管理 13

2 GIT 版本控制 **16**

2.1 Git 簡介 16
2.2 Git 命令 17
2.3 * 建立 Git Repository 17
2.4 *Http Git Server 19
2.5 *http 存取 Git Repository 20
2.6 Git Clone 21
2.7 Git Config 22
2.8 Git Push 22
2.9 實機練習 23

3 自動部署環境 **25**

3.1 Ansible 簡介 25
3.2 環境介紹 26
3.3 Ansible 命令 26
3.4 * 設定 SSH Keys 27

3.5 * 設定 SUDOER 29
3.6 實機練習 29

4 Inventory 檔案 **30**
4.1 前言 30
4.2 基本設定 30
4.3 變數說明 32
4.4 進階設定 33
4.5 設定 inventory 34
4.6 檢查 inventory 35
4.7 實機練習 35

5 組態設定 **37**
5.1 前言 37
5.2 ansible.cfg 範例 37
5.3 練習系統 ansible.cfg 38
5.4 測試 ansible.cfg 39
5.5 實機練習 40

6 Ad-hoc 命令 **41**
6.1 前言 41
6.2 模組文件 41
6.3 檔案傳輸 - copy 模組 42
6.4 檔案管理 - file 模組 43
6.5 實機練習 46

7 Playbooks **47**
7.1 前言 47
7.2 基本語法 47
7.3 yum_repository 模組 48
7.4 第一支 playbook 49
7.5 移除套件庫 51
7.6 實機練習 52

8 Dnf 模組 **54**
8.1 文件查詢 54
8.2 使用變數 55
8.3 loop 迴圈 56
8.4 block 區塊 58

8.5 更新套件 59
8.6 實機練習 60

9 Collections 內容集 **62**
9.1 前言 62
9.2 ansible-galaxy 指令 63
9.3 * 初始化內容集 66
9.4 * 編輯打包壓縮內容集 67
9.5 預設路徑安裝內容集 68
9.6 * 指定路徑安裝內容集 69
9.7 實機練習 70

10 Roles - RHEL System **71**
10.1 前言 71
10.2 內容集的角色文件 71
10.3 使用 RHEL 角色 72
10.4 實機練習 76

11 Roles - Ansible Galaxy **77**
11.1 前言 77
11.2 ansible-galaxy 命令 77
11.3 search 78
11.4 install/remove 80
11.5 requirements.yml 81
11.6 requirements.yml 安裝實例 82
11.7 實機練習 83

12 Jinja2 樣板 **85**
12.1 前言 85
12.2 基本語法 85
12.3 變數 86
12.4 debug 輸出 magic 變數 87
12.5 setup 輸出 facts 變數 90
12.6 產生 hosts 檔案 91
12.7 測試 hosts.yml 92
12.8 實機練習 93

13 Create Roles **95**
13.1 項目結構 95
13.2 初始化角色 apache 96
13.3 編輯角色 apache 任務 97
13.4 編輯角色 apache 樣板 98
13.5 實機練習 100

14 Use Roles **102**
14.1 使用 apache 角色 102
14.2 使用 haproxy 角色 103
14.3 使用 phpinfo 角色 105
14.4 實機練習 107

15 Shell 模組 **109**
15.1 前言 109
15.2 模組文件 109
15.3 修改檔案–單一群組 111
15.4 修改檔案–不同群組不同內容 113
15.5 實機練習 114

16 硬碟分割掛載 **116**
16.1 相關模組 116
16.2 parted 模組 - 新增分割區 119
16.3 filesystem 模組 - 檔案系統格式化 121
16.4 mount 模組 - 掛載檔案系統 122
16.5 解除掛載並移除分割區 124
16.6 實機練習 126

17 LVM 檔案系統 **127**
17.1 LVM 相關模組 127
17.2 lvg 模組 127
17.3 刪除 VG 130
17.4 刪除分割區 131
17.5 lvol 模組 132
17.6 filesystem 模組 134
17.7 移除 lv 135
17.8 實機練習 136

18 Block 錯誤處理 **138**
18.1 Block-rescue-always 138
18.2 變數查詢 139
18.3 實例–正常任務 140
18.4 實例–rescue 142
18.5 實例–debug 143
18.6 實機練習 144

19 Group 模組 **146**
19.1 前言 146
19.2 模組文件 146
19.3 建立群組帳號 146
19.4 實機練習 147

20 File 模組 **149**
20.1 前言 149
20.2 模組文件 149
20.3 建立目錄 152
20.4 建立軟連結 153
20.5 新增含內容的檔案 155
20.6 實機練習 157

21 Sefcontext 模組 **159**
21.1 前言 159
21.2 模組文件 159
21.3 修改 setype 160
21.4 實機練習 161

22 Lineinfile 模組 **163**
22.1 前言 163
22.2 模組文件 163
22.3 Facts 變數查詢 166
22.4 修改檔案內容 171
22.5 變數不存在 173
22.6 實機練習 175

23 Vault 加密 **177**
23.1 前言 177
23.2 ansible-vault 命令 177

23.3 建立加密檔案 178
23.4 加解密現有檔案 179
23.5 密碼檔解密 Vault 180
23.6 rekey 重設密碼 181
23.7 實機練習 182

24 User 模組 **184**
24.1 前言 184
24.2 模組文件 184
24.3 建立用户 186
24.4 設定帳號密碼 187
24.5 設定密碼過期 189
24.6 Vault 密碼檔 190
24.7 建立不同帳號 193
24.8 測試帳號 197
24.9 實機練習 198

25 Cron 模組 **200**
25.1 前言 200
25.2 模組文件 200
25.3 建立例行性命令 202
25.4 實機練習 203

Chapter 1

實機練習系統

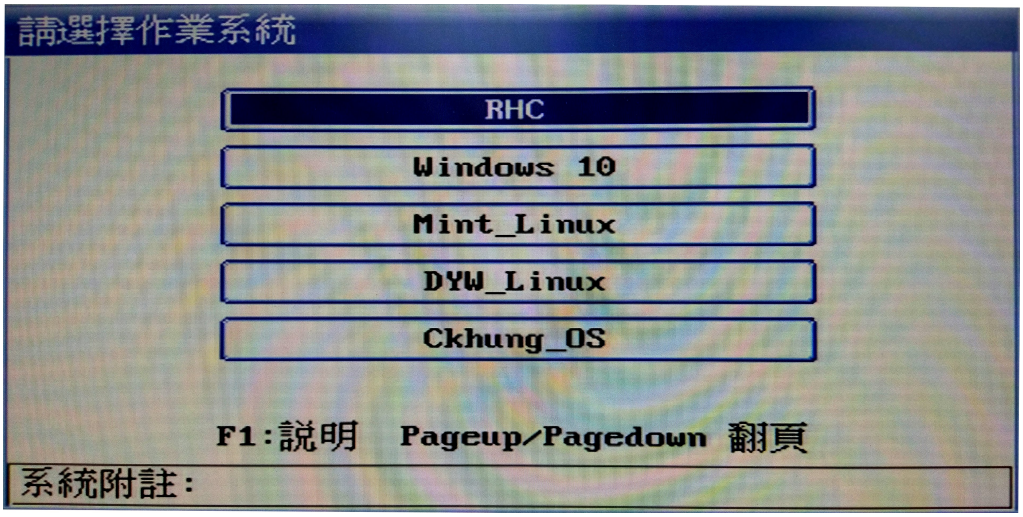
1.1 動機

1. 教學若有實際環境，讓學生實機操作練習，可以大大提昇學習成效。
2. Linux 實機練習，很容易因學生操作錯誤，而造成系統損壞。例如：硬碟分割、格式化及管理的練習。
3. 實機練習結束或操作不當造成當機，系統必須可以迅速還原，才能重複練習。
4. 系統必須可以批次快速修改或更新。
5. 系統必須可以依據學生實際操作狀況，批次進行檢查或評分。
6. 虛擬機練習所需伺服器資源，例如：LDAP 網路帳號、YUM repository 套件庫伺服器、NFS 分享、NTP 校時伺服器、ISCSI 硬碟分享、內部網域 DNS 伺服器、FTP 資源下載... 等環境必須架設好，才能提供虛擬機做各種狀況的練習。

1.2 系統設計

1. Linux 教學系統設計
   (a) 客製化 Linux 系統。
   (b) Linux 伺服器架設。
   (c) VNC 遠端桌面與 KVM 虛擬機。
2. Linux 教學系統安裝

(a) E517 電腦教室：方便學生上課使用。



(b) 個人隨身碟：插上隨身碟，以安裝光碟開機，不需要任何設定，自動安裝於隨身碟，不會影響電腦中的硬碟。

(c) LiveCD：以 LiveCD 光碟開機，開機後即可使用，只是所有操作與設定都是暫存，重新開機後即回到原始狀態。

1.3 練習系統製作-VirtualBox

1. 下載 DYW Linux 練習系統 vdi 檔，下載網址公告於本平台課程內，請登入查看。

```
[root@dywmsi ~]# unxz crt9.vdi.xz
```

2. 依自己使用的 OS (Mac Windows Linux) 下載安裝 VirtualBox。

```
https://www.virtualbox.org/wiki/Downloads
```

3. DYW Linux 練習系統，Host 主機以 virsh 管理 VM，最低硬體因所需 VM 個數不同：

(a) CPU 4 核心以上，若修的課不用內部虛擬機，設定 2 核心也可以。

(b) RAM 8G 以上，若修的課不用內部虛擬機，設定 4G RAM 也可以。

(c) RHCSA, RHCE 請設定 4 核心，8G RAM 以上

(d) CPU 支援虛擬化技術 (virtualization technology)，BIOS 啓用 VT。

4. VirtualBox 新增 VM

```
New(Ctrl+N)
    Name: 自訂，例如 crt9
    Type: Linux
    Version: 只要是 64-bit 就可以，例如 Fedora (64-bit)
    Next >
Memory size
    8192 MB
    Next >
Hard disk
    Use an existing virtual hard disk file
    新增解壓縮的 crt9.vdi，並選用這個硬碟檔。
    Create
```

5. 若要使用內部虛擬機，必須保證巢式虛擬化有開啓，假設匯入的 VM 名稱爲 crt9

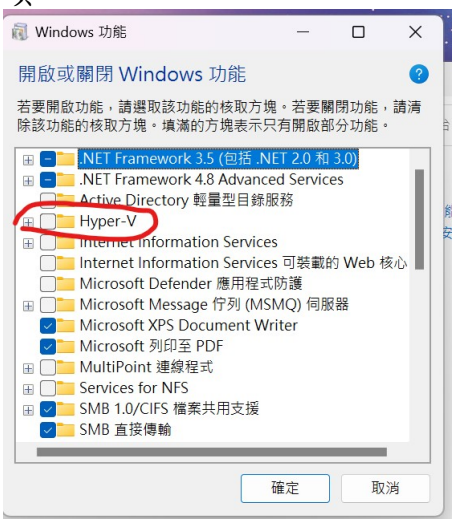
   (a) Host 是 Linux，在終端執行以下命令：

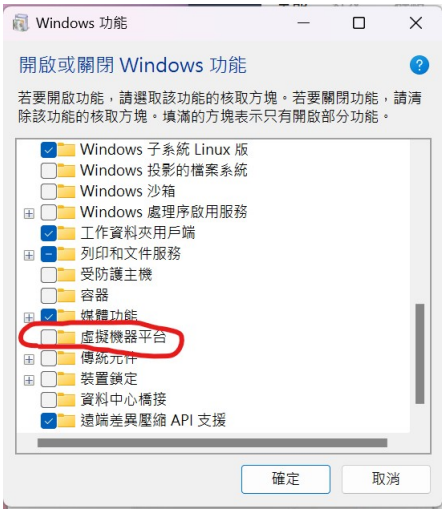
   ```
   [dywang@dywmsi ~]$ VBoxManage modifyvm crt9 --nested-hw-virt on
   ```

   (b) Host 是 window，在命令提示 cmd 執行以下命令：

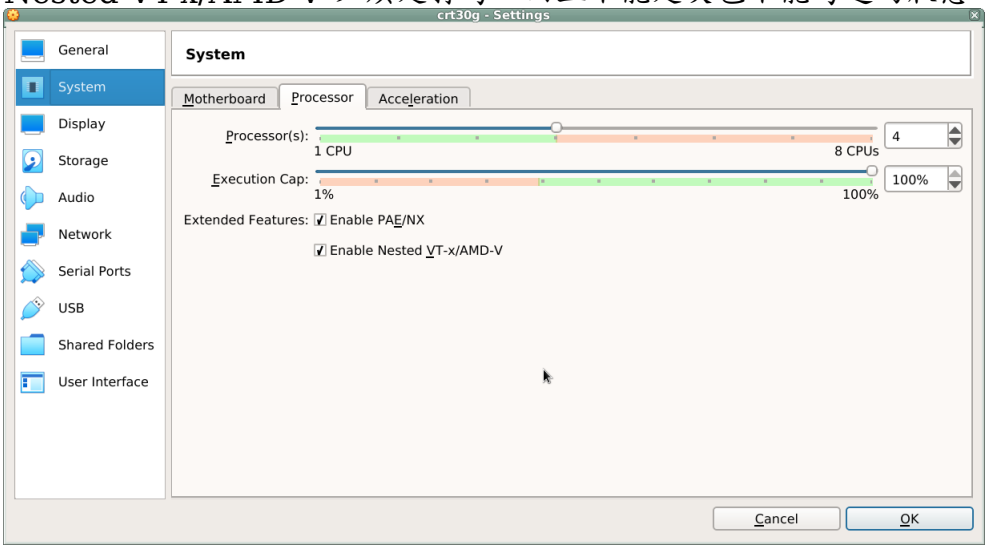
   ```
   cd "c:\Program Files\Oracle\VirtualBox"
   VBoxManage.exe modifyvm crt9 --nested-hw-virt on
   ```

   (c) Host 是 window，還必須關閉「Hyper-V」及「虛擬機器平台」，兩個選項。

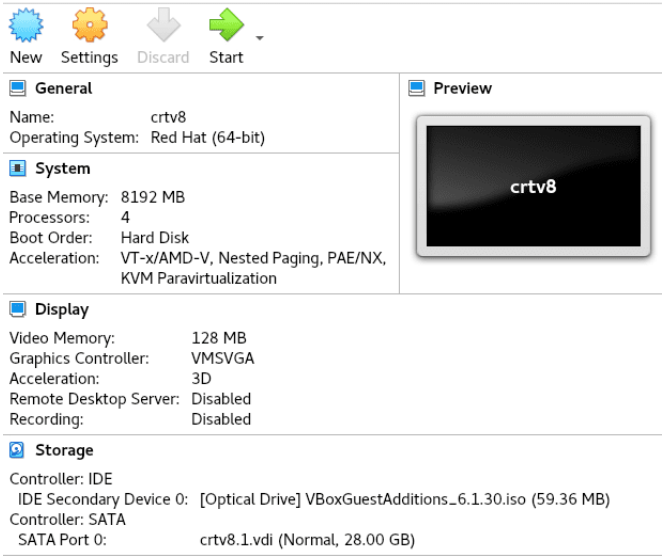




6. 進 VirtualBox 查看 VM crt9 的設定，system -> Processor -> Enable Nested VT-x/AMD-V 必須是打勾，而且不能是灰色不能勾選的狀態。



7. 使用 VirtualBox 或其他虛擬化工具，都要提供 VM crt9 足夠的 CPU 及 RAM，且要開啓巢式虛擬化，只要這三個要求達到，應該不會有□題，其他設定不懂可以先用預設值。若進不了圖形界面，可以打開 Display 的 3D 加速看看，VirtualBox VM 設定 (Settings) 參考值如下：



8. * 爲方便 VirtualBox 使用，建議下載附加套件 Oracle_VM_VirtualBox_Extension_Pack-6.1.30.vbox-extpack，放到你的 VirtualBox 目錄中，並於 VirtualBox 選單中加入附加 CD image。

```
Devices --> Insert Guest Additions CD image...
```

9. 開啓 DYW Linux 後，若虛擬機沒安裝，請先 rebuild 安裝虛擬機，練習後只需要 revert 還原，不用再要重新 rebuild，rebuild 要從學校 server 下載 image，要花較多時間。

1.4 練習系統製作-Raw Image

1. 下載 DYW Linux 練習系統 image 檔，或由 VDI 檔轉換格式，下載網址公告於本平台課程內，請登入查看。

2. 下載的 image 檔是 xz 壓縮檔，先用 unxz 解壓縮，windows 可用 winzip 解壓縮。

```
1 [root@dywmsi ~]# unxz crt9.img.xz
```

3. DYW Linux 練習系統，Host 主機以 virsh 管理 VM，最低硬體因所需 VM 個數不同：

   (a) CPU 4 核心以上，若修的課不用內部虛擬機，設定 2 核心也可以。

   (b) RAM 8G 以上，若修的課不用內部虛擬機，設定 4G RAM 也可以。

(c) RHCSA, RHCE 請設定 4 核心，8G RAM 以上

(d) CPU 支援虛擬化技術 (virtualization technology)，BIOS 啓用 VT。

4. 將 image 檔 dd 寫到 32G 以上硬碟或隨身碟，其中 /dev/sdX 是您要寫入的硬碟或隨身碟。

```
1 [root@dywmsi ~]# dd if=crt9.img of=/dev/sdX bs=100M
```

5. /dev/sdX 整顆硬碟的資料會被覆蓋，請確認要寫入的硬碟無誤再按 Enter 執行。

6. dd 完成後以 /dev/sdX 開機。

7. 開啓 DYW Linux 後，若虛擬機沒安裝，請先 rebuild 安裝虛擬機，練習後只需要 revert 還原，不用再要重新 rebuild，rebuild 要從學校 server 下載 image，要花較多時間。

1.5 NAT 設計

1. NAT 架設：電腦教室網路設置、管控學生上課上網。

2. 在 Linux，一行命令即可完成設定；在 Windows，Honda 老師亦幫忙撰寫批次檔。控制方式：

3. 任何設定皆不管控老師機，老師機可自由上網。

4. 學生機上網方式：

(a) 自由上網

(b) 不能上網

(c) 限制只能上學內網

(d) 限制只能上某一網站

(e) 限制只能上「依老師設定」的部分網站

(f) 只有某一學生機可自由上網

5. 不管老師做任何網路設定，下課後系統自動恢復成所有電腦皆可自由上網，以避免因老師忘了復原，而影響下節上課的老師。

1.6 虛擬機管理界面

1. 設計虛擬機管理 GUI 程式，方便學生管理虛擬機。程式包含虛擬機重安裝、還原、重新啓動及終端界面等四個功能。

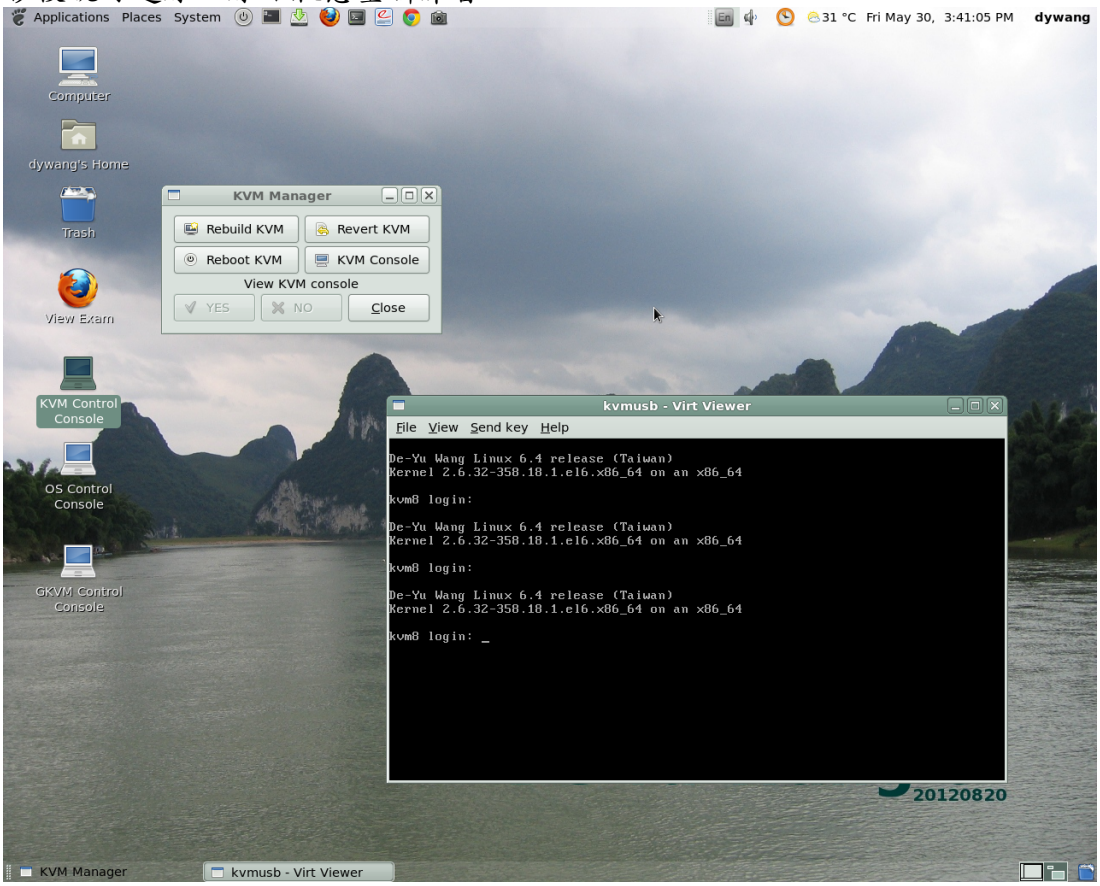
   (a) Rebuild KVM：重新安裝 KVM 虛擬機，約 10 幾分鐘。

   (b) Revert KVM：還原 KVM 虛擬機，約幾秒鐘到幾分鐘，請耐心等侯，不要在還原過程按管理界面的其他功能，以免造成虛擬機損壞。

   (c) Reboot KVM：重新啓動 KVM 虛擬機。

   (d) KVM Console：打開 KVM 虛擬機終端畫面。

2. 系統桌面中黑色背景視窗爲 KVM 虛擬機操作界面。原始 KVM 虛擬機主機名稱爲 111111，因已練習過，主機也已被設定成 kvm8，點擊 Revert KVM，幾秒後就可還原回原始狀態重新練習。



3. 在終端機以指令 virt-view 直接開啓虛擬機，出現錯誤訊息。

```
[dywang@dywH ~]$ sudo virt-viewer kvmusb
No protocol specified
Cannot open display:
Run 'virt-viewer --help' to see a full list of available command line
    options
```

4. 先檢查 Xwindow 認證變數 $XAUTHORITY 存不存在？

```
[dywang@dywH ~]$ echo $XAUTHORITY
```

5. 變數 XAUTHORITY 不存在，必須先設定 XAUTHORITY=~/.Xauthority 後再使用 sudo 啓動虛擬機，才能正常啓動。

```
[dywang@dywH ~]$ XAUTHORITY=~/.Xauthority sudo virt-viewer kvmusb
```

1.7 虛擬機管理界面 V2

1. 設計虛擬機管理 GUI 程式，方便學生管理虛擬機。程式包含虛擬機重安裝、還原、重新啓動、終端界面、啓用實體 USB 隨身碟、解除實體 USB 隨身碟的使用等六個功能。



(a) Rebuild KVM：重新安裝 KVM 虛擬機，直接下載 qcow2 映像檔，產生虛擬機，校內網路約 1 分多鐘。

(b) Revert KVM：還原 KVM 虛擬機，約幾秒鐘到幾分鐘，請耐心等侯，務必等黑色終端出現登入畫面，表示還原完成。練習前先關閉黑色終端畫面讓還原程式拍照，記下還原狀態，以免造成虛擬機損壞。

(c) Reboot KVM：重新啓動 KVM 虛擬機。

(d) KVM Console：打開 KVM 虛擬機終端畫面。

(e) Attach USB：插上 USB 隨身碟，按下此按鈕，KVM 虛擬機啓動實體 USB 隨身碟。

(f) Detach USB：KVM 虛擬機解除實體 USB 隨身碟的使用。

1.8 系統管理與評分

1. 系統管理腳本程式自動掃瞄目前開機的電腦，選擇要進行的工作，輸入密碼後即可自動完成。

```
[root@dywH ~]# e517sshpass.sh
myip=104
hosts=103,105
q(quit) iscsi halt reboot c(clear) l(lvm) rhcsa rhce
>  t(mounttest) k(kvmreboot) v(kvmrevert) s(servicerestart)
>  vi1 c1 lpvar c999 sh999 hp(hostpatch)? c999
keyin remote host root password:
```

2. 預設工作項目：

   (a) q(quit)：退出，無動作。

   (b) iscsi：學生機 ISCSI 硬碟分享架設。

   (c) halt：學生機關機。

   (d) reboot：學生機重新啓動。

   (e) c(clear)：登入學生機後清除部分資料，此部分保留修改彈性，目前入作爲限制學生機互相登入之限制，以防代作答情況。

   (f) l(lvm)：硬碟分割、格式化、自動掛載評分。

   (g) rhcsa：RHCSA 考照練習總評分。

   (h) rhce：RHCE 考照練習總評分。

   (i) t(mounttest)：測試學生機是否有掛載隨身碟或光碟。

   (j) k(kvmreboot)：學生機 KVM 虛擬機重新啓動。

   (k) v(kvmrevert)：學生機 KVM 虛擬機還原。

   (l) s(servicerestart)：啓動學生機主機服務。

   (m) vi1：vi 編輯器實機練習評分。

   (n) c1：C 語言第一支程式練習評分。

   (o) lpvar：shell 變數變化練習評分。

   (p) c999：C 語言程式實機練習評分。

   (q) sh999：Shell script 程式練習評分。

   (r) hp(hostpatch)：學生機網路修正後重新啓動網路。

3. 遠距教學，在家練習，自己評分上傳：

(a) 登入本平台，下載評分腳本及參數檔，以「資料庫」第一次評分爲例。

(b) 切換身分爲 root，root 密碼在平台部落格，請登入。

```
[dywang@deyu ~]$ su -
Password:
[root@deyu ~]#
```

(c) 切換工作目錄到評分程式的下載目錄，用瀏覽器下載預設是放在家目錄的 Downloads。

```
[root@deyu ~]# cd /home/dywang/Downloads
```

(d) 下載的評分程式沒有可執行的權限，先增加評分程式 mariadbeval1.sh.x 可執行屬性，使評分腳本 mariadbeval1.sh.x 成爲可執行檔。

```
[root@deyu Downloads]# chmod +x mariadbeval1.sh.x
```

(e) 查看目前有兩個檔案，評分程式爲 mariadbeval1.sh.x，參數檔爲 mariadbpara1.sh。

```
[root@deyu Downloads]# ll
-rwxrwxr-x. 1 root root 58656 May 21 13:29 mariadbeval1.sh.x
```

(f) 執行評分腳本 mariadbeval1.sh.x

```
[root@deyu Downloads]# ./mariadbeval1.sh.x
20210521-13:46:36
./mariadbeval1.sh.x
163.17.10.123
kvm=kvm19
163.17.10.123 10927000 王大呆
mariadbpara1.sh
mariadbeval1.sh.x_kvm19_1.mysqlversion:0
mariadbeval1.sh.x_kvm19_2.mariadb_enabled:0
mariadbeval1.sh.x_kvm19_3.mariadb_active:0
mariadbeval1.sh.x_kvm19_4.rootpw:0
```

(g) 如果評分參數檔不在同目錄，評分腳本 mariadbeval1.sh.x 無法執行。

```
[root@deyu Downloads]# mv mariadbpara1.sh /tmp/
```

```
[root@deyu Downloads]# ./mariadbeval1.sh
20210521-13:48:51
./mariadbeval1.sh
163.17.10.123
kvm=kvm19
163.17.10.123 10927000 王大呆
ls: cannot access mariadbpara1: No such file or directory
NO mariadbpara1 file
```

1.9 系統註冊-上課前先確定三件事

1. E517 電腦教室

   (a) E517 教室位置，靠窗爲第一排，確認自己電腦位置爲?-?。例如：1-1。

   (b) 在伺服器 (也就是開機進入圖形界面中有虛擬機管理界面的這台主機)，不是虛擬機，確認自己的電腦 IP。以下例子爲 192.168.1.140，IP 尾數爲 140。

```
[dywang@deyu ~]$ ifconfig | grep 192
          inet addr:192.168.1.140  Bcast:192.168.1.255  Mask
              :255.255.255.0
          inet addr:192.168.122.1  Bcast:192.168.122.255  Mask
              :255.255.255.0
```

   (c) 在伺服器 deyu.wang 主機一般帳號 dywang 家目錄，產生學號姓名檔 sid，檔案資訊格式爲「電腦位置 IP 尾數學號姓名」，每一欄位中間以一個空格隔開。以下例子電腦位置 1-11、IP 尾數 199、學號 123456，姓名王大呆。

```
[dywang@deyu ~]$ echo '1-11 199 123456 王大呆' > ~/sid
[dywang@deyu ~]$ cat ~/sid
1-11 199 123456 王大呆
```

2. 在家練習

   (a) 查詢練習系統對外 public ip 有三種方式：

```
[dywang@deyu ~]$ curl -4 ipinfo.io/ip
163.17.10.122[dywang@deyu ~]$
[dywang@deyu ~]$ curl -4 icanhazip.com
163.17.10.122
```

```
[dywang@deyu ~]$ curl -4  ifconfig.me
163.17.10.122[dywang@deyu ~]$
```

(b) 將 ip 學號姓名寫入 dywang 家目錄的 sid 檔，如下範例：

```
[dywang@deyu ~]$ echo '163.17.10.123 10827000 王大呆' > sid
```

(c) 查看 sid，ip 學號姓名務必正確，以免評分錯誤或無法匯入成績。

```
[dywang@deyu ~]$ cat sid
163.17.10.123 10827000 王大呆
```

1.10 虛擬機設定

1. 伺服器(也就是開機進入圖形界面中有虛擬機管理界面的這台主機)，不確定的話，先看命令列提示符號是不是 [dywang@deyu ~]$，或是 ifconfig 查目前所在主機的 IP：

   (a) Server 主機有兩個名稱 deyu.wang 及 server.deyu.wang

   (b) Server 與虛擬機網域 deyu.wang，使用網段 192.168.122.0/24。

   (c) Server IP 爲 192.168.122.1，虛擬機 IP 爲 192.168.122.X ，其中 X 依題目要求設定。

2. 虛擬機環境說明：

   (a) 虛擬機主機名稱 kvmX.deyu.wang IP 192.168.122.X，X 爲虛擬機編號，例如 5 或 7。

   (b) 虛擬機 root 密碼無法取得，統一重新設定爲 123qwe。

   (c) RHCE 練習環境增加
   網域 my111.wang，使用網段 192.168.111.0/24，
   server 主機增加此網域 IP 192.168.111.1。

   (d) 虛擬機網路設定：

```
IP Address:   192.168.122.X
Netmask:      255.255.255.0
Gateway:      192.168.122.1
Name server:  192.168.122.1
```

(e) RHCSA/RHCE 6,7 repository 套件庫網址：
http://dywang.csie.cyut.edu.tw/centos7.

(f) RHCSA 8 repository 套件庫網址：

i. BaseOS repo server:
http://dywang.csie.cyut.edu.tw/centos8/BaseOS.

ii. AppStream repo server:
http://dywang.csie.cyut.edu.tw/centos8/AppStream.

(g) kernel packages: 網路下載核心 RPM 檔
http://dywang.csie.cyut.edu.tw/kernel/centos7.

(h) LDAP server 設定：

i. 系統 deyu.wang

ii. base DN 爲 dc=deyu,dc=wang

iii. 證書 ftp://deyu.wang/pub/cacert.pem.

1.11 系統伺服器管理

1. 使用虛擬機練習時，伺服器 (也就是開機進入圖形界面中有虛擬機管理界面的這台主機) 必須正常運作，否則有些驗證無法進行。

2. 練習用到的 server 都已寫成腳本，如果運作有問題，可以執行腳本重啓。

1.11.1 DNS server

(a) 如果確定你的練習虛擬機 DNS 設定沒問題，但 ssh 卻無法使用主機名稱連線到虛擬機，可以在 server 機使用 host 命令看是否可以查到虛擬機的 IP？

(b) 沒有 DNS server 可查。

```
[dywang@deyu ~]$ host kvm7.deyu.wang
;; connection timed out; trying next origin
;; connection timed out; no servers could be reached
```

(c) 在 DNS server 查不到 kvm7.deyu.wang 的 IP。

```
[dywang@deyu ~]$ host kvm7.deyu.wang
Host kvm7.deyu.wang not found: 3(NXDOMAIN)
```

(d) 在伺服器 deyu.wang 主機切換成 root 身份。

```
[dywang@deyu ~]$ su -
Password: 123123
[root@deyu ~]#
```

(e) 執行 setdnsmasq.sh 腳本重啓 dnsmasq DNS 服務。

```
[root@deyu ~]# setdnsmasq.sh restart
```

(f) 再使用 host 查詢 kvm7.deyu.wang，已可查到其 IP 爲 192.168.122.7。

```
[root@deyu ~]# host kvm7.deyu.wang
kvm7.deyu.wang has address 192.168.122.7
kvm7.deyu.wang mail is handled by 1 20.
```

(g) 如果執行 setdnsmasq.sh 腳本重啓 dnsmasq DNS 服務未能解決問題，還可以使用 setdns.sh 腳本啓動 named DNS 服務。

```
[root@deyu ~]# setdns.sh restart
```

1.11.2 LDAP server

(a) 如果確認你的虛擬機 LDAP 設定沒問題，但還是無法以 ldapuser1 登入。在伺服器 deyu.wang 主機 (不是你的練習虛擬機)，先切換成 root 身份。

```
[dywang@deyu ~]$ su -
Password: 123123
[root@deyu ~]#
```

(b) 在伺服器執行 setldap1.sh restart 重新產生 LDAP 憑證。因爲憑證是新產生的，所以虛擬機必須再重新執行認證設定更新。

```
[root@deyu ~]# setldap1.sh restart
```

1.11.3 NFS server

(a) 虛擬機設定自動掛載必須查得到 server 有分享目錄，server 的 NFS 不像 LDAP 有憑證過期問題，所以一般都不需要重啓，但如果要重啓 NFS server，先在伺服器 deyu.wang 主機切換成 root 身份。

```
[dywang@deyu ~]$ su -
Password: 123123
[root@deyu ~]#
```

(b) 執行 setnfs.sh 腳本重新啓動。

```
[root@deyu ~]# setnfs.sh restart
```

Chapter 2

GIT 版本控制

2.1 Git 簡介

1. Git 是 Linux 之父 linus Trovalds 開發的版本控制系統，用來管理 Linux kernel 的 source code。
2. 採用快照 (snapshot) 方式記錄版本之間的更動、新增、刪除。
3. 快照方式，只會紀錄版本與版本之間的相異之處，可提升存取速度，也減少儲存的空間。
4. github, gitlab 等網站是儲存空間的 remote repository。
5. Git 順向流程：從工作區存到遠端儲存庫。
   (a) 工作區 (workspace)：git add 後檔案送到暫存區。
   (b) 暫存區 (staging area)：檔案處於 staged 狀態，git commit 後，檔案存入本地儲存庫。
   (c) 本地儲存庫 (local repository)：git push 後存入遠端儲存庫。
   (d) 遠端儲存庫 (remote repository)：github, gitlab 等網站。
6. Git 逆向流程：從遠端儲存庫取回最新資料到工作區
   (a) 遠端儲存庫 (remote repository)：github, gitlab 等網站，存有順向流程的檔案。
   (b) 工作區 (workspace)：git pull 後，工作區的檔案會被更新到與遠端儲存庫同步。
7. 新手也能懂的 Git 教學

2.2 Git 命令

1. host 已安裝 git 套件，提供 git 開頭的命令：

```
ceuser@deyu:~# git[TAB][TAB]
git                 git-shell           git-upload-pack
git-receive-pack    git-upload-archive
```

2. git 工具的命令：

```
ceuser@deyu:~# git [TAB][TAB]
add               describe          mergetool         revert
am                diff              mv                rm
apply             difftool          notes             scalar
archive           fetch             prune             send-email
bisect            format-patch      pull              shortlog
blame             fsck              push              show
branch            gc                range-diff        show-branch
bundle            gitk              rebase            sparse-checkout
checkout          grep              reflog            stage
cherry            gui               refs              stash
cherry-pick       help              remote            status
citool            init              repack            submodule
clean             instaweb          replace           switch
clone             log               request-pull      tag
commit            maintenance       reset             whatchanged
config            merge             restore           worktree
```

2.3 * 建立 Git Repository

1. 建立 git repository 資料儲存目錄

```
root@deyu:~# mkdir -p /srv/git
```

2. 進入 git repository 資料儲存目錄

```
root@deyu:~# cd /srv/git
```

3. 建立一個空的 git repository `ansible_pb.git`，一般副檔名設定爲.git。

```
root@deyu:/srv/git# git init --bare ansible_pb.git
hint: Using 'master' as the name for the initial branch. This default
    branch name
hint: is subject to change. To configure the initial branch name to use
    in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this
    command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /srv/git/ansible_pb.git/
```

4. git repository ansible_pb.git 目錄產生的資料。

```
root@deyu:/srv/git# ll ansible_pb.git/
total 16
drwxr-xr-x. 2 root root    6 Feb  3 07:38 branches
-rw-r--r--. 1 root root   66 Feb  3 07:38 config
-rw-r--r--. 1 root root   73 Feb  3 07:38 description
-rw-r--r--. 1 root root   23 Feb  3 07:38 HEAD
drwxr-xr-x. 2 root root 4096 Feb  3 07:38 hooks
drwxr-xr-x. 2 root root   21 Feb  3 07:38 info
drwxr-xr-x. 4 root root   30 Feb  3 07:38 objects
drwxr-xr-x. 4 root root   31 Feb  3 07:38 refs
```

5. Git 提供掛鉤 (hook) 事件觸發功能，在某些時間點執行客製化的工作，掛鉤檔都在 hooks 目錄下，範例檔都以.sample 爲副檔名，先使用範例 post-update.sample。

```
root@deyu:/srv/git# mv ansible_pb.git/hooks/post-update.sample
    ansible_pb.git/hooks/post-update
root@deyu:/srv/git# cat ansible_pb.git/hooks/post-update
#!/usr/bin/sh
#
# An example hook script to prepare a packed repository for use over
# dumb transports.
#
# To enable this hook, rename this file to "post-update".

exec git update-server-info
```

6. 變更掛鉤檔 post-updates 屬性爲可執行。

```
root@deyu:/srv/git# chmod +x ansible_pb.git/hooks/post-update
```

7. 爲允許用户經由 http push 到 git repository，必須在 config 配置檔中加入 http 章節，設定「receivepack = true」。

```
root@deyu:/srv/git# vi ansible_pb.git/config
[core]
    repositoryformatversion = 0
    filemode = true
    bare = true
[http]
    receivepack = true
```

2.4 *Http Git Server

1. Git server 支援 SSH 及 HTTP 兩種協定，本系統採用 http 協定存取 git repository，先安裝 httpd 套件。

```
root@deyu:/srv/git# dnf install httpd
```

2. vi 編輯 /etc/httpd/conf.d/git.conf，限制用户存取 ansible_pb.git。

```
root@deyu:/srv/git# vi /etc/httpd/conf.d/git.conf
<VirtualHost *:80>
    ServerName deyu.wang

    # Set environment variables for git-http-backend
    SetEnv GIT_PROJECT_ROOT /srv/git
    SetEnv GIT_HTTP_EXPORT_ALL
    ScriptAliasMatch "(?x)^/(.*/(HEAD | \
        info/refs | \
        objects/(info/[^/]+ | \
        [0-9a-f]{2}/[0-9a-f]{38} | \
        pack/pack-[0-9a-f]{40}\.(pack|idx)) | \
        git-(upload|receive)-pack))$" \
        "/usr/libexec/git-core/git-http-backend/$1"

    # Directory for the git-http-backend script
    <Directory "/usr/libexec/git-core">
        Options +ExecCGI
        Require all granted
    </Directory>
```

```
21
    <Location "^/ansible_pb.git">
23      AuthType Basic
        AuthName "Git Access"
25      AuthUserFile "/etc/httpd/conf.d/git.ceuser.htpasswd"
        Require valid-user
27  </Location>
</VirtualHost>
```

3. 使用 htpasswd 建立可存取 ansible_pb.git 的用户檔。

```
root@deyu:/srv/git# htpasswd -c /etc/httpd/conf.d/git.ceuser.htpasswd
    ceuser
2 New password:
Re-type new password:
4 Adding password for user ceuser
```

4. 設定 httpd 開機啓動且現在啓動。

```
root@deyu:/srv/git# systemctl enable --now httpd
```

2.5 *http 存取 Git Repository

1. 本系統架設的 git repository ansible_pb.git 爲 http 協定存取，所以其用户及群組變更爲 apache。

```
1 root@deyu:/srv/git# chown -R apache:apache ansible_pb.git
root@deyu:/srv/git# ll ansible_pb.git/
3 total 16
drwxr-xr-x. 2 apache apache    6 Feb  3 07:38 branches
5 -rw-r--r--. 1 apache apache   66 Feb  3 07:38 config
-rw-r--r--. 1 apache apache   73 Feb  3 07:38 description
7 -rw-r--r--. 1 apache apache   23 Feb  3 07:38 HEAD
drwxr-xr-x. 2 apache apache 4096 Feb  3 07:38 hooks
9 drwxr-xr-x. 2 apache apache   21 Feb  3 07:38 info
drwxr-xr-x. 4 apache apache   30 Feb  3 07:38 objects
11 drwxr-xr-x. 4 apache apache   31 Feb  3 07:38 refs
```

2. 爲確保 SELinux enforcing 情況下，可以存取 git repository ansible_pb.git，所以其用户及群組變更爲 apache。

```
root@deyu:/srv/git# semanage fcontext -a -t httpd_sys_rw_content_t "/srv
    /git/ansible_pb.git(/.*)?"
root@deyu:/srv/git# restorecon -rv /srv/git/ansible_pb.git
Relabeled /srv/git/ansible_pb.git from unconfined_u:object_r:var_t:s0
to unconfined_u:object_r:httpd_sys_rw_content_t:s0
Relabeled /srv/git/ansible_pb.git/branches from unconfined_u:object_r:
    var_t:s0
to unconfined_u:object_r:httpd_sys_rw_content_t:s0
.....以下省略.............
```

3. 查看 git repository ansible_pb.git 的 SELinux fcontext type，已變更爲 httpd_sys_rw_content_t。

```
root@deyu:/srv/git# ls -Z ansible_pb.git/
unconfined_u:object_r:httpd_sys_rw_content_t:s0 branches
unconfined_u:object_r:httpd_sys_rw_content_t:s0 config
unconfined_u:object_r:httpd_sys_rw_content_t:s0 description
unconfined_u:object_r:httpd_sys_rw_content_t:s0 HEAD
unconfined_u:object_r:httpd_sys_rw_content_t:s0 hooks
unconfined_u:object_r:httpd_sys_rw_content_t:s0 info
unconfined_u:object_r:httpd_sys_rw_content_t:s0 objects
unconfined_u:object_r:httpd_sys_rw_content_t:s0 refs
```

4. 重新啓動 httpd 服務。

```
root@deyu:/srv/git# systemctl restart httpd
```

2.6 Git Clone

1. 用户 ceuser 在其家目錄，以帳號 ceuser 密碼 123qwe，從遠端伺服器 deyu.wang，clone ansible_pb 遠端儲存庫 ansible_pb，clone 的 repository 是空的。

```
ceuser@deyu:~$ git clone http://ceuser:123qwe@deyu.wang/ansible_pb.git
Cloning into 'ansible_pb'...
warning: You appear to have cloned an empty repository.
```

2. 沒有設定目錄，clone 建立的目錄名稱爲遠端儲存庫名稱 ansible_pb。

```
ceuser@deyu:~$ ll
total 4
drwxr-xr-x. 3 ceuser ceuser 18 Feb 14 17:43 ansible_pb
-rw-r--r--. 1 ceuser ceuser 33 Feb  9 11:25 sid
```

3. clone 遠端儲存庫 ansible_pb，並設定本地端的目錄爲學號。

```
ceuser@deyu:~$ git clone http://ceuser:123qwe@deyu.wang/ansible_pb.git
    10927000
Cloning into '10927000'...
warning: You appear to have cloned an empty repository.
```

4. 設定本地端目錄爲 10927000，clone 建立的目錄名稱爲 10927000。

```
ceuser@deyu:~$ ll
total 4
drwxr-xr-x. 3 ceuser ceuser 18 Feb 14 17:53 10927000
drwxr-xr-x. 3 ceuser ceuser 18 Feb 14 17:43 ansible_pb
-rw-r--r--. 1 ceuser ceuser 33 Feb  9 11:25 sid
```

2.7 Git Config

1. 進入 ansible_pb 目錄。

```
ceuser@deyu:~$ cd ansible_pb
```

2. 使用 ansible_pb repository 前最重要的是設定用户 email 及 name。

```
ceuser@deyu:~$ cd ansible_pb
ceuser@deyu:~/ansible_pb$ git config user.email 'ceuser@deyu.wang'
ceuser@deyu:~/ansible_pb$ git config user.name 'ceuser'
```

2.8 Git Push

1. 建立一個測試檔案 test.txt。

```
ceuser@deyu:~/ansible_pb$ echo 'git test' > test.txt
```

2. git add 將 test.txt 送到暫存區。

```
ceuser@deyu:~/ansible_pb$ git add test.txt
```

3. git commit 將 test.txt 存入本地 repository。

```
ceuser@deyu:~/ansible_pb$ git commit -m "test.txt"
[master (root-commit) f99a587] test.txt
 1 file changed, 1 insertion(+)
 create mode 100644 test.txt
```

4. git push 將本地 repository 存入遠端的 git repository。

```
ceuser@deyu:~/ansible_pb$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 211 bytes | 211.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To http://deyu.wang/ansible_pb.git
 * [new branch]      master -> master
```

2.9 實機練習

1. 練習系統的 GIT Server 架設在 `http://deyu.wang/ansible_pb.git`，用户 ceuser，密碼 123qwe。
2. 以 ceuser 登入 host 主機。
3. 在 ceuser 家目錄 git clone 遠端 deyu.wang 的 `ansible_pb` repository，並設定本地端的目錄爲 `$mysid`，變數 `$mysid` 爲學號不加 s。
4. git config 設定 email 爲 `$mysid@deyu.wang` 及 name 爲 `$mysid`。
5. 編輯一個測試檔案 test.txt，內容爲 `$mysid git test`。
6. git add 將 test.txt 送到本地暫存區。

7. git commit 將 test.txt 存入本地 git repository。
8. git push 本地 git repository 同步到遠端 git repository。

Chapter 3

自動部署環境

3.1 Ansible 簡介

1. Ansible 是一種開源配置工具，Linux 系統管理員使用的自動化工具。
2. Ansible 能讓系統管理員從一個控制節點（即 Ansible 服務器）管理數百台服務器。
3. Ansible 不需要任何代理，並且可以在 SSH 和 python 上工作。
4. 觀念說明：
   (a) 控制節點 (Control node)：安裝 Ansible 的主機，可以運行 ansible 相關命令。
   (b) 受管節點 (Managed nodes)：被 Ansible 管理的主機。
   (c) Inventory：受管節點的列表，inventory 檔案有時也稱爲 hostfile。
   (d) 內容集 (Collections)：Ansible 內容的分發 (distribution) 格式，內容包括特定主題的 playbooks, roles, modules 及 plugins，可以經由 distribution servers 來安裝使用集合，例如：Ansible Galaxy。
   (e) 模組 (Modules)：Ansible 執行單元，種類有套件 (Packaging)、資料庫 (Database)、通知 (Notification)、雲端 (Cloud)、Windows 到網路 (Network) 等。要是找不到想要的功能，還可以透過 command 和 shell 等 Modules 自行擴充。
   (f) 任務 (Tasks)：Ansible 中的工作單位，可以使用 ad-hoc 命令一次執行一個任務。
   (g) Playbooks：Ansible 配置、部署等任務的腳本，腳本描述希望遠端主機執行命令的方案，或者一組 IT 程式執行的命令集合，Playbooks 可以包含變數。

3.2 環境介紹

1. 實作環境：AlmaLinux 10
2. 練習系統：
   (a) 控制節點 (Control node) ：deyu.wang 192.168.122.1 的帳號 ceuser
   (b) 節點一 (Managed node 1) ：kvm11.deyu.wang 192.168.122.11
   (c) 節點二 (Managed node 2) ：kvm12.deyu.wang 192.168.122.12
   (d) 節點三 (Managed node 3) ：kvm13.deyu.wang 192.168.122.13
   (e) 節點四 (Managed node 4) ：kvm14.deyu.wang 192.168.122.14
   (f) 節點五 (Managed node 5) ：kvm15.deyu.wang 192.168.122.15
3. 所有系統的密碼都是 123qwe，不要變更密碼。
4. 控制節點在 host (deyu.wang 192.168.122.1)，使用一般帳號 ceuser 操作。
5. 所有受控節點都已新增帳號 ceuser，且控制節點的 ceuser 都可免密碼 ssh ceuser@kvm1X 登入。不要變更任何 SSH 設定。
6. GIT 倉庫 (git repositories) 架設在 http:// ceuser@deyu.wang/ ansible_pb.git，所有 ansible 設定檔、playbook yaml 腳本、內容集、角色，都要同步到 GIT 倉庫
7. 若需要安裝套件，套件庫位於 dywang.csie.cyut.edu.tw：http:// dywang.csie.cyut.edu.tw/ alma10/ BaseOS http:// dywang.csie.cyut.edu.tw/alma10/AppStream
8. 實作需要的附加檔案：http://dywang.csie.cyut.edu.tw/materials

3.3 Ansible 命令

1. host 已安裝 ansible-core 套件，提供 ansible 相關指令，ansible 開頭的命令：

```
ceuser@deyu:~# ansible[TAB][TAB]
ansible              ansible-doc          ansible-pull
ansible-config       ansible-galaxy       ansible-vault
ansible-connection   ansible-inventory
ansible-console      ansible-playbook
```

2. 查看 ansible 版本 2.16.14，python 版本 3.12.9。

```
ceuser@deyu:~# ansible --version
ansible [core 2.16.14]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ceuser/.ansible/plugins/
      modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.12/site-packages/
      ansible
  ansible collection location = /home/ceuser/.ansible/collections:/usr/
      share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.9 (main, Aug 14 2025, 00:00:00) [GCC 14.2.1
      20250110 (Red Hat 14.2.1-7)] (/usr/bin/python3)
  jinja version = 3.1.6
  libyaml = True
```

3.4 * 設定 SSH Keys

1. Ansible 控制節點與受管節點間藉由 ssh 連線，必須設定免密碼連線，或執行 ansible 時使用 –ask-pass 選項要求輸入密碼。爲方便自動完成任務一般會使用免密碼連線。

2. 產生一組金鑰

```
[ceuser@deyu ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:F9OoHjIT3jPlN95CnwEAKnrIQsySYaZMQOZU6wVpwsA ceuser@deyu.deyu.wang
The key's randomart image is:
+---[RSA 2048]----+
|XB.o.    ...     |
|%E oo   .   +     |
|+*o...o   = o     |
|o..o.o o + o .    |
|. +.. = S o + .   |
| . .   = = + + o  |
|        .   o +   |
|             .    |
|                  |
+----[SHA256]-----+
```

3. 查看.ssh 目錄下新增了 id_rsa 私鑰及 id_rsa.pub 公鑰。

```
[ceuser@deyu ~]# ll .ssh/
total 12
-rw-r--r--. 1 root root  401 Mar  2 10:32 authorized_keys
-rw-------. 1 root root 1831 Mar  5 10:59 id_rsa
-rw-r--r--. 1 root root  402 Mar  5 10:59 id_rsa.pub
```

4. ssh-copy-id 將 id_rsa.pub 公鑰匯入到 kvm11，輸入密碼完成匯入。

```
[ceuser@deyu ~]# ssh-copy-id kvm11.deyu.wang
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
 "/root/.ssh/id_rsa.pub"
The authenticity of host 'kvm11.deyu.wang (192.168.122.11)' can't be
    established.
ECDSA key fingerprint is SHA256:l6jIMoK+
    z82ydTlG8MKarNcUDLVSfjCRjWRzhGOtDNc.
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 prompted now it is to install the new keys
root@kvm11.deyu.wang's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'kvm11.deyu.wang'"
and check to make sure that only the key(s) you wanted were added.
```

5. ssh 試著連線 kvm11，不用輸入密碼即可登入。

```
[ceuser@deyu ~]# ssh kvm11.deyu.wang
Last login: Mon Mar  2 18:40:01 2020 from 192.168.122.1
```

6. 退出 kvm11。

```
[root@kvm11 ~]# exit
logout
Connection to kvm11.deyu.wang closed.
```

3.5 * 設定 SUDOER

1. 控制節點以一般用户 ceuser 執行 ansible 時，受管節點也是以用户 ceuser 權限執行，必須以 root 權限才能執行的命令要用 sudo 執行。

2. 本練習系統 ansible 控制節點及受管節點都已建立 ceuser 帳號，將此帳號設定成免密碼可以使用 sudo 執行所有命令。

```
[ceuser@deyu ~]# vim /etc/sudoers.d/ansible
[ceuser@deyu ~]# cat /etc/sudoers.d/ansible
ceuser ALL=(ALL) NOPASSWD: ALL
```

3. 其他受管節點也做相同的 sudoer 設定。

3.6 實機練習

1. 主機說明：

   (a) 控制節點 (Control node) ：deyu.wang 192.168.122.1 的帳號 ceuser

   (b) 節點一 (Managed node 1) ：kvm11.deyu.wang 192.168.122.11

   (c) 節點二 (Managed node 2) ：kvm12.deyu.wang 192.168.122.12

   (d) 節點三 (Managed node 3) ：kvm13.deyu.wang 192.168.122.13

   (e) 節點四 (Managed node 4) ：kvm14.deyu.wang 192.168.122.14

   (f) 節點五 (Managed node 5) ：kvm15.deyu.wang 192.168.122.15

2. 在控制主機 deyu.wang 以 ceuser 帳號，完成以下工作：

   (a) 重新安裝 (rebuild) 所有受控節點。

   (b) 查詢 ansible 版本，導向到 /ansibleversion.txt

3. * 使用同一組 ssh keys，讓控制節點與受管節點間，用户 root 彼此都可免密碼登入，用户 ceuser 彼此也都可免密碼登入。

4. * 設定控制主機及所有節點，都可以免密碼執行 sudo。

Chapter 4

Inventory 檔案

4.1 前言

1. Ansible 可透過 inventory 檔案配置，同時操作屬於一個組的多台主機。
2. Inventory 檔案預設爲 /etc/ansible/hosts。
3. 也可以下命令時使用 -i <path> 指定 inventory 檔案，而不使用預設檔。
4. 還可以同時使用多個 inventory 檔案，也可以從動態源或雲端取得 inventory 配置資訊。

4.2 基本設定

1. ansible-core 套件中 /etc/ansible/hosts 是一個 inventory 範例檔。
   範例說明：

```
ceuser@deyu:$ cat /etc/ansible/hosts
# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
#   - Comments begin with the '#' character  # 井號開頭爲註解
#   - Blank lines are ignored  # 空白行忽略
#   - Groups of hosts are delimited by [header] elements # 群組以 [...] 區隔
#   - You can enter hostnames or ip addresses  # 可以使用主機名稱或IP位址
#   - A hostname/ip can be a member of multiple groups # 主機名稱/ip 可以是多個
```

2. 範例一：不屬於群組的主機。

```
# Ex 1: Ungrouped hosts, specify before any group headers:

## green.example.com
## blue.example.com
## 192.168.100.1
## 192.168.100.10
```

3. 範例二：屬於 webservers 群組的主機。

```
# Ex 2: A collection of hosts belonging to the 'webservers' group:

## [webservers]
## alpha.example.org
## beta.example.org
## 192.168.1.100
## 192.168.1.110
```

4. 使用 pattern 指定多台主機，以下範例是 www1.example.com 至 www6.example.com 6 台主機，db-99-node.example.com 至 db-101-node.example.com 三台主機。

```
# If you have multiple hosts following a pattern, you can specify
# them like this:

## www[001:006].example.com

# You can also use ranges for multiple hosts:

## db-[99:101]-node.example.com
```

5. 範例三：屬於 dbservers 群組的主機。

```
# Ex 3: A collection of database servers in the 'dbservers' group:

## [dbservers]
##
## db01.intranet.mydomain.net
## db02.intranet.mydomain.net
## 10.25.1.56
## 10.25.1.57
```

6. 多台主機被分組，例如：“Debian” 和 “openSUSE”。

```
# Ex4: Multiple hosts arranged into groups such as 'Debian' and '
    openSUSE':

## [Debian]
## alpha.example.org
## beta.example.org

## [openSUSE]
## green.example.com
## blue.example.com
```

4.3 變數說明

1. 連線的遠端主機名稱，例如前一節設定別名時使用。

```
ansible_ssh_host
```

2. ssh 埠號，如果不是預設的埠號，可透過此變數設定。

```
ansible_ssh_port
```

3. 預設的 ssh 使用者名稱

```
ansible_ssh_user
```

4. ssh 密碼，使用變數設定 ssh 密碼不安全，建議使用 –ask-pass 或 SSH 金鑰。

```
ansible_ssh_pass
```

5. sudo 密碼，使用變數設定 sudo 密碼不安全，建議使用 –ask-sudo-pass。

```
ansible_sudo_pass
```

6. sudo 命令路徑

```
ansible_sudo_exe
```

7. 與主機的連線類型，例如:local, ssh 或者 paramiko。

```
ansible_connection
```

8. ssh 使用的私鑰檔案

```
ansible_ssh_private_key_file
```

9. 目標系統的 shell 類型，預設使用 sh 語法

```
ansible_shell_type
```

4.4 進階設定

1. 埠號：如果有主機的 SSH 不是標準的 22 port，可在主機名之後加上埠號，用冒號分隔。

```
www1.example.com:2299
```

2. 別名：node1 會連線 192.168.1.11:2299

```
node1 ansible_ssh_port=2299 ansible_ssh_host=192.168.1.11
```

3. 組的子成員：以下範例 webservers, dbservers 群組都是 servers 群組的子成員。

```
[webservers]
alpha.example.org
beta.example.org
192.168.1.100
192.168.1.110
```

```
7  [dbservers]
   db01.intranet.mydomain.net
9  db02.intranet.mydomain.net
   10.25.1.56
11 10.25.1.57

13 [servers:children]
   webservers
15 dbservers
```

4.5 設定 inventory

1. 控制節點主機 deyu.wang 已建立用户 ceuser，此用户也使用 git clone 同步工作目錄 ansible_pb。

2. ceuser 登入 deyu.wang，切換到工作目錄 ansible_pb。

```
1  ceuser@deyu:~$ cd ansible_pb
   ceuser@deyu:~/ansible_pb$
```

3. 以如下要求建立 inventory 檔：

   (a) node1 是 test 主機群組的成員

   (b) node2 是 develop 主機群組的成員

   (c) node3 及 node4 是 product 主機群組的成員

   (d) node5 是 balancer 主機群組的成員

   (e) product 群組是 web 主機群組的成員

4. 編輯 inventory 檔案：

```
   ceuser@deyu:~/ansible_pb$ vim inventory
2  [test]
   node1
4
   [develop]
6  node2

8  [product]
   node3
10 node4

12 [balancer]
```

```
node5

[web:children]
product
```

4.6 檢查 inventory

1. 使用 inventory 列出群組 product 群組主機：

```
ceuser@deyu:~/ansible_pb$ ansible product -i inventory --list-hosts
  hosts (2):
    node3
    node4
```

2. 使用 inventory 列出所有受管主機，pattern 'all' 也可用'*' 代表全部：

```
ceuser@deyu:~/ansible_pb$ ansible all -i inventory --list-hosts
  hosts (5):
    node1
    node2
    node5
    node3
    node4
```

3. 不指定 inventory 下，使用預設檔 /etc/ansible/hosts 受管主機是空的。

```
ceuser@deyu:~/ansible_pb$ ansible all --list-hosts
[WARNING]: provided hosts list is empty, only localhost is available.
   Note that
the implicit localhost does not match 'all'
  hosts (0):
```

4.7 實機練習

1. 控制節點主機 deyu.wang 已建立用户 ceuser，此用户也使用 git clone 同步工作目錄 ansible_pb 了，若未同步，則先用 git clone 同步。

2. ceuser 登入 deyu.wang，切換到工作目錄 ansible_pb。

3. 在 ansible_pb 工作目錄下產生 inventory 檔，要求如下：
   (a) kvm11 是 dev 主機群組的成員
   (b) kvm12 是 test 主機群組的成員
   (c) kvm13 及 kvm14 是 prod 主機群組的成員
   (d) kvm15 是 balancers 主機群組的成員
   (e) prod 群組是 webservers 主機群組的成員
4. 使用 inventory 檔列出所有群組主機。
5. 使用 inventory 檔列出群組 webservers 主機。

Chapter 5

組態設定

5.1 前言

1. Ansible 支援多種來源設定組態 (Configuration)，包含名爲 ansible.cfg 的 ini 、環境變數、命令選項、playbook 關鍵字、變數等。
2. ansible.cfg 讓使用者看到所有可用的組態設定、其預設值、如何設定等。
3. 因有多種組態設定來源，可能不一致，Ansible 以下列順序搜尋，先找到後其他就忽略。
   (a) 環境變數 ANSIBLE_CONFIG
   (b) 現在工作目錄下的 ansible.cfg
   (c) 使用家目錄下的 ~/.ansible.cfg
   (d) 系統 /etc 目錄下的 /etc/ansible/ansible.cfg

5.2 ansible.cfg 範例

1. 複製 /etc/ansible/ansible.cfg 到工作目錄，查看其內容：

```
ceuser@deyu:~/ansible_pb$ cat /etc/ansible/ansible.cfg
# Since Ansible 2.12 (core):
# To generate an example config file (a "disabled" one with all default settings, commented out):
#               $ ansible-config init --disabled > ansible.cfg
#
# Also you can now have a more complete file by including existing plugins:
# ansible-config init --disabled -t all > ansible.cfg

```

```
# For previous versions of Ansible you can check for examples in the '
    stable' branches of each version
# Note that this file was always incomplete  and lagging changes to
    configuration settings

# for example, for 2.9: https://github.com/ansible/ansible/blob/stable
    -2.9/examples/ansible.cfg
```

2. 依據 /etc/ansible/ansible.cfg 內容，執行 ansible-config init 產生 ansible.cfg。

```
ceuser@deyu:~/ansible_pb$ ansible-config init --disabled > ansible.cfg
```

3. 產生的 ansible.cfg [...] 爲章節，目前只需設定 [default] 章節的變數，# 開頭爲說明，; 開頭爲變數設定預設並註解。

```
ceuser@deyu:~/ansible_pb$ grep -Ev '^(#|$|;)' ansible.cfg
[defaults]
[privilege_escalation]
[persistent_connection]
[connection]
[colors]
[selinux]
[diff]
[galaxy]
[inventory]
[netconf_connection]
[paramiko_connection]
[jinja2]
[tags]
```

5.3 練習系統 ansible.cfg

1. 控制節點主機 deyu.wang 已建立用户 ceuser，此用户已 git clone 同步 ansible 工作目錄 ansible_pb。

2. ceuser 登入 deyu.wang，切換到工作目錄 ansible_pb，新增 ansible.cfg，要求如下：

   (a) 指定 inventory 檔案爲 /home/ceuser/ansible_pb/inventory

   (b) 使用於 playbooks 的角色，其位置包含 /home/ceuser/ansible_pb/roles

(c) 內容集路徑包含 /home/ceuser/ansible_pb/collections

3. 在 ansible_pb 目錄編輯檔案 ansible.cfg

```
ceuser@deyu:~/ansible_pb$ vi ansible.cfg
```

4. ansible.cfg 檔案依要求設定如下，host_key_checking=False 不使用金鑰檢查，remote_user=root 登入的用户爲 root：

```
ceuser@deyu:~/ansible_pb$ vi ansible.cfg
[defaults]
inventory      = /home/ceuser/ansible_pb/inventory
roles_path     = /home/ceuser/ansible_pb/roles:/usr/share/ansible/roles
collections_path = /home/ceuser/ansible_pb/collections
host_key_checking = False
remote_user = root
```

5. 新增 /home/ceuser/ansible_pb/roles 目錄。

```
ceuser@deyu:~/ansible_pb$ mkdir roles
```

5.4 測試 ansible.cfg

1. 查看 ansible 的工作目錄下有 ansible.cfg 設定檔及 inventory 檔。

```
ceuser@deyu:~/ansible_pb$ ll
total 12
-rw-r--r--. 1 ceuser ceuser 233 Feb  3 19:48 ansible.cfg
-rw-r--r--. 1 ceuser ceuser 168 Feb  3 18:49 inventory
-rw-r--r--. 1 ceuser ceuser   9 Feb  3 08:54 test.txt
```

2. 執行 ansible 列出 webservers 群組的主機，此時會使用工作目錄下的 ansible.cfg 設定檔及 inventory 檔。

```
ceuser@deyu:~/ansible_pb$ ansible web --list-host
  hosts (2):
    node3
    node4
```

5.5 實機練習

1. 以用户 ceuser 登入控制節點 deyu.wang，在 ansible 工作目錄下新增 ansible.cfg，要求如下：
   (a) 指定 inventory 檔案爲 /home/ceuser/ansible_pb/inventory
   (b) 使用於 playbooks 的角色，其位置包含 /home/ceuser/ansible_pb/roles
   (c) 內容集路徑包含 /home/ceuser/ansible_pb/collections
   (d) 遠端控制節點用户爲 root。
   (e) ssh 登入受管節點，不檢查金鑰。
2. ansible 命令不下 -i 指定 inventory 檔案，列出所有群組主機。

Chapter 6

Ad-hoc 命令

6.1 前言

1. Linux 的文字界面可以輸入簡單命令來完成工作，但對於比較複雜或經常重複的工作，一般會寫成 shell scripts 來完成。
2. Ansible 提供兩種方式在受管節點自動完成任務，一是臨時命令 (ad-hoc command)，另一是寫 ansible playbook。
3. Ansible 的 ad-hoc 命令如同 linux 文字界面命令，playbooks 則類似 shell scripts。
4. ad-hoc 命令適合很少重複執行的任務，重複複雜的任務則由 playbook 來完成。

6.2 模組文件

1. Ansible-doc 可查詢各模組的文件，-h 選項查輔助說明。

```
ceuser@deyu:~/ansible_pb$ ansible-doc -h
usage: ansible-doc [-h] [--version] [-v] [-M MODULE_PATH]
                   [--playbook-dir BASEDIR]
..............省略..............

options:
  --metadata-dump       **For internal use only** Dump json metadata for
      all
                        entries, ignores other options.
..............省略..............
  --version             show program's version number, config file
      location,
                        configured module search path, module location,
                        executable location and exit
```

```
..............省略..............
  -h, --help            show this help message and exit
  -j, --json            Change output into json format.
  -l, --list            List available plugins. A supplied argument will
      be
                        used for filtering, can be a namespace or full
                        collection name.
..............省略..............
See man pages for Ansible CLI options or website for tutorials
https://docs.ansible.com
```

2. ansible-doc 列出可用的模組

```
ceuser@deyu:~/ansible_pb$ ansible-doc -l
```

3. ansible-doc 列出可用的模組，過濾 yum，找到有 yum 及 yum_repository 兩個模組。

```
ceuser@deyu:~/ansible_pb$ ansible-doc -l | grep yum
ansible.builtin.yum                     Manages packages with the `yum'
    pack...
ansible.builtin.yum_repository          Add or remove YUM repositories
```

6.3 檔案傳輸 - copy 模組

1. Ansible 可以用並行的方式同時 scp 大量的檔案到多台主機，完整文件可用 ansible-doc 查詢 copy 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc copy | grep -A8 ^EX
EXAMPLES:

- name: Copy file with owner and permissions
  ansible.builtin.copy:
    src: /srv/myfiles/foo.conf
    dest: /etc/foo.conf
    owner: foo
    group: foo
    mode: '0644'
```

2. 先編輯一個測試檔 adhoc.txt

```
ceuser@deyu:~/ansible_pb$ echo 'ad-hoc file transfer' > adhoc.txt
ceuser@deyu:~/ansible_pb$ cat adhoc.txt
ad-hoc file transfer
```

3. 使用 copy 模組將 adhoc.txt 複製到 dev 群組裡所有主機的 /home/ceuser 目錄:

```
ceuser@deyu:~/ansible_pb$ ansible dev -m copy -a "src=adhoc.txt dest=/
    home/ceuser/"
kvm11 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "checksum": "5ced2367ff60e242577e9a5d90a61233d9336c97",
    "dest": "/home/ceuser/adhoc.txt",
    "gid": 0,
    "group": "root",
    "md5sum": "bb660c5cbe1c8defa638bd8e7592b48a",
    "mode": "0644",
    "owner": "root",
    "secontext": "unconfined_u:object_r:user_home_t:s0",
    "size": 21,
    "src": "/root/.ansible/tmp/ansible-tmp
        -1770260586.953098-81322-277034741284408/source",
    "state": "file",
    "uid": 0
}
```

4. 使用 shell 模組查看 dev 群組裡所有主機，有複製過來的檔案 adhoc.txt，由於 ansible.cfg 設定遠端用户爲 root，所以擁有者及群組皆爲 root。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "ls -l /home/ceuser/"
kvm11 | CHANGED | rc=0 >>
total 4
-rw-r--r--. 1 root root 21 Feb  5 11:03 adhoc.txt
```

6.4 檔案管理 - file 模組

1. file 模組可以做檔案管理，完整文件可用 ansible-doc 查詢 file 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc file | grep -A88 ^EX
```

```
EXAMPLES:

- name: Change file ownership, group and permissions
  ansible.builtin.file:
    path: /etc/foo.conf
    owner: foo      # 檔案擁有者
    group: foo      # 檔案所屬群組
    mode: '0644'    # 檔案屬性

- name: Give insecure permissions to an existing file
  ansible.builtin.file:
    path: /work
    owner: root
    group: root
    mode: '1777'

- name: Create a symbolic link
  ansible.builtin.file:
    src: /file/to/link/to
    dest: /path/to/symlink
    owner: foo
    group: foo
    state: link     # 建立軟連結檔

- name: Create two hard links
  ansible.builtin.file:
    src: '/tmp/{{ item.src }}'
    dest: '{{ item.dest }}'
    state: hard
  loop:
    - { src: x, dest: y }
    - { src: z, dest: k }

- name: Touch a file, using symbolic modes to set the permissions (
    equivalent to 0644)
  ansible.builtin.file:
    path: /etc/foo.conf
    state: touch
    mode: u=rw,g=r,o=r

- name: Touch the same file, but add/remove some permissions
  ansible.builtin.file:
    path: /etc/foo.conf
    state: touch
    mode: u+rw,g-wx,o-rwx

- name: Touch again the same file, but do not change times this makes
    the task idempotent
  ansible.builtin.file:
    path: /etc/foo.conf
    state: touch
    mode: u+rw,g-wx,o-rwx
    modification_time: preserve
    access_time: preserve
```

```
54
- name: Create a directory if it does not exist
56   ansible.builtin.file:
       path: /etc/some_directory
58     state: directory   # 建立目錄
       mode: '0755'
60
- name: Update modification and access time of given file
62   ansible.builtin.file:
       path: /etc/some_file
64     state: file
       modification_time: now
66     access_time: now

68 - name: Set access time based on seconds from epoch value
     ansible.builtin.file:
70     path: /etc/another_file
       state: file
72     access_time: '{{ "%Y%m%d%H%M.%S" | strftime(stat_var.stat.atime) }}'

74 - name: Recursively change ownership of a directory
     ansible.builtin.file:
76     path: /etc/foo
       state: directory
78     recurse: yes
       owner: foo
80     group: foo

82 - name: Remove file (delete file)
     ansible.builtin.file:
84     path: /etc/foo.txt
       state: absent  # 刪除檔案
86
- name: Recursively remove directory
88   ansible.builtin.file:
       path: /etc/foo
90     state: absent  # 刪除目錄
```

2. 使用 file 模組修改檔案 /home/ceuser/adhoc.txt 的屬性成 660，擁有者及群組皆爲 ceuser。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m file -a "path=/home/ceuser/
    adhoc.txt \
2 > mode=660 owner=ceuser group=ceuser"
kvm11 | CHANGED => {
4     "ansible_facts": {
          "discovered_interpreter_python": "/usr/bin/python3"
6     },
      "changed": true,
8     "gid": 1000,
      "group": "ceuser",
```

```
10      "mode": "0660",
        "owner": "ceuser",
12      "path": "/home/ceuser/adhoc.txt",
        "secontext": "unconfined_u:object_r:user_home_t:s0",
14      "size": 21,
        "state": "file",
16      "uid": 1000
}
```

3. 使用 shell 模組查看 dev 群組裡所有主機，adhoc.txt 的屬性已改成 660，擁有者及群組也都改成 ceuser 了。

```
1  ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "ls -l /home/ceuser/
       adhoc.txt"
   kvm11 | CHANGED | rc=0 >>
3  -rw-rw----. 1 ceuser ceuser 21 Feb  5 11:03 /home/ceuser/adhoc.txt
```

6.5 實機練習

1. 使用 copy, shell, file 模組，完成以下要求：

   (a) 複製控制主機的 adhoc.txt 到節點主機 test 的 /home/ceuser 目錄。

   (b) adhoc.txt 內容如講義”ad-hoc file transfer”。

   (c) 查看複製到節點主機 test 的 adhoc.txt，其擁有者、所屬群組及檔案權限，爲什麼？

   (d) 將複製到節點主機 test 的 adhoc.txt 檔案權限改成 660，擁有者及群組都改成 ceuser。

Chapter 7

Playbooks

7.1 前言

1. 相較於 ad-hoc 命令，Playbooks 是比較完整且功能強的 ansible 配置方法。
2. Ansible 的腳本 (playbook) 是用 YAML (YAML Ain't Markup Language) 語言撰寫透過編寫腳本告訴 Ansible 需要做的事項。
3. YAML (Yet Another Markup Language)，原意是另一種標記語言。
4. 爲了強調 YAML 語言以資料爲中心，而不是以標記語言爲重點，而用反向縮寫重新命名。YAML (YAML Ain't a Markup Language)，不是一種標記語言。
5. YAML 以空白符號 (不能用 TAB) 縮排來編輯資料，適合表達階層式資料結構。
6. YAML 符號

```
1 ---       分隔檔案
  -         減號+空白：清單（陣列）
3 #         註解
```

7. YAML 具有高度可讀性特性，不會像是程式語言或指令碼，所以一份好的 Ansible playbook 是容易理解及修改維護。

7.2 基本語法

1. 看到 playbook 腳本中關鍵字 (Keywords) 不清楚時，可上 ansible 官網查詢。
2. 最基本的 playbook 分成三個部分：

(a) 在哪些主機以什麼身份執行，例如：

```
hosts: dev
user: root
```

(b) 執行的任務，notify 當任務執行通知"restart httpd"，例如：

```
tasks:
  - name: install httpd
    yum: pkg=httpd state=latest
    notify: restart httpd
```

(c) 善後任務，例如：

```
handlers:
  - name: restart httpd
    service: name=httpd state=restarted
```

7.3 yum_repository 模組

1. 任務 (tasks) 及善後處理 (handlers) 內要處理的工作，都可以參考 ansible-doc module-name 查詢模組文件範例來寫，例如 yum_repository。

```
ceuser@deyu:~/ansible_pb$ ansible-doc yum_repository | grep ^EX -A14
EXAMPLES:

- name: Add repository
  ansible.builtin.yum_repository:
    name: epel
    description: EPEL YUM repo
    baseurl: https://download.fedoraproject.org/pub/epel/$releasever/
        $basearch/

- name: Add multiple repositories into the same file (1/2)
  ansible.builtin.yum_repository:
    name: epel
    description: EPEL YUM repo
    file: external_repos
    baseurl: https://download.fedoraproject.org/pub/epel/$releasever/
        $basearch/
    gpgcheck: no
```

7.4 第一支 playbook

1. yum_repository 模組可以增加或刪除套件庫，第一支 playbook 寫一支 yum_repo.yml 腳本，只有一個任務，設定主機 dev 的套件庫，沒有善後處理工作，yum repository 要求如下：
2. Repository 1
   (a) 名稱 (name)：DYW_BASE
   (b) 描述 (description)：DYW base software
   (c) 套件來源網址：http://dywang.csie.cyut.edu.tw/alma10/BaseOS
   (d) 啓用 GPG 簽章檢查
   (e) GPG 金鑰網址：http://dywang.csie.cyut.edu.tw/materials/RPM-GPG-KEY-AlmaLinux-10
3. Repository 2
   (a) 名稱 (name)：DYW_STREAM
   (b) 描述 (description)：DYW stream software
   (c) 套件來源網址：http://dywang.csie.cyut.edu.tw/alma10/AppStream
   (d) 啓用 GPG 簽章檢查
   (e) GPG 金鑰網址：http://dywang.csie.cyut.edu.tw/materials/RPM-GPG-KEY-AlmaLinux-10
4. 編輯 yum_repo.yml。

```
ceuser@deyu:~/ansible_pb$ vi yum_repo.yml
---
- hosts: dev
  tasks:
    - name: Add BaseOS repository
      ansible.builtin.yum_repository:
        name: DYW_BASE
        description: DYW base software
        baseurl: http://dywang.csie.cyut.edu.tw/alma10/BaseOS
        gpgcheck: yes
        gpgkey: http://dywang.csie.cyut.edu.tw/materials/RPM-GPG-KEY-
            AlmaLinux-10

    - name: Add Stream repository
      ansible.builtin.yum_repository:
        name: DYW_STREAM
        description: DYW stream software
        baseurl: http://dywang.csie.cyut.edu.tw/alma10/AppStream
```

```
        gpgcheck: yes
        gpgkey: http://dywang.csie.cyut.edu.tw/materials/RPM-GPG-KEY-
            AlmaLinux-10
```

5. ansible-playbook 執行 yum_repo.yml，結果成功，kvm11 被更動。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook yum_repo.yml
ceuser@deyu:~/ansible_pb$ ansible-playbook yum_repo.yml

PLAY [dev]
    ***********************************************************************


TASK [Gathering Facts]
    ********************************************************
ok: [kvm11]

TASK [Add BaseOS repository]
    ***************************************************
changed: [kvm11]

TASK [Add Stream repository]
    ***************************************************
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm11 : ok=3  changed=2  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

6. 查詢主機 dev 的 /etc/yum.repos.d 目錄，產生了兩個.repo 檔，檔名是 yum_repo.yml 設定的名稱 name。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "ls -l /etc/yum.repos.
    d/"
kvm11 | CHANGED | rc=0 >>
total 8
-rw-r--r--. 1 root root 180 Feb  4 15:10 DYW_BASE.repo
-rw-r--r--. 1 root root 187 Feb  4 15:10 DYW_STREAM.repo
```

7. 查詢主機 dev 已設定好 repositories，其中 repo id 及 repo name 是 yum_repo.yml 設定的名稱 name 及描述 description。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "dnf repolist"
```

```
kvm11 | CHANGED | rc=0 >>
repo id                                 repo name
DYW_BASE                                DYW base software
DYW_STREAM                              DYW stream software
```

7.5 移除套件庫

1. adhoc 命令移除 DYW_BASE 套件庫。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m yum_repository -a "name=
    DYW_BASE state=absent"
kvm11 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "repo": "DYW_BASE",
    "state": "absent"
}
```

2. adhoc 命令移除 DYW_BASE 套件庫。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m yum_repository -a "name=
    DYW_STREAM state=absent"
kvm11 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "repo": "DYW_STREAM",
    "state": "absent"
}
```

3. ansible-playbook 執行 yum_repo.yml，就可重新設定。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook yum_repo.yml

PLAY [dev]
    ********************************************************************


TASK [Gathering Facts]
    *********************************************************
ok: [kvm11]
```

```
7
  TASK [Add BaseOS repository]
     ***************************************************
9 changed: [kvm11]

11 TASK [Add Stream repository]
     ***************************************************
  changed: [kvm11]
13
  PLAY RECAP
     *********************************************************************

15 kvm11 : ok=3  changed=2  unreachable=0  failed=0  skipped=0  rescued=0
     ignored=0
```

7.6 實機練習

1. 寫 yaml 腳本 pbook1.yml，對 dev 主機要求如下：
   (a) 使用 shell 模組執行"ls -l /bin/passwd"。
   (b) ansible-playbook 執行結果導向到 pbook1.txt。
2. 在控制主機以 ceuser 用户的 ansible 工作目錄，寫一腳本 yum_repo.yml，在每一個受管節點產生 yum repository 設定，要求如下：
   (a) Repository 1
      i. 名稱 (name)：DYW_BASE
      ii. 描述 (description)：DYW base software
      iii. 套件來源網址：http://dywang.csie.cyut.edu.tw/alma10/BaseOS
      iv. 啓用 GPG 簽章檢查
      v. GPG 金鑰網址：http:// dywang.csie.cyut.edu.tw/ materials/ RPM-GPG-KEY-AlmaLinux-10
   (b) Repository 2
      i. 名稱 (name)：DYW_STREAM
      ii. 描述 (description)：DYW stream software
      iii. 套件來源網址：http:// dywang.csie.cyut.edu.tw/ alma10/ App-Stream
      iv. 啓用 GPG 簽章檢查

v. GPG 金鑰網址：http:// dywang.csie.cyut.edu.tw/ materials/ RPM-GPG-KEY-AlmaLinux-10

(c) ansible-playbook 執行結果導向到 `yum_repo.txt`。

Chapter 8

Dnf 模組

8.1 文件查詢

1. dnf 模組可以做套件管理，完整文件可用 ansible-doc 查詢 dnf 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc dnf | grep ^EX -A10
EXAMPLES:

- name: Install the latest version of Apache
  ansible.builtin.dnf:
    name: httpd
    state: latest

- name: Install Apache >= 2.4
  ansible.builtin.dnf:
    name: httpd>=2.4
    state: present
```

2. 確認 dev 群組裡所有主機，套件 vim 還沒安裝，回傳成功。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m dnf -a "name=vim state=absent"
kvm11 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "msg": "Nothing to do",
    "rc": 0,
    "results": []
}
```

3. 使用 dnf 模組安裝 vim 至最新版本，回傳成功，含相依套件共安裝了 4 的套件。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m dnf -a "name=vim state=latest"
kvm11 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "msg": "",
    "rc": 0,
    "results": [
        "Installed: vim-common-2:9.1.083-6.el10_1.x86_64",
        "Installed: vim-enhanced-2:9.1.083-6.el10_1.x86_64",
        "Installed: vim-filesystem-2:9.1.083-6.el10_1.noarch",
        "Installed: xxd-2:9.1.083-6.el10_1.x86_64"
    ]
}
```

8.2 使用變數

1. 腳本 deploy1.yml，在 dev 群組主機安裝套件，套件名稱爲變數 item。

```
ceuser@deyu:~/ansible_pb$ vi deploy1.yml
ceuser@deyu:~/ansible_pb$ cat deploy1.yml
---
- hosts: dev
  tasks:
    - name: Install the latest version of ...
      ansible.builtin.dnf:
        name: "{{item}}"
        state: latest
```

2. 命令列使用 ansible-playbook 執行 deploy1.xml 腳本時，使用 -extra-vars 提供變數”item=php”，成功安裝 php 套件。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook deploy1.yml --extra-vars "
    item=php"

PLAY [dev]
    ***********************************************************************



TASK [Gathering Facts]
    *********************************************************
ok: [kvm11]

TASK [Install the latest version of ...]
    ***************************************
```

```
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm11 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. 使用 ad-hoc 命令安裝 dev 的 php 套件，回傳"changed": false, 表示 php 套件已經安裝，沒有做任何改變。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m dnf -a "name=php state=present"
kvm11 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "msg": "Nothing to do",
    "rc": 0,
    "results": []
}
```

8.3 loop 迴圈

1. ansible 2.5 前使用 with_* 關鍵字，例如 with_items 來運作迴圈。ansible 2.6 版後官方推薦 loop 關鍵字代替 with_*，一般而言，關鍵字 with_* 大多可用 loop 代替，兩者差異及迴圈詳細用法請參考 ansible 官方文件。

2. loop 類似程式的 for 迴圈，腳本 deploy2.yml，在 dev 群組主機安裝套件，套件名稱變數 item，此爲 loop 迴圈的預設變數，loop 迴圈列出 php 及 mariadb 兩個套件。

```
ceuser@deyu:~/ansible_pb$ vi deploy2.yml
ceuser@deyu:~/ansible_pb$ cat deploy2.yml
---
- hosts: dev
  tasks:
  - name: install package
    ansible.builtin.dnf:
      name: "{{item}}"
      state: latest
    loop:
    - php
    - mariadb
```

3. 命令列使用 ansible-playbook 執行 deploy2.xml 腳本，回傳 php 已經安裝，mariadb 成功安裝。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook deploy2.yml

PLAY [dev]
    ***********************************************************************


TASK [Gathering Facts]
    ***********************************************************
ok: [kvm11]

TASK [install package]
    ***********************************************************
ok: [kvm11] => (item=php)
changed: [kvm11] => (item=mariadb)

PLAY RECAP
    ***********************************************************************

kvm11      : ok=2 changed=1 unreachable=0 failed=0 skipped=0 rescued=0
    ignored=0
```

4. deploy2.yml 使用 loop 提供 dnf 安裝套件名稱，每個套件安裝都獨立執行 dnf。目前的 ansible 版本建議改用 dnf 模組內建的 name，一次指定多個套件，如下 deploy2a.yml。

```
ceuser@deyu:~/ansible_pb$ cat deploy2a.yml
---
- hosts: dev
  tasks:
  - name: install package
    ansible.builtin.dnf:
      name:
      - php
      - mariadb
      state: latest
```

5. 命令列使用 ansible-playbook 執行 deploy2a.xml 腳本，回傳 php 已經安裝，mariadb 成功安裝。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook deploy2a.yml
```

```
2
PLAY [dev]
    ********************************************************************

4
TASK [Gathering Facts]
    ***********************************************************
6 ok: [kvm11]

8 TASK [install package]
    ***********************************************************
ok: [kvm11]
10
PLAY RECAP
    ********************************************************************

12 kvm11      : ok=2 changed=0 unreachable=0 failed=0 skipped=0 rescued=0
    ignored=0
```

8.4 block 區塊

1. 當定義的任務只有符合條件的群組或主機才需要執行時，會把這部分任務定義成 block，配合 when 條件判斷，決定是否執行此 block。

2. 腳本 deploy3.yml，在 dev 及 test 群組主機安裝套件，block 定義安裝 Development Tools 群組套件，when 內縮位置對齊 block，表示條件 test 在 group_names 時才執行 block，group_names 變數是指目前執行的主機群組。

```
ceuser@deyu:~/ansible_pb$ vi deploy3.yml
2 ceuser@deyu:~/ansible_pb$ cat deploy3.yml
---
4 - hosts: dev test
  tasks:
6   - name: install package
      ansible.builtin.dnf:
8       name:
        - php
10      - mariadb
        state: latest
12  - block:
      - name: install the '@RPM Development Tools' package group
14      ansible.builtin.dnf:
          name: "@RPM Development Tools"
16        state: present
      when: "'test' in group_names"
```

3. 命令列使用 ansible-playbook 執行 deploy3.xml 腳本，回傳 kvm11 已安裝 php 及 mariadb，kvm12 成功安裝 php 及 mariadb。block 部分，kvm11 跳過，kvm12 成功安裝 Development Tools 群組套件。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook deploy3.yml

PLAY [dev test]
    ********************************************************************

TASK [Gathering Facts]
    ***********************************************************
ok: [kvm11]
ok: [kvm12]

TASK [install package]
    ***********************************************************
ok: [kvm11]
changed: [kvm12]

TASK [install the '@RPM Development Tools' package group]
    **********************
skipping: [kvm11.deyu.wang]
changed: [kvm12]

PLAY RECAP
    ********************************************************************

kvm11      : ok=2 changed=0 unreachable=0 failed=0 skipped=1 rescued=0
    ignored=0
kvm12      : ok=3 changed=1 unreachable=0 failed=0 skipped=0 rescued=0
    ignored=0
```

8.5 更新套件

1. 修改腳本 deploy3.yml 爲 deploy4.yml，增加任務在 test 更新 openssl 開頭的套件。

```
ceuser@deyu:~/ansible_pb$ cat deploy4.yml
---
- hosts: dev test
  tasks:
  - name: install package
    ansible.builtin.dnf:
      name:
      - php
      - mariadb
      state: latest
  - block:
```

```
    - name: install the '@RPM Development Tools' package group
      ansible.builtin.dnf:
        name: "@RPM Development Tools"
        state: present
    - name: upgrade packages
      ansible.builtin.dnf:
        name: 'openssl*'
        state: latest
    when: "'test' in group_names"
```

2. 命令列使用 ansible-playbook 執行 deploy4.xml 腳本。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook deploy4.yml

PLAY [dev test]
    ********************************************************************

TASK [Gathering Facts]
    ************************************************************
ok: [kvm12]
ok: [kvm11]

TASK [install package]
    ************************************************************
ok: [kvm11]
ok: [kvm12]

TASK [install the '@RPM Development Tools' package group]
    **********************
skipping: [kvm11]
ok: [kvm12]

TASK [upgrade packages]
    ***************************************************
skipping: [kvm11]
ok: [kvm12]

PLAY RECAP
    ********************************************************************************

kvm11      : ok=2 changed=0 unreachable=0 failed=0 skipped=2 rescued=0
    ignored=0
kvm12      : ok=4 changed=0 unreachable=0 failed=0 skipped=0 rescued=0
    ignored=0
```

8.6 實機練習

1. ad-hoc 命令使用 dnf 模組，在主機 test 安裝最新版的 $dnfpkg 套件。

2. 寫 yaml 腳本 pbook2.yml，對 dev 主機要求如下：
   (a) dnf 模組移除 xz 套件
   (b) 移除後 notify ”install xz”
   (c) handlers 使用 dnf 模組再安裝 xz
   (d) ansible-playbook 執行結果導向到 pbook2.txt。
3. 修改 pbook2.yml 爲 pbook3.yml，對 dev 主機要求如下：
   (a) dnf 模組移除的套件改成變數 item
   (b) 移除後 notify ”install item”
   (c) handlers 使用 dnf 模組再安裝 item
   (d) ansible-playbook 執行時使用 `--extra-vars` 指定套件名稱。
4. 寫 yaml 腳本 pbook4.yml，對 dev 主機要求如下：
   (a) 使用 loop 迴圈。
   (b) shell 模組 ls 列出 loop 迴圈的檔案，檔名由外加變數 f1 及 f2 決定。
5. 寫 yaml 腳本 pbook5.yml，對 dev 及 test 群組主機要求如下：
   (a) 安裝 xz 及 gzip 兩個套件
   (b) 使用 block ... when 語法，dev 才安裝 vim 套件。
6. 修改 pbook5.yml 爲 pbook6.yml，增加只有 dev 主機才更新 openssh 套件的任務。
7. 練習講義 deploy3.yml。

Chapter 9

Collections 內容集

9.1 前言

1. 外掛 (plugins)
   (a) 外掛是增強 ansible 核心功能的代碼片段。
   (b) ansible 使用外掛來實現豐富、靈活及可擴展的功能集。
   (c) Ansible 提供許多方便的外掛，例如：action, cache, callback, inventory, lookup, shell, vars，若無法滿足需求，也可自行編寫外掛。
   (d) 外掛與模組不同，只在 Ansible 控制節點上本地運行。例如：在 Ansible 控制節點任何時間都可以運行模組，是因爲先運行了 action 外掛。
   (e) ansible 外掛文件
2. 角色 (roles)
   (a) 很多時候希望有部分的部署內容是可以被其他不同的 playbook 重新使用，但卻不希望在每一個不同的 playbook 中都要重新定義一次。
   (b) Ansible 提供了角色 (role) 的概念，可以透過撰寫客製的 role 來讓所有 playbook 重複使用。
   (c) role 根據檔案結構自動載入變數檔 (`vars_files`)、任務 (tasks)、善後程序 (handlers)。
3. 內容集 (collections)
   (a) 以往模組開發者必須等模組被標記在 ansible 發行版中，或者將模組加到角色。
   (b) 內容集 (Collections) 是 Ansible 內容的分發 (distribution) 格式，內容包括特定主題的 playbooks, roles, modules 及 plugins。

(c) 內容集採用一致的格式，讓內容集開發者將 modules, plugins, roles, playbooks 綁在一起交付，並讓用户能在單一位置使用這些內容。

(d) 內容集的發布及維護獨立於 ansible 發行版本，內容的更新可以更頻繁快速。

9.2 ansible-galaxy 指令

1. ansible-galaxy 可以管理 collection 及 role 兩種類型。

```
[deyu1@kvm19 ~]$ ansible-galaxy -h
usage: ansible-galaxy [-h] [--version] [-v] TYPE ...

Perform various Role and Collection related operations.

positional arguments:
  TYPE
    collection   Manage an Ansible Galaxy collection.
    role         Manage an Ansible Galaxy role.

optional arguments:
  --version      show program's version number, config file location,
                 configured module search path, module location,
                     executable
                 location and exit
  -h, --help     show this help message and exit
  -v, --verbose  Causes Ansible to print more debug messages. Adding
      multiple
                 -v will increase the verbosity, the builtin plugins
                     currently
                 evaluate up to -vvvvvv. A reasonable level to start is
                     -vvv,
                 connection debugging might require -vvvv.
```

2. ansible-galaxy collection 工具可以 download, init, build, build, publish, install, list 內容集。

```
[deyu1@kvm19 ~]$ ansible-galaxy collection -h
usage: ansible-galaxy collection [-h] COLLECTION_ACTION ...

positional arguments:
  COLLECTION_ACTION
    download         Download collections and their dependencies as a
        tarball
                     for an offline install.
    init             Initialize new collection with the base structure
        of a
                     collection.
```

```
    build           Build an Ansible collection artifact that can be
        published
                    to Ansible Galaxy.
    publish         Publish a collection artifact to Ansible Galaxy.
    install         Install collection(s) from file(s), URL(s) or
        Ansible
                    Galaxy
    list            Show the name and version of each collection
        installed in
                    the collections_path.
    verify          Compare checksums with the collection(s) found on
        the
                    server and the installed copy. This does not verify
                    dependencies.

optional arguments:
  -h, --help        show this help message and exit
```

3. ansible-galaxy collection 工具的每個子命令都有輔助說明，例如以下查詢子命令 install 的輔助說明。

```
[deyu1@kvm19 ansible]$ ansible-galaxy collection install -h
usage: ansible-galaxy collection install [-h] [-s API_SERVER] [--token
    API_KEY]
                                         [-c] [-v] [-f]
                                         [--clear-response-cache] [--no-
                                             cache]
                                         [-i] [-n | --force-with-deps]
                                         [-p COLLECTIONS_PATH]
                                         [-r REQUIREMENTS] [--pre] [-U]
                                         [--keyring KEYRING]
                                         [--disable-gpg-verify]
                                         [--signature SIGNATURES]
                                         [--required-valid-signature-
                                             count
                                             REQUIRED_VALID_SIGNATURE_COUNT
                                             ]
                                         [--ignore-signature-status-code
                                              {EXPSIG,EXPKEYSIG,
                                             REVKEYSIG,BADSIG,ERRSIG,
                                             NO_PUBKEY,
                                             MISSING_PASSPHRASE,
                                             BAD_PASSPHRASE,NODATA,
                                             UNEXPECTED,ERROR,FAILURE,
                                             BADARMOR,KEYEXPIRED,
                                             KEYREVOKED,NO_SECKEY}]
                                         [collection_name ...]

positional arguments:
  collection_name       The collection(s) name or path/url to a tar.gz
                        collection artifact. This is mutually exclusive
```

```
                        with
                    --requirements-file.

optional arguments:
  -h, --help            show this help message and exit
  -s API_SERVER, --server API_SERVER
                        The Galaxy API server URL
  --token API_KEY, --api-key API_KEY
                        The Ansible Galaxy API key which can be found at
                        https://galaxy.ansible.com/me/preferences.
  -c, --ignore-certs    Ignore SSL certificate validation errors.
  -v, --verbose         Causes Ansible to print more debug messages.
    Adding
                        multiple -v will increase the verbosity, the
                            builtin
                        plugins currently evaluate up to -vvvvvv. A
                            reasonable
                        level to start is -vvv, connection debugging
                            might
                        require -vvvv.
  -f, --force           Force overwriting an existing role or collection
  --clear-response-cache
                        Clear the existing server response cache.
  --no-cache            Do not use the server response cache.
  -i, --ignore-errors   Ignore errors during installation and continue
    with the
                        next specified collection. This will not ignore
                        dependency conflict errors.
  -n, --no-deps         Don't download collections listed as
    dependencies.
  --force-with-deps     Force overwriting an existing collection and its
                        dependencies.
  -p COLLECTIONS_PATH, --collections-path COLLECTIONS_PATH
                        The path to the directory containing your
                            collections.
  -r REQUIREMENTS, --requirements-file REQUIREMENTS
                        A file containing a list of collections to be
                        installed.
  --pre                 Include pre-release versions. Semantic
    versioning pre-
                        releases are ignored by default
  -U, --upgrade         Upgrade installed collection artifacts. This
    will also
                        update dependencies unless --no-deps is provided
  --keyring KEYRING     The keyring used during signature verification
  --disable-gpg-verify  Disable GPG signature verification when
    installing
                        collections from a Galaxy server
  --signature SIGNATURES
                        An additional signature source to verify the
                        authenticity of the MANIFEST.json before
                            installing the
                        collection from a Galaxy server. Use in
                            conjunction
```

```
                        with a positional collection name (mutually
                            exclusive
60                      with --requirements-file).
    --required-valid-signature-count REQUIRED_VALID_SIGNATURE_COUNT
62                      The number of signatures that must successfully
                            verify
                        the collection. This should be a positive
                            integer or -1
64                      to signify that all signatures must be used to
                            verify
                        the collection. Prepend the value with + to fail
                             if no
66                      valid signatures are found for the collection (e
                            .g.
                        +all).
68  --ignore-signature-status-code {EXPSIG,EXPKEYSIG,REVKEYSIG,BADSIG,
        ERRSIG,NO_PUBKEY,MISSING_PASSPHRASE,BAD_PASSPHRASE,NODATA,
        UNEXPECTED,ERROR,FAILURE,BADARMOR,KEYEXPIRED,KEYREVOKED,NO_SECKEY}
                        A status code to ignore during signature
                            verification
70                      (for example, NO_PUBKEY). Provide this option
                            multiple
                        times to ignore a list of status codes.
                            Descriptions
72                      for the choices can be seen at L(https://github.
                            com/gpg
                        /gnupg/blob/master/doc/DETAILS#general-status-
                            codes).
```

9.3 * 初始化內容集

1. 建立 ansible/collections 目錄，並進入目錄。

```
1 [deyu1@kvm19 ~]$ mkdir -p ansible/collections
  [deyu1@kvm19 ~]$ cd ansible/collections
3 [deyu1@kvm19 collections]$
```

2. init 初使化一個內容集，名稱格式必須是 <namespace>.<collection>。

```
1 [deyu1@kvm19 collections]$ ansible-galaxy collection init mynmspace.
      mycollect
- Collection mynmspace.mycollect was created successfully
```

3. 查看目錄，建立內容集需要的目錄及檔案。

```
[deyu1@kvm19 collections]$ tree .
.
    mynmspace
        mycollect
            docs
            galaxy.yml
            plugins
                README.md
            README.md
            roles

5 directories, 3 files
```

9.4 * 編輯打包壓縮內容集

1. 編輯 galayxy.yml。

```
[deyu1@kvm19 collections]$ vim mynmspace/mycollect/galaxy.yml
[deyu1@kvm19 collections]$ grep deyu.wang mynmspace/mycollect/galaxy.yml
- deyu1@deyu.wang
repository: http://deyu.wang/repository
documentation: http://docs.deyu.wang
homepage: http://deyu.wang
issues: http://deyu.wang/issue/tracker
```

2. build 建立內容集打包壓縮檔。

```
[deyu1@kvm19 collections]$ ansible-galaxy collection build mynmspace/
    mycollect/
Created collection for mynmspace.mycollect at
/home/deyu1/ansible/collections/mynmspace-mycollect-1.0.0.tar.gz
```

3. 查看結果，產生第一版的 mynmsapce-mycollect。

```
[deyu1@kvm19 collections]$ ll
total 8
drwxr-xr-x. 3 deyu1 deyu1   23 Feb 14 16:57 mynmspace
-rw-r--r--. 1 deyu1 deyu1 1396 Feb 14 19:12 mynmspace-mycollect-1.0.0.
    tar.gz
-rw-r--r--. 1 deyu1 deyu1 1873 Feb 14 18:51 mynmspace.tar.gz
[deyu1@kvm19 collections]$ rm mynmspace.tar.gz
[deyu1@kvm19 collections]$ ll
total 4
```

```
drwxr-xr-x. 3 deyu1 deyu1   23 Feb 14 16:57 mynmspace
-rw-r--r--. 1 deyu1 deyu1 1396 Feb 14 19:12 mynmspace-mycollect-1.0.0.
    tar.gz
```

9.5 預設路徑安裝內容集

1. 設定 Collections 路徑：修改工作目錄下的 ansible.cfg，設定內容集的路徑爲 /home/ceuser/ansible_pb/collections。

```
ceuser@deyu:~/ansible_pb$ grep collections_path= ansible.cfg
collections_path=/home/ceuser/ansible_pb/collections
```

2. 不指定路徑安裝 mynmspace.mycollect:1.0.0。

```
ceuser@deyu:~/ansible_pb$ ansible-galaxy collection install http://
    dywang.csie.cyut.edu.tw/materials/mynmspace-mycollect-1.0.0.tar.gz
Downloading http://dywang.csie.cyut.edu.tw/materials/mynmspace-mycollect
    -1.0.0.tar.gz to /home/ceuser/.ansible/tmp/ansible-local-182235
    pynh7qf2/tmpljon00ig/mynmspace-mycollect-1.0.0-s7thcr7d
Starting galaxy collection install process
Process install dependency map
Starting collection install process
Installing 'mynmspace.mycollect:1.0.0' to '/home/ceuser/ansible_pb/
    collections/ansible_collections/mynmspace/mycollect'
mynmspace.mycollect:1.0.0 was installed successfully
```

3. 安裝時沒指定安裝路徑，預設安裝在 ansible.cfg 設定的路徑 ~/ansible/collections/，不用先建目錄 ~/ansible/collections，安裝時會自動產生。

```
ceuser@deyu:~/ansible_pb$ ansible-galaxy collection list

# /home/ceuser/ansible_pb/collections/ansible_collections
Collection          Version
------------------- -------
mynmspace.mycollect 1.0.0
```

4. 查看目錄 ~/ansible/collections 下安裝的內容集檔案。

```
ceuser@deyu:~/ansible_pb$ tree collections/
collections/
```

```
    ansible_collections
        mynmspace
            mycollect
                docs
                FILES.json
                MANIFEST.json
                plugins
                    README.md
                README.md
                roles

7 directories, 4 files
```

9.6 * 指定路徑安裝內容集

1. 安裝時指定安裝路徑 ~/myansible，出現指定路徑不在 ansible 設定的內容集路徑中，但還是可以成功安裝。

```
[deyu1@kvm19 ansible]$ ansible-galaxy collection install \
http://dywang.csie.cyut.edu.tw/materials/mynmspace-mycollect-1.0.0.tar.
    gz -p ~/myansible
Downloading http://dywang.csie.cyut.edu.tw/materials/mynmspace-mycollect
    -1.0.0.tar.gz to
/home/deyu1/.ansible/tmp/ansible-local-2303w0b9nla7/tmpldyx_ivo/
    mynmspace-mycollect-1.0.0-rap_7_4q
Starting galaxy collection install process
[WARNING]: The specified collections path '/home/deyu1/myansible' is not
     part of
the configured Ansible collections paths '/home/deyu1/ansible/
    collections'. The
installed collection won't be picked up in an Ansible run.
Process install dependency map
Starting collection install process
Installing 'mynmspace.mycollect:1.0.0' to
'/home/deyu1/myansible/ansible_collections/mynmspace/mycollect'
mynmspace.mycollect:1.0.0 was installed successfully
```

2. 查看 ~/myansible/。

```
[deyu1@kvm19 ansible]$ tree ~/myansible/
/home/deyu1/myansible/
    ansible_collections
        mynmspace
            mycollect
                docs
                FILES.json
```

```
                MANIFEST.json
9               plugins
                    README.md
11              README.md
                roles
13
6 directories, 4 files
```

9.7 實機練習

1. 安裝以下內容集
   (a) ansible-posix-1.5.1.tar.gz
   (b) community-general-6.3.0.tar.gz
   (c) rhel-system_roles-1.0.0.tar.gz
2. 安裝成功後工作目錄下，執行以下命令：
   (a) ansible-galaxy collection list 查看出現哪些內容集？並導向到 collectionlist.txt。
   (b) ll 查看內容集路徑下的 ansible_collections，出現哪些目錄？並導向到 llcollections.txt。

Chapter 10

Roles - RHEL System

10.1 前言

1. RHEL 已建立了幾個常用的 system roles。
2. RHEL system roles 可用 dnf 安裝內建的 rhel-system-roles.xxxx.rpm 套件，也可從前一章實機練習安裝的 `rhel.system_roles` 內容集取得。
3. 本章不使用 dnf 安裝 rhel-system-roles.xxxx.rpm 套件，只練習使用 `rhel.system_roles` 內容集中的角色。
4. `rhel.system_roles` 包含了 selinux, timesync, ssh, sshd, postfix, network, ... 等許多系統設定必用或常用的角色。

10.2 內容集的角色文件

1. 查詢可用的 RHEL system roles。

```
ceuser@deyu:/ansible_pb$ ls collections/ansible_collections/rhel/
    system_roles/roles
certificate                         private_metrics_subrole_elasticsearch
cockpit                             private_metrics_subrole_grafana
crypto_policies                     private_metrics_subrole_mssql
firewall                            private_metrics_subrole_pcp
ha_cluster                          private_metrics_subrole_postfix
kdump                               private_metrics_subrole_redis
kernel_settings                     private_metrics_subrole_repository
logging                             private_metrics_subrole_spark
metrics                             selinux
nbde_client                         ssh
nbde_server                         sshd
network                             storage
postfix                             timesync
```

```
private_logging_subrole_rsyslog   tlog
private_metrics_subrole_bpftrace  vpn
```

2. 查看角色 timesync 的使用文件，其中範例如下：

```
ceuser@deyu:/ansible_pb$ grep ^Example -A17 ...../system_roles/roles/
    timesync/README.md
Example Playbook
----------------

Install and configure ntp to synchronize the system clock with three NTP
     servers:

```yaml
- hosts: targets
  vars:
    timesync_ntp_servers:
      - hostname: foo.example.com
        iburst: yes
      - hostname: bar.example.com
        iburst: yes
      - hostname: baz.example.com
        iburst: yes
  roles:
    - rhel.system_roles.timesync
```
```

10.3 使用 RHEL 角色

1. 參考 timesync 角色文件的範例，撰寫 timesync.yml 腳本，在 dev 受管主機使用 timesync 角色啓用校時服務，時間伺服器 192.168.122.1，啓用 iburst 參數。

```
ceuser@deyu:/ansible_pb$ vi timesync.yml
ceuser@deyu:/ansible_pb$ cat timesync.yml
---
- hosts: dev
  vars:
    timesync_ntp_servers:
      - hostname: 192.168.122.1
        iburst: yes
  roles:
    - rhel.system_roles.timesync
```

2. ansible-playbook 執行 timesync.yml 腳本，輸出 failed=0 表示成功。

```
ceuser@deyu:/ansible_pb$ ansible-playbook timesync.yml

PLAY [dev] ********************************************************************

TASK [Gathering Facts] ********************************************************
ok: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Set version specific variables] *******
ok: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Populate service facts] ***************
[WARNING]: Unable to query 'service' tool (1):
ok: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Set variable `timesync_services` with filtered uniq service names] ***
ok: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Check that variable 'timesync_services' is defined] ***
ok: [kvm11] => {
    "changed": false,
    "msg": "All assertions passed"
}

TASK [redhat.rhel_system_roles.timesync : Check if only NTP is needed] **********
ok: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Check if single PTP is needed] ********
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Check if both NTP and PTP are needed] ***
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Determine current NTP provider] *******
ok: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Select NTP provider] ******************
ok: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Install chrony] **********************
```

```
ok: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Install ntp]
    ***************************
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Install linuxptp]
    **********************
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Gather package facts]
    ******************
ok: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Run phc_ctl on PTP interface]
    *********
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Check if PTP interface
    supports HW timestamping] ***
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Generate chrony.conf file]
    ************
changed: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Generate chronyd sysconfig
    file] ******
changed: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Generate ntp.conf file]
    ***************
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Generate ntpd sysconfig file]
    *********
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Generate ptp4l.conf file]
    **************
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Generate ptp4l sysconfig file]
     ********
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Generate phc2sys sysconfig
    file] ******
skipping: [kvm11]

TASK [redhat.rhel_system_roles.timesync : Generate timemaster.conf file]
     ********
skipping: [kvm11]

```

```
TASK [redhat.rhel_system_roles.timesync : Update network sysconfig file]
    ********
82 changed: [kvm11]

84 TASK [redhat.rhel_system_roles.timesync : Disable chronyd]
    **********************
   skipping: [kvm11]
86
   TASK [redhat.rhel_system_roles.timesync : Disable ntpd]
    *************************
88 skipping: [kvm11]

90 TASK [redhat.rhel_system_roles.timesync : Disable ntpdate]
    **********************
   skipping: [kvm11]
92
   TASK [redhat.rhel_system_roles.timesync : Disable sntp]
    *************************
94 skipping: [kvm11]

96 TASK [redhat.rhel_system_roles.timesync : Disable ptp4l]
    *************************
   skipping: [kvm11]
98
   TASK [redhat.rhel_system_roles.timesync : Disable phc2sys]
    **********************
100 skipping: [kvm11]

102 TASK [redhat.rhel_system_roles.timesync : Disable timemaster]
    ********************
    skipping: [kvm11]
104
    TASK [redhat.rhel_system_roles.timesync : Enable chronyd]
    **********************
106 ok: [kvm11]

108 TASK [redhat.rhel_system_roles.timesync : Enable ntpd]
    *************************
    skipping: [kvm11]
110
    TASK [redhat.rhel_system_roles.timesync : Enable ptp4l]
    *************************
112 skipping: [kvm11]

114 TASK [redhat.rhel_system_roles.timesync : Enable phc2sys]
    **********************
    skipping: [kvm11]
116
    TASK [redhat.rhel_system_roles.timesync : Enable timemaster]
    ********************
118 skipping: [kvm11]

120 RUNNING HANDLER [redhat.rhel_system_roles.timesync : restart chronyd]
    ***********
```

```
changed: [kvm11]
122
PLAY RECAP
    *********************************************************************

124 kvm11  : ok=15 changed=4  unreachable=0  failed=0 skipped=23 rescued=0
    ignored=0
```

3. 查詢 dev 主機的時間校時服務的設定，是否依 timesync.yml 腳本配置校時伺服器及參數？

```
1 ceuser@deyu:/ansible_pb$ ansible dev -m shell -a "grep ^server /etc/
    chrony.conf"
kvm11 | CHANGED | rc=0 >>
3 server 192.168.122.1 iburst
```

10.4 實機練習

1. 撰寫 timesync.yml，在所有受管主機使用 timesync 角色啓用校時服務，時間伺服器 192.168.122.1，啓用 iburst 參數。

2. 撰寫 postfix.yml，對 prod 主機使用 rhel 系統角色 postfix，設定 relayhost 爲 deyu.wang。

3. 撰寫 selinux.yml，在所有受管主機使用 selinux 角色，設定 SELinux 策略爲 targeted，狀態爲 enforcing。

Chapter 11

Roles - Ansible Galaxy

11.1 前言

1. Ansible Galaxy 是一個共享和下載 Ansible 角色的網站。
2. Ansible Galaxy 網站可免費查詢下載角色，評論各種社群開發的 Ansible 角色。

11.2 ansible-galaxy 命令

1. ansible-galaxy 管理 role 的輔助說明，ROLE_ACTION 列出的 [init|remove|... 是可用的子命令。

```
ceuser@deyu:/ansible_pb$ ansible-galaxy role --help
usage: ansible-galaxy role [-h] ROLE_ACTION ...

positional arguments:
  ROLE_ACTION
    init        Initialize new role with the base structure of a role.
    remove      Delete roles from roles_path.
    delete      Removes the role from Galaxy. It does not remove or alter
         the
                actual GitHub repository.
    list        Show the name and version of each role installed in the
                roles_path.
    search      Search the Galaxy database by tags, platforms, author and
                multiple keywords.
    import      Import a role into a galaxy server
    setup       Manage the integration between Galaxy and the given
        source.
    info        View more details about a specific role.
    install     Install role(s) from file(s), URL(s) or Ansible Galaxy

optional arguments:
```

```
  -h, --help    show this help message and exit
```

2. ansible-galaxy role 每個子命令都有輔助說明，例如以下查詢 search 的輔助說明。

```
ceuser@deyu:/ansible_pb$ ansible-galaxy role search -h
usage: ansible-galaxy role search [-h] [-s API_SERVER] [--token API_KEY]
    [-c]
                                  [-v] [--platforms PLATFORMS]
                                  [--galaxy-tags GALAXY_TAGS] [--author
                                      AUTHOR]
                                  [searchterm ...]

positional arguments:
  searchterm            Search terms

optional arguments:
  -h, --help            show this help message and exit
  -s API_SERVER, --server API_SERVER
                        The Galaxy API server URL
  --token API_KEY, --api-key API_KEY
                        The Ansible Galaxy API key which can be found at
                        https://galaxy.ansible.com/me/preferences.
  -c, --ignore-certs    Ignore SSL certificate validation errors.
  -v, --verbose         Causes Ansible to print more debug messages.
      Adding
                        multiple -v will increase the verbosity, the
                            builtin
                        plugins currently evaluate up to -vvvvvv. A
                            reasonable
                        level to start is -vvv, connection debugging
                            might
                        require -vvvv.
  --platforms PLATFORMS
                        list of OS platforms to filter by
  --galaxy-tags GALAXY_TAGS
                        list of galaxy tags to filter by
  --author AUTHOR       GitHub username
```

11.3 search

1. ansible-galaxy 搜尋 haproxy，找到 200 項

```
ceuser@deyu:/ansible_pb$ ansible-galaxy role search haproxy

Found 200 roles matching your search:
```

```
4
                                                    
Name                                                Description
6   ----                                            -----------
    9brandon2.haproxy                               Install and configure
        haproxy>
8   aaronpederson.haproxy                           The Reliable, High
        Performanc>
    acandid.haproxy                                 Install and Configure
        haproxy>
10  alikins.haproxy                                 HAProxy installation
        and conf>
    andrewrothstein.haproxy                         installs HAProxy
12  ansiblebit.haproxy                              Role to setup haproxy.
    antonmatsiuk.prom_haproxy_exporter              Prometheus HAProxy
        exporter o>
14  automium.haproxy                                Role for deploying
        HAProxy
    awcrosby.ansible_haproxy                        Playbook for HAProxy.
16  bdellegrazie.haproxy_exporter                   Role to install
        Prometheus HA>
    Blue-Bag.haproxy                                HAProxy installation
        and conf>
18  blurrcat.haproxy                                Installs and configure
         haproxy
    budimanjojo.haproxy                             Role to install and
        configure>
20  buluma.haproxy                                  Install and configure
        haproxy>
    caermeglaeddyv.ansible_role_haproxy             Linux / Kubernetes
        Administra>
22  calvinbui.ansible_haproxy_docker                HAProxy in Docker
    CaptTofu.galera-haproxy                         An ansible role for
        HAProxy w>
24  castironclay.heartbeat_haproxy                  your description
    Chillisystems.ansible-haproxy                   Set up the latest
        version of >
26  chmduquesne.haproxy                             Installs and
        configures hapro>
    chrislmayes.ansible_haproxy                     Installs and configure
         haproxy
28  ckaserer.haproxy                                Set up the latest
        version of >
    Cloud-PG.haproxy-consul                         Install haproxy-consul
30  cmndrsp0ck.ansible-haproxy-tls-passthrough      haproxy configured
        with float>
    colstrom.haproxy                                The Reliable, High
        Performanc>
32  cowops.debian-haproxy                           Install Haproxy for
        debian sy>
    CSCfi.haproxy                                   HAProxy installation
        and conf>
34  cwerner1.haproxy                                Set up the latest
        version of >
    danielino.haproxy                               Role to manage haproxy
```

```
     instal>
36
 ----                                    -----------
```

11.4 install/remove

1. install 從網站下載角色並安裝。

```
1 ceuser@deyu:/ansible_pb$ ansible-galaxy role install ansiblebit.haproxy
  ceuser@deyu:~/ansible_pb$ ansible-galaxy role install ansiblebit.haproxy
3 Starting galaxy role install process
  - downloading role 'haproxy', owned by ansiblebit
5 - downloading role from https://github.com/ansiblebit/haproxy/archive/
      master.tar.gz
  - extracting ansiblebit.haproxy to /home/ceuser/ansible_pb/roles/
      ansiblebit.haproxy
7 - ansiblebit.haproxy (master) was installed successfully
```

2. ansible-galaxy list 列出 ansiblebit.haproxy 角色。

```
1 ceuser@deyu:/ansible_pb$ ansible-galaxy role list
  # /home/ceuser/ansible_pb/roles
3 - ansiblebit.haproxy, master
  # /usr/share/ansible/roles
5 # /etc/ansible/roles
```

3. 查詢 roles 目錄新增了 ansiblebit.haproxy 目錄，目錄內有角色的各種檔案。

```
1  ceuser@deyu:/ansible_pb$ ll roles/ansiblebit.haproxy/
   total 20
3  drwxr-xr-x. 2 ceuser ceuser   22 Apr  8 14:02 defaults
   drwxr-xr-x. 2 ceuser ceuser   22 Apr  8 14:02 handlers
5  -rw-r--r--. 1 ceuser ceuser 1479 Jun 23  2016 LICENSE
   drwxr-xr-x. 2 ceuser ceuser   50 Apr  8 14:02 meta
7  -rw-r--r--. 1 ceuser ceuser  298 Jun 23  2016 NOTICE.md
   -rw-r--r--. 1 ceuser ceuser 2524 Jun 23  2016 README.md
9  -rw-r--r--. 1 ceuser ceuser    0 Jun 23  2016 requirements.yml
   drwxr-xr-x. 2 ceuser ceuser   92 Apr  8 14:02 tasks
11 drwxr-xr-x. 3 ceuser ceuser   17 Apr  8 14:02 templates
   drwxr-xr-x. 7 ceuser ceuser 4096 Apr  8 14:02 tests
13 -rw-r--r--. 1 ceuser ceuser  369 Jun 23  2016 tox.ini
   drwxr-xr-x. 2 ceuser ceuser   22 Apr  8 14:02 vars
```

4. remove 移除角色 ansiblebit.haproxy。

```
ceuser@deyu:/ansible_pb$ ansible-galaxy role remove ansiblebit.haproxy
- successfully removed ansiblebit.haproxy
```

5. ansible-galaxy list 沒有列出 ansiblebit.haproxy 角色。

```
ceuser@deyu:/ansible_pb$ ansible-galaxy role list
# /home/ceuser/ansible_pb/roles
# /usr/share/ansible/roles
# /etc/ansible/roles
```

6. 查詢 roles 目錄，ansiblebit.haproxy 目錄已被刪除。

```
ceuser@deyu:/ansible_pb$ ll roles/
total 0
```

11.5 requirements.yml

1. 除了直接從網站下載安裝角色外，也可同時將多個安裝來源及參數寫成 YAML 格式檔，而且副檔名一定要.yml 或.yaml。

2. 通常 ansible-galaxy 安裝要求檔命名成 requirements.yml，以選項 -r 指定此檔案進行安裝。

```
$ ansible-galaxy role install -r requirements.yml
```

3. requirements.yml 屬性：

   (a) src：必要屬性，指定安裝角色的來源。

   (b) scm：如果 src 是 URL，必須指定 scm 是 git 或 hg。

   (c) version：角色的版本。

   (d) name：角色的名稱。

4. requirements.yml 範例：

```
# from galaxy
- src: yatesr.timezone

# from GitHub
- src: https://github.com/bennojoy/mysql

# from GitHub, overriding the name and specifying a specific tag
- src: https://github.com/bennojoy/mysql
  version: master
  name: mysql_role

# from a webserver, where the role is packaged in a tar.bz2
- src: https://dywang.csie.cyut.edu.tw/materials/haproxy.tar.bz2
  name: haproxy-role

# from Bitbucket
- src: git+http://bitbucket.org/willthames/git-ansible-galaxy
  version: v1.4

# from Bitbucket, alternative syntax and caveats
- src: http://bitbucket.org/willthames/hg-ansible-galaxy
  scm: hg

# from GitLab or other git-based scm
- src: git@gitlab.company.com:mygroup/ansible-base.git
  scm: git
  version: "0.1" #quoted, so YAML doesn't parse this as a floating-point
       value
```

11.6 requirements.yml 安裝實例

1. 在 ansible 目錄下編輯 requirements.yml，角色來源爲 http 網站的 haproxy.tar.bz2，安裝名稱爲 balancer 的角色。

```
ceuser@deyu:/ansible_pb$ vi requirements.yml
ceuser@deyu:/ansible_pb$ cat requirements.yml
---
- src: http://dywang.csie.cyut.edu.tw/materials/haproxy.tar.bz2
  name: balancer

- src: http://dywang.csie.cyut.edu.tw/materials/phpinfo.tar.bz2
  name: phpinfo
```

2. 使用 requirements.yml 安裝角色。

```
ceuser@deyu:/ansible_pb$ ansible-galaxy role install -r requirements.yml
```

```
Starting galaxy role install process
- downloading role from http://dywang.csie.cyut.edu.tw/materials/haproxy
    .tar.bz2
- extracting balancer to /home/ceuser/ansible_pb/roles/balancer
- balancer was installed successfully
- downloading role from http://dywang.csie.cyut.edu.tw/materials/phpinfo
    .tar.bz2
- extracting phpinfo to /home/ceuser/ansible_pb/roles/phpinfo
- phpinfo was installed successfully
```

3. ansible-galaxy list 列出 balancer 及 phpinfo 兩個角色。

```
ceuser@deyu:/ansible_pb$ ansible-galaxy role list
# /home/ceuser/ansible_pb/roles
- phpinfo, (unknown version)
- balancer, (unknown version)
# /usr/share/ansible/roles
# /etc/ansible/roles
```

4. 查詢 roles 目錄，新增了 balancer 及 phpinfo 兩個角色目錄。

```
ceuser@deyu:/ansible_pb$ ll roles/
total 0
drwxr-xr-x. 9 ceuser ceuser 122 Apr  8 14:07 balancer
drwxr-xr-x. 9 ceuser ceuser 122 Apr  8 14:07 phpinfo
```

11.7 實機練習

1. ansible-galaxy role 搜尋 gastr (查參數檔)，並將結果存到 gastr.txt
2. 在 ansible 目錄下編輯 requirements.yml，角色來源爲 dywang.csie.cyut.edu.tw 網站：
   (a) haproxy.tar.bz2，安裝名稱爲 balancer 的角色。
   (b) phpinfo.tar.bz2，安裝名稱爲 phpinfo 的角色。
3. 撰寫 logrotate.yml
   (a) ansible-galaxy role 搜尋 arillso，找到有關 logrotate 角色。
   (b) 安裝....logrotate 角色

(c) 查看角色....logrotate 的 README.md，找到 Example Playbook 的 yml 腳本。

(d) 參考範例 yml 腳本，只針對 test 主機，使用....logrotate 角色，不變更任務，設定 logrotate。

Chapter 12

Jinja2 樣板

12.1 前言

1. 如果在 100 台主機安裝 httpd 服務，每台的歡迎網頁要顯示主機 IP 或名稱，如何解決？
2. ansible 使用 jinja2 樣板修改受管主機的配置文件，可解決此問題。
3. 使用 jinja2 樣板也就是使用 template 模組，template 模組和 copy 模組一樣，都是將檔案複製到遠端主機，不同的是，template 模組可以取得檔案中的變數，而 copy 則是原封不動的把檔案複製過去。
4. Ansible 還允許 jinja2 樣板使用條件判斷和迴圈，但不允許在 playbook 中使用。

12.2 基本語法

1. 輸出自定義或 facts 變數值。

```
{{ EXPR }}
```

2. 註解

```
1 {# COMMENT #}
```

3. 邏輯判斷

```
{% if EXPR %}
{% elif EXPR %}
{% else %}
{% endif %}
```

4. for 迴圈

```
{% for i in EXPR %}
{% endfor %}
```

12.3 變數

1. 合法變數名：可以是字母、數字或下底線，例如：

```
foo_port
foo7
```

2. Magic 變數

   (a) ansible 保留變數，user 不能設定。

   (b) 變數包括使用的 python 版本、inventory 中的主機群組、roles 的目錄等。

   (c) 常用的 magic 變數例如：

```
ansible_version
group_names
groups
hostvars
inventory_hostname
inventory_dir
inventory_file
playbook_dir
role_name
role_names
role_path
```

   (d) 執行 debug 模組取得變數內容，不指定變數 debug 模組只印出"Hello world!"。

```
ceuser@deyu:/ansible_pb$ ansible dev -m debug
kvm11 | SUCCESS => {
    "msg": "Hello world!"
}
```

3. Facts 變數：

   (a) Facts 透過訪問遠端主機取得相應的資訊。

   (b) 例如：遠端主機 IP、作業系統、主機名稱。

   (c) 變數都以 ansible_ 開頭。

   (d) 執行 setup 模組時收集。

```
ceuser@deyu:/ansible_pb$ ansible dev -m setup | head
kvm11 | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "192.168.122.11"
        ],
        "ansible_all_ipv6_addresses": [
            "2001:ac18::5054:ff:fe46:94e7",
            "fe80::5054:ff:fe46:94e7"
        ],
        "ansible_apparmor": {
```

12.4 debug 輸出 magic 變數

1. debug 模組列印變數 groups，內容由 inventory 設定取得。

```
ceuser@deyu:/ansible_pb$ ansible -m debug dev -a "var=groups"
kvm11 | SUCCESS => {
    "groups": {
        "all": [
            "kvm11",
            "kvm12",
            "kvm15",
            "kvm13",
            "kvm14"
        ],
        "balancers": [
            "kvm15"
        ],
        "dev": [
            "kvm11"
```

```
        ],
        "prod": [
            "kvm13",
            "kvm14"
        ],
        "test": [
            "kvm12"
        ],
        "ungrouped": [],
        "webservers": [
            "kvm13",
            "kvm14"
        ]
    }
}
```

2. debug 模組列印變數 group_names，內容由 inventory 設定取得。

```
ceuser@deyu:/ansible_pb$ ansible -m debug all -a "var=group_names"
kvm11 | SUCCESS => {
    "group_names": [
        "dev"
    ]
}
kvm12 | SUCCESS => {
    "group_names": [
        "test"
    ]
}
kvm15 | SUCCESS => {
    "group_names": [
        "balancers"
    ]
}
kvm13 | SUCCESS => {
    "group_names": [
        "prod",
        "webservers"
    ]
}
kvm14 | SUCCESS => {
    "group_names": [
        "prod",
        "webservers"
    ]
}
```

3. debug 模組列印變數 hostvars['kvm11']，ansbile_facts 是空的，因爲 debug 模組沒有執行 setup 模組。

```
ceuser@deyu:/ansible_pb$ ansible -m debug dev -a "var=hostvars['kvm11']"
kvm11 | SUCCESS => {
    "hostvars['kvm11']": {
        "ansible_check_mode": false,
        "ansible_config_file": "/home/ceuser/ansible_pb/ansible.cfg",
        "ansible_diff_mode": false,
        "ansible_facts": {},
        "ansible_forks": 5,
        "ansible_inventory_sources": [
            "/home/ceuser/10927001/inventory"
        ],
        "ansible_playbook_python": "/usr/bin/python3",
        "ansible_verbosity": 0,
        "ansible_version": {
            "full": "2.16.14",
            "major": 2,
            "minor": 16,
            "revision": 14,
            "string": "2.16.14"
        },
        "group_names": [
            "dev"
        ],
        "groups": {
            "all": [
                "kvm11",
                "kvm12",
                "kvm15",
                "kvm13",
                "kvm14"
            ],
            "balancers": [
                "kvm15"
            ],
            "dev": [
                "kvm11"
            ],
            "prod": [
                "kvm13",
                "kvm14"
            ],
            "test": [
                "kvm12"
            ],
            "ungrouped": [],
            "webservers": [
                "kvm13",
                "kvm14"
            ]
        },
        "inventory_dir": "/home/ceuser/10927001",
        "inventory_file": "/home/ceuser/10927001/inventory",
        "inventory_hostname": "kvm11",
        "inventory_hostname_short": "kvm11",
```

```
        "omit": "
            __omit_place_holder__a4fe4b9c18460a4cd43dde6435869b5ceaf22319
            ",
56      "playbook_dir": "/home/ceuser/ansible_pb"
    }
58  }
```

12.5 setup 輸出 facts 變數

1. setup 模組可以列印所有 ansible_facts 變數內容。

```
ceuser@deyu:/ansible_pb$ ansible dev -m setup
```

2. setup 模組列印 ansible_facts 變數，增加參數 filter=ansible_default_ipv4，取得其內容。

```
1   ceuser@deyu:/ansible_pb$ ansible dev -m setup -a "filter=
        ansible_default_ipv4"
    kvm11 | SUCCESS => {
3       "ansible_facts": {
            "ansible_default_ipv4": {
5               "address": "192.168.122.11",
                "alias": "enp1s0",
7               "broadcast": "192.168.122.255",
                "gateway": "192.168.122.1",
9               "interface": "enp1s0",
                "macaddress": "52:54:00:46:94:e7",
11              "mtu": 1500,
                "netmask": "255.255.255.0",
13              "network": "192.168.122.0",
                "prefix": "24",
15              "type": "ether"
            },
17          "discovered_interpreter_python": "/usr/bin/python3"
        },
19      "changed": false
    }
```

3. setup 模組列印 ansible_facts 變數，增加參數 filter=ansible_fqdn，取得其內容。

```
ceuser@deyu:/ansible_pb$ ansible dev -m setup -a "filter=ansible_fqdn"
2   kvm11 | SUCCESS => {
```

```
    "ansible_facts": {
        "ansible_fqdn": "kvm11.deyu.wang",
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false
}
```

4. setup 模組列印 ansible_facts 變數，增加參數 filter=ansible_hostname，取得其內容。

```
ceuser@deyu:/ansible_pb$ ansible dev -m setup -a "filter=
    ansible_hostname"
kvm11 | SUCCESS => {
    "ansible_facts": {
        "ansible_hostname": "kvm11",
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false
}
```

12.6 產生 hosts 檔案

1. 下載 hosts.j2 樣板檔。

2. host2.j2 格式爲 IP FQDN HOSTNAME (FQDN, Fully qualified domain name, 完整網域名稱)，修改 host2.j2 產生所有 inventroy hosts。

```
ceuser@deyu:/ansible_pb$ vi hosts.j2
ceuser@deyu:/ansible_pb$ cat hosts.j2
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.
    localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain

{% for host in groups['all'] %}
{{ hostvars[host]['ansible_default_ipv4']['address'] }} {{ hostvars[host
    ]
['ansible_fqdn'] }} {{ hostvars[host]['ansible_hostname'] }}
{% endfor %}
```

3. 查詢 template 模組文件的範例。

```
ceuser@deyu:~/ansible_pb$ ansible-doc template | grep ^EX -A8
EXAMPLES:
```

```
3
- name: Template a file to /etc/file.conf
5   ansible.builtin.template:
      src: /mytemplates/foo.j2
7     dest: /etc/file.conf
      owner: bin
9     group: wheel
      mode: '0644'
```

4. 建立 playbook 腳本 hosts.yml，當主機在 dev 群組時，使用 hosts.j2 樣板產生 /etc/myhosts。

```
ceuser@deyu:/ansible_pb$ vi hosts.yml
2 ceuser@deyu:/ansible_pb$ cat hosts.yml
---
4 - name: Create /etc/myhosts
    hosts: all
6   tasks:
    - name: Template a file to /etc/myhosts
8     ansible.builtin.template:
        src: /home/ceuser/ansible_pb/hosts.j2
10      dest: /etc/myhosts
      when: '"dev" in group_names'
```

12.7 測試 hosts.yml

1. ansible-playbook 執行 hosts.yml 腳本，成功在 dev 主機 kvm11 產生 /etc/myhosts。

```
1 ceuser@deyu:/ansible_pb$ ansible-playbook hosts.yml

3 PLAY [Create /etc/myhosts]
    ****************************************************

5 TASK [Gathering Facts]
    ************************************************************
ok: [kvm15]
7 ok: [kvm12]
ok: [kvm11]
9 ok: [kvm13]
ok: [kvm14]
11
TASK [Template a file to /etc/myhosts]
    ****************************************
13 skipping: [kvm12]
skipping: [kvm15]
```

```
skipping: [kvm13]
skipping: [kvm14]
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm11 : ok=2 changed=1 unreachable=0 failed=0 skipped=0 rescued=0
    ignored=0
kvm12 : ok=1 changed=0 unreachable=0 failed=0 skipped=1 rescued=0
    ignored=0
kvm13 : ok=1 changed=0 unreachable=0 failed=0 skipped=1 rescued=0
    ignored=0
kvm14 : ok=1 changed=0 unreachable=0 failed=0 skipped=1 rescued=0
    ignored=0
kvm15 : ok=1 changed=0 unreachable=0 failed=0 skipped=1 rescued=0
    ignored=0
```

2. 查詢 kvm11 的 /etc/myhosts，群組所有主機皆已依格式列出。

```
ceuser@deyu:/ansible_pb$ ansible dev -m shell -a "cat /etc/myhosts"
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.
    localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain

192.168.122.11 kvm11.deyu.wang kvm11
192.168.122.12 kvm12.deyu.wang kvm12
192.168.122.15 kvm15.deyu.wang kvm15
192.168.122.13 kvm13.deyu.wang kvm13
192.168.122.14 kvm14.deyu.wang kvm14
```

12.8 實機練習

1. 下載 hosts.j2 樣板檔。
2. 修改 hosts.j2
   (a) 增加所有受管主機名稱與 IP 對應
   (b) 對應格式爲：IPV4 FQDN inventory_hostname
   (c) 格式中的三項都使用 ansible 變數取得
3. 撰寫 yaml 腳本 hosts1.yml，要求如下：
   (a) 對所有受管主機執行任務。

(b) 只有 test 主機時，使用 hosts.j2 樣板產生 /opt/myhosts。

4. 修改 hosts.j2 存成 hosts2.j2

(a) for 迴圈中增加 if 判斷

(b) 如果是 prod 主機，則輸出 IP FQDN prod (prod 是字串)，結果如下：

```
192.168.122.11 kvm11.deyu.wang kvm11
192.168.122.12 kvm12.deyu.wang kvm12
192.168.122.15 kvm15.deyu.wang kvm15
192.168.122.13 kvm13.deyu.wang prod
192.168.122.14 kvm14.deyu.wang prod
```

5. 修改 hosts1.yml 存成 hosts2.yml

(a) 使用 hosts2.j2

(b) 目標檔案改成 /opt/myhosts2。

Chapter 13

Create Roles

13.1 項目結構

1. role 必須至少包含以下目錄之一，但最好排除任何未使用的目錄。
   (a) 任務 (tasks)：role 要執行的任務列表。
   (b) 善後程序 (handlers)：該角色甚至該角色之外的任何地方都可以使用這些善後程序。
   (c) 預設 (defaults)：role 預設的變數。
   (d) vars：role 的其他變數。
   (e) 檔案 (files)：可以透過此角色部署的檔案。
   (f) 樣板 (templates)：可以透過此角色部署的樣板。
   (g) meta：爲此角色定義一些 meta data。
2. roles 目錄結構範例：roles 目錄下有 common 及 webservers 兩個 roles。

```
roles/
    common/
        tasks/
        handlers/
        files/
        templates/
        vars/
        defaults/
        meta/
    webservers/
        tasks/
        defaults/
        meta/
```

3. 使用時，每個目錄必須包含一個 main.yml 檔案。

13.2 初始化角色 apache

1. 查看 ansible.cfg 配置檔的 roles_path，是否已設定 /home/ceuser/ansible_pb/roles 目錄。

```
ceuser@deyu:~/ansible_pb$ vi ansible.cfg
ceuser@deyu:~/ansible_pb$ grep ^roles_path ansible.cfg
roles_path    = /home/ceuser/ansible_pb/roles
```

2. 建立 roles 目錄。

```
ceuser@deyu:~/ansible_pb$ mkdir roles/
```

3. 進入 roles 目錄。

```
ceuser@deyu:~/ansible_pb$ cd roles/
ceuser@deyu:~/ansible_pb/roles$ pwd
/home/ceuser/ansible_pb/roles
```

4. ansible-galaxy 初使化 apache 角色。

```
ceuser@deyu:~/ansible_pb/roles$ ansible-galaxy role init apache
- Role apache was created successfully
```

5. 查看 apache 目錄，已建立角色需要的目錄及檔案。

```
ceuser@deyu:~/ansible_pb/roles$ tree apache/
apache/
    defaults
        main.yml
    files
    handlers
        main.yml
    meta
        main.yml
    README.md
    tasks
        main.yml
    templates
    tests
        inventory
        test.yml
```

```
    vars
18      main.yml

20  9 directories, 8 files
```

6. 回到 ansible 目錄，查看角色，列出 apache 角色。

```
ceuser@deyu:~/ansible_pb$ cd ..
2  ceuser@deyu:~/ansible_pb$ ansible-galaxy role list
   # /home/ceuser/ansible_pb/roles
4  - phpinfo, (unknown version)
   - balancer, (unknown version)
6  - apache, (unknown version)
   # /usr/share/ansible/roles
8  # /etc/ansible/roles
```

13.3 編輯角色 apache 任務

1. 編輯任務主檔 main.yml，新增任務：

   (a) 安裝最新版本的 httpd

   (b) 設定 htppd 開機啓動且立即啓動

   (c) 防火牆開放 http 服務

```
ceuser@deyu:~/ansible_pb$ vi roles/apache/tasks/main.yml
2  ceuser@deyu:~/ansible_pb$ cat roles/apache/tasks/main.yml
   ---
4  - name: install the latest version of Apache
     ansible.builtin.dnf:
6      name: httpd
       state: latest
8
   - name: Start service httpd
10   ansible.builtin.service:
       name: httpd
12     state: started
       enabled: yes
14
   - name: Enable service firewalld
16   ansible.builtin.service:
       name: firewalld
18     enabled: yes
       state: started
20
```

```
- name: firewalld add http service
  ansible.builtin.firewalld:
    service: http
    permanent: yes
    state: enabled
    immediate: yes
```

13.4 編輯角色 apache 樣板

1. 在樣板目錄 templates 新增樣板檔案 index.html.j2，其中變數 ansible_fqdn

```
ceuser@deyu:~/ansible_pb$ vi roles/apache/templates/index.html.j2
ceuser@deyu:~/ansible_pb$ cat roles/apache/templates/index.html.j2
Welcome to {{ ansible_fqdn }} on {{ ansible_default_ipv4.address }}
```

2. 其中主機名稱及 IP 可用 ansible -m setup 查詢，過濾字串 ansible_fqdn。

```
ceuser@deyu:~/ansible_pb$ ansible kvm14 -m setup | grep fqdn
        "ansible_fqdn": "kvm14.deyu.wang",
ceuser@deyu:~/ansible_pb$ ansible kvm14 -m setup -a "filter=ansible_fqdn
    "
kvm14 | SUCCESS => {
    "ansible_facts": {
        "ansible_fqdn": "kvm14.deyu.wang",
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

3. 其中主機名稱及 IP 可用 ansible -m setup 查詢，過濾字串 default_ipv4。

```
ceuser@deyu:~/ansible_pb$ ansible kvm14 -m setup | grep ipv4
        "ansible_all_ipv4_addresses": [
        "ansible_default_ipv4": {
                "tx_checksum_ipv4": "off [fixed]",
            "ipv4": {
                "tx_checksum_ipv4": "off [fixed]",
            "ipv4": {
            "ipv4": [
ceuser@ip123:~/10927000$ ansible kvm14 -m setup -a "filter=
    ansible_default_ipv4"kvm14 | SUCCESS => {
    "ansible_facts": {
        "ansible_default_ipv4": {
            "address": "192.168.122.14",
```

```
            "alias": "enp1s0",
            "broadcast": "192.168.122.255",
            "gateway": "192.168.122.1",
            "interface": "enp1s0",
            "macaddress": "52:54:00:87:2e:b2",
            "mtu": 1500,
            "netmask": "255.255.255.0",
            "network": "192.168.122.0",
            "prefix": "24",
            "type": "ether"
        },
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

4. 在任務主檔 main.yml 新增任務，將樣板 index.html.j2 複製到 /var/www/html/index.html。

```
ceuser@deyu:~/ansible_pb$ vi roles/apache/tasks/main.yml
ceuser@deyu:~/ansible_pb$ cat roles/apache/tasks/main.yml
---
- name: install the latest version of Apache
  ansible.builtin.dnf:
    name: httpd
    state: latest
- name: Start service httpd
  ansible.builtin.service:
    name: httpd
    state: started
    enabled: yes

- name: Enable service firewalld
  ansible.builtin.service:
    name: firewalld
    enabled: yes
    state: started

- name: firewalld add http service
  ansible.builtin.firewalld:
    service: http
    permanent: yes
    state: enabled
    immediate: yes

- name: Template a file
  ansible.builtin.template:
    src: index.html.j2
    dest: /var/www/html/index.html
```

5. 再查看 roles/apache 目錄，樣板目錄 templates 下有 index.html.j2：

```
ceuser@deyu:~/ansible_pb$ tree roles/apache/
roles/apache/
    defaults
        main.yml
    files
    handlers
        main.yml
    meta
        main.yml
    README.md
    tasks
        main.yml
    templates
        index.html.j2
    tests
        inventory
        test.yml
    vars
        main.yml

9 directories, 9 files
```

13.5 實機練習

1. 初使化一個 nfs 角色。
2. 編輯 nfs 角色，任務如下：
   (a) 安裝 nfs-utils 套件
   (b) 設定開機啓動 nfs-server 服務，且現在啓動。
   (c) 啓用 firewalld
   (d) firewalld 服務開放 nfs, rpc-bind, mountd 等服務，開啓三個服務要使用 loop 迴圈。

```
  ansible.builtin.firewalld:
    service: "{{ item }}"
    permanent: yes
    state: enabled
    immediate: yes
  loop:
    - nfs
    - rpc-bind
    - mountd
```

(e) 建立分享目錄 /public，提示：file 模組範例。

```
- name: Create a directory if it does not exist
  ansible.builtin.file:
    path: /etc/some_directory
    state: directory
    mode: '0755'
```

(f) 新增樣板 exports.j2，並複製到 /etc/exports。

i. 樣板檔案 exports.j2 內容如下，其中分享網段 xxx.xxx.xxx.0 由變數 ansible_default_ipv4.network 取得。

```
/nfsdir xxx.xxx.xxx.0/24(rw,sync)
```

ii. 目標位置：/etc/exports

iii. 複製樣板檔案後，必須重啓 nfs-server 服務才會生效，所以 nfs-server 服務的狀態要使用"restarted"，且 templates 模組也必須放在 nfs-server 重啓任務之前。

3. 建立 nfs.yml 腳本，對 dev 主機執行任務，要求如下：

(a) 使用 nfs 角色

(b) 完成任務後，showmount 查詢 dev 主機是否分享 /public 目錄？

(c) 若沒有 showmount 命令，請自行安裝 nfs-utils 套件。

```
showmount -e kvm11
```

Chapter 14

Use Roles

14.1 使用 apache 角色

1. 在 ansible 目錄下新增 roles.yml，在 webservers 群組主機使用角色 apache，安裝 httpd 網頁服務。

```
ceuser@deyu:~/ansible_pb$ vi roles.yml
ceuser@deyu:~/ansible_pb$ cat roles.yml
---
- name: install httpd
  hosts: webservers
  roles:
  - apache
```

2. ansible-playbook 執行腳本 roles.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook roles.yml

PLAY [install httpd]
    ************************************************************

TASK [Gathering Facts]
    ************************************************************
ok: [kvm14]
ok: [kvm13]

TASK [apache : install the latest version of Apache]
    ***************************
ok: [kvm14]
ok: [kvm13]

TASK [apache : Start service httpd]
    *****************************************
ok: [kvm14]
ok: [kvm13]
```

```
17 TASK [apache : firewalld]
       ***************************************************
   ok: [kvm13]
19 ok: [kvm14]

21 TASK [apache : Template a file]
       ********************************************
   ok: [kvm13]
23 ok: [kvm14]

25 PLAY RECAP
       *********************************************************************

   kvm13 : ok=5 changed=0  unreachable=0  failed=0  skipped=0  rescued=0
       ignored=0
27 kvm14 : ok=5 changed=0  unreachable=0  failed=0  skipped=0  rescued=0
       ignored=0
```

14.2 使用 haproxy 角色

1. 在 roles.yml 新增任務，在 balancers 群組主機使用角色 balancer 安裝負載平衡代理伺服器 haproxy。

```
1 ceuser@deyu:~/ansible_pb$ cat roles.yml
  ---
3 - name: install httpd
    hosts: webservers
5   roles:
    - apache
7
  - name: use balancer
9   hosts: balancers
    roles:
11  - balancer
```

2. ansible-playbook 執行腳本 roles.yml。

```
1 ceuser@deyu:~/ansible_pb$ ansible-playbook roles.yml
  TASK [Gathering Facts]
       ***********************************************************
3 ok: [kvm13]
  ok: [kvm14]
5
  TASK [apache : install the latest version of Apache]
       ***************************
```

```
ok: [kvm14]
ok: [kvm13]

TASK [apache : Start service httpd]
    ********************************************
ok: [kvm13]
ok: [kvm14]

TASK [apache : Enable service firewalld]
    ************************************
ok: [kvm13]
ok: [kvm14]

TASK [apache : firewalld add http service]
    ***********************************
ok: [kvm14]
ok: [kvm13]

TASK [apache : Template a file]
    ************************************************
ok: [kvm13]
ok: [kvm14]

PLAY [use balancer]
    ************************************************************

TASK [Gathering Facts]
    ************************************************************
ok: [kvm15]

TASK [balancer : Download and install haproxy]
    *********************************
changed: [kvm15]

TASK [balancer : Configure the haproxy cnf file with hosts]
    *********************
changed: [kvm15]

TASK [balancer : Start the haproxy service]
    ***********************************
changed: [kvm15]

RUNNING HANDLER [balancer : restart haproxy]
    ***********************************
changed: [kvm15]

PLAY RECAP
    *********************************************************************

kvm13 : ok=6 changed=0  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm14 : ok=6 changed=0  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm15 : ok=5 changed=4  unreachable=0  failed=0  skipped=0  rescued=0
```

```
    ignored=0
```

3. kvm15 的 selinux 沒處理，還不能連線。

```
ceuser@deyu:~/ansible_pb$ curl http://kvm15/
curl: (7) Failed to connect to kvm15 port 80 after 3 ms: Could not
    connect to server
```

4. 登入 kvm15 測試連線負載平衡器主機 kvm15，第一次回應來自 kvm13，再連線一次，回應來自 kvm14，確定 haproxy 工作正常。

```
ceuser@deyu:~/ansible_pb$ ssh kvm15
Last login: Mon Feb  2 07:53:03 2026 from 192.168.122.1
[ceuser@kvm15 ~]$ curl http://kvm15/
Welcome to kvm13.deyu.wang on 192.168.122.13
[ceuser@kvm15 ~]$ curl http://kvm15/
Welcome to kvm14.deyu.wang on 192.168.122.14
[ceuser@kvm15 ~]$ exit
logout
Connection to kvm15 closed.
```

14.3 使用 phpinfo 角色

1. 再編輯 roles.yml，新增任務 hello php，在 webservers 群組主機使用角色 phpinfo。

```
ceuser@deyu:~/ansible_pb$ cat roles.yml
---
- name: install httpd
  hosts: webservers
  roles:
  - apache

- name: use balancer
  hosts: balancers
  roles:
  - balancer

- name: hello php
  hosts: webservers
  roles:
  - phpinfo
```

2. ansible-playbook 執行 roles.yml 腳本。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook roles.yml

PLAY [install httpd] *********************************************************

TASK [Gathering Facts] *******************************************************
ok: [kvm14]
ok: [kvm13]

TASK [apache : install the latest version of Apache] *************************
ok: [kvm13]
ok: [kvm14]

TASK [apache : Start service httpd] ******************************************
ok: [kvm14]
ok: [kvm13]

TASK [apache : Enable service firewalld] *************************************
ok: [kvm14]
ok: [kvm13]

TASK [apache : firewalld add http service] ***********************************
ok: [kvm14]
ok: [kvm13]

TASK [apache : Template a file] **********************************************
ok: [kvm13]
ok: [kvm14]

PLAY [use balancer] **********************************************************

TASK [Gathering Facts] *******************************************************
ok: [kvm15]

TASK [balancer : Download and install haproxy] *******************************
ok: [kvm15]

TASK [balancer : Configure the haproxy cnf file with hosts] ******************
ok: [kvm15]

TASK [balancer : Start the haproxy service] **********************************
```

```
ok: [kvm15]

PLAY [hello php] ***************************************************************

TASK [Gathering Facts] *********************************************************
ok: [kvm13]
ok: [kvm14]

TASK [phpinfo : generate php info page] ****************************************
changed: [kvm14]
changed: [kvm13]

PLAY RECAP *********************************************************************
kvm13 : ok=8 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
kvm14 : ok=8 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
kvm15 : ok=4 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
```

3. curl 連線 kvm13 的 hello.php，回應" Hello PHP World from kvm13.deyu.wang" 及 php 訊息。

```
ceuser@deyu:~/ansible_pb$ curl -s http://kvm13/hello.php
<?php echo 'Hello PHP World from kvm13.deyu.wang'; ?>
<p>
<?php phpinfo(); ?>
```

4. curl 連線 kvm14 的 hello.php，查看回應訊息。

```
ceuser@deyu:~/ansible_pb$ curl -s http://kvm14/hello.php
<?php echo 'Hello PHP World from kvm14.deyu.wang'; ?>
<p>
<?php phpinfo(); ?>
```

14.4 實機練習

1. 撰寫 yaml 腳本 roles.yml，要求如下：

(a) 在 webservers 群組主機使用角色 apache，安裝 httpd 網頁服務。

(b) 在 balancers 群組主機使用角色 balancer 安裝負載平衡代理伺服器 haproxy。

(c) 在 webservers 群組主機使用角色 phpinfo。

Chapter 15

Shell 模組

15.1 前言

1. Shell 模組可在受管主機執行 shell 命令。
2. Shell 模組預設使用 /bin/sh 執行 shell 命令，但也可以由 executable 參數指定其他的 shell，例如 /bin/bash。

15.2 模組文件

1. 查詢 shell 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc shell
```

2. 查詢 shell 模組文件的範例，執行腳本並將輸出導向到檔案。

```
ceuser@deyu:~/ansible_pb$ ansible-doc shell | grep ^EX -A3
EXAMPLES:

- name: Execute the command in remote shell; stdout goes to the
    specified file
 on the remote.
  ansible.builtin.shell: somescript.sh >> somelog.txt
```

3. 切換到 somedir 目錄，執行腳本並將輸出導向到檔案。

```
- name: Change the working directory to somedir/ before executing the
    command.
  ansible.builtin.shell: somescript.sh >> somelog.txt
```

```
  args:
    chdir: somedir/
```

4. 切換到 somedir 目錄，只在 somelog.txt 不存在時，才執行腳本並將輸出導向到檔案 somelog.txt。

```
# You can also use the 'args' form to provide the options.
- name: This command will change the working directory to somedir/ and
    will
 only run when somedir/somelog.txt doesn't exist.
  ansible.builtin.shell: somescript.sh >> somelog.txt
  args:
    chdir: somedir/
    creates: somelog.txt
```

5. 也可以用 cmd 參數來指定執行命令。

```
# You can also use the 'cmd' parameter instead of free form format.
- name: This command will change the working directory to somedir/
  ansible.builtin.shell:
    cmd: ls -l | grep log
    chdir: somedir/
```

6. /bin/sh 不支援 cat < /tmp/*txt，參數 executable 指定使用 /bin/bash。

```
- name: Run a command that uses non-posix shell-isms (in this example /
    bin/sh
 doesn't handle redirection and wildcards together but bash does)
  ansible.builtin.shell: cat < /tmp/*txt
  args:
    executable: /bin/bash
```

7. 使用樣板變數。

```
- name: Run a command using a templated variable (always use quote
    filter to
 avoid injection)
  ansible.builtin.shell: cat {{ myfile|quote }}
```

8. expect 互動程式。

```
# You can use shell to run other executables to perform actions inline
- name: Run expect to wait for a successful PXE boot via out-of-band
    CIMC
  ansible.builtin.shell: |
    set timeout 300
    spawn ssh admin@{{ cimc_host }}

    expect "password:"
    send "{{ cimc_password }}\n"

    expect "\n{{ cimc_name }}"
    send "connect host\n"

    expect "pxeboot.n12"
    send "\n"
    exit 0
  args:
    executable: /usr/bin/expect
  delegate_to: localhost
```

9. 關閉警告訊息。

```
# Disabling warnings
- name: Using curl to connect to a host via SOCKS proxy (unsupported in
    uri).
 Ordinarily this would throw a warning.
  ansible.builtin.shell: curl --socks5 localhost:9000 http://www.ansible
      .com
  args:
    warn: no
```

15.3 修改檔案-單一群組

1. 撰寫 shell1.yml 腳本，對 dev 主機的 /etc/issue，修改成特定的字串。

```
ceuser@deyu:~/ansible_pb$ vim shell1.yml
ceuser@deyu:~/ansible_pb$ cat shell1.yml
---
- hosts: all
  tasks:
    - ansible.builtin.shell:
        echo 'Development {{inventory_hostname}}' > /etc/issue
      when: '"dev" in group_names'
```

2. ansible-playbook 執行腳本 shell1.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook shell1.yml

PLAY [all]
    ********************************************************************


TASK [Gathering Facts]
    ***********************************************************
ok: [kvm15]
ok: [kvm11]
ok: [kvm14]
ok: [kvm13]
ok: [kvm12]

TASK [shell]
    ******************************************************************
skipping: [kvm12]
skipping: [kvm15]
skipping: [kvm13]
skipping: [kvm14]
changed: [kvm11]

PLAY RECAP
    ********************************************************************

kvm11 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm12 : ok=1 changed=0  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm13 : ok=1 changed=0  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm14 : ok=1 changed=0  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm15 : ok=1 changed=0  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. ansible 使用 shell 模組查看主機 dev 的 /etc/issue 檔案內容，符合要求。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "cat /etc/issue"
kvm11 | CHANGED | rc=0 >>
Development kvm11
```

15.4 修改檔案-不同群組不同內容

1. 修改 shelli1.yml 腳本成 shell2.yml，dev, test, prod 群組主機的 /etc/issue，寫入字串不同。

```
ceuser@deyu:~/ansible_pb$ vim shell2.yml
ceuser@deyu:~/ansible_pb$ cat shell2.yml
---
- hosts: all
  tasks:
    - ansible.builtin.shell:
        echo 'Development {{inventory_hostname}}' > /etc/issue
      when: '"dev" in group_names'
    - ansible.builtin.shell:
        echo 'Test {{inventory_hostname}}' > /etc/issue
      when: '"test" in group_names'
    - ansible.builtin.shell:
        echo 'Production {{inventory_hostname}}' > /etc/issue
      when: '"prod" in group_names'
```

2. ansible-playbook 執行腳本 shell2.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook shell2.yml

PLAY [all] ********************************************************************


TASK [Gathering Facts] ********************************************************
ok: [kvm14]
ok: [kvm15]
ok: [kvm11]
ok: [kvm12]
ok: [kvm13]

TASK [shell] ******************************************************************
skipping: [kvm12]
skipping: [kvm15]
skipping: [kvm13]
skipping: [kvm14]
changed: [kvm11]

TASK [shell] ******************************************************************
skipping: [kvm11]
skipping: [kvm15]
skipping: [kvm13]
skipping: [kvm14]
```

```
changed: [kvm12]

TASK [shell]
    ********************************************************************
skipping: [kvm11]
skipping: [kvm12]
skipping: [kvm15]
changed: [kvm13]
changed: [kvm14]

PLAY RECAP
    ********************************************************************

kvm11 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm12 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm13 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm14 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
kvm15 : ok=1 changed=0  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. ansible 使用 shell 模組查看主機 dev, test, prod 等主機的 /etc/issue 檔案內容，符合要求。

```
ceuser@deyu:~/ansible_pb$ ansible all -m shell -a "cat /etc/issue"
kvm15 | CHANGED | rc=0 >>
\S
Kernel \r on an \m
kvm12 | CHANGED | rc=0 >>
Test kvm12
kvm14 | CHANGED | rc=0 >>
Production kvm14
kvm13 | CHANGED | rc=0 >>
Production kvm13
kvm11 | CHANGED | rc=0 >>
Development kvm11
```

15.5 實機練習

1. 撰寫 YAML 腳本 sh1.yml，對主機 dev 執行任務，要求如下：

   (a) 使用 shell 模組，切換到 dir 目錄，dir 變數由 ansible-playbook 命令列外加參數決定。

(b) ansible-playbook 執行時變數 dir 用 /usr/local 測試看看。

(c) 執行 grep ID /etc/os-release，並導向到 os.txt，若 os.txt 存在，則覆蓋其內容。

2. 修改腳本 sh1.yml 爲 sh2.yml，對 prod 及 balancers 主機使用 shell 模組：

(a) 切換到 dir 目錄，dir 變數由 ansible-playbook 命令列外加參數決定。

(b) ansible-playbook 執行時變數 dir 用 /tmp 測試看看。

(c) 執行 grep ID /etc/os-release，並導向到 os.txt，若 os.txt 存在，則覆蓋其內容。

(d) 但只有 /dev/vda4 存在的主機才執行 shell 任務。

(e) 提示：使用變數 ansible_devices.vda.....，變數名稱用 setup 模組查詢。

(f) 提示：判斷 ansible_devices.vda..... is defined 或 is not defined。

3. 撰寫 shell1a.yml，對所有主機執行修改任務，要求如下：

(a) 使用 shell 模組，當主機爲 dev 時才將其 /etc/issue 內容修改爲 Development IPV4。

(b) 其中 IPV4 以變數 ansible_default_ipv4..... 取得主機 ipv4 的 IP。

4. 撰寫 shell2a.yml，對所有主機執行修改任務，要求如下：

(a) 使用 shell 模組在 dev, test, prod 群組主機的 /etc/domain，寫入不同字串 XXXX YYYY。

(b) 其中 XXXX 在 dev, test, prod 分別爲 Development, Test, Production。

(c) YYYY 爲網域名稱，使用變數 ansible_domain 取得。

5. 修改腳本 sh2.yml 爲 sh3.yml，對 prod 及 balancers 主機使用 shell 模組：

(a) 切換到 dir 目錄，dir 變數由 ansible-playbook 命令列外加參數決定。

(b) ansible-playbook 執行時變數 dir 用 /opt 測試看看。

(c) /dev/vda4 存在的主機，執行 shell 任務 grep ID /etc/os-release，並導向到 id.txt。若 id.txt 存在，則覆蓋其內容。

(d) /dev/vda4 不存在的主機，執行 shell 任務 grep ID /etc/profile，並導向到 id.txt。若 id.txt 存在，則覆蓋其內容。

(e) 提示：使用變數 ansible_devices.vda.....，變數名稱用 setup 模組查詢。

Chapter 16

硬碟分割掛載

16.1 相關模組

1. parted 分割硬碟，parted 模組文件，fs_type: ext4 可以立即將分割區格式化爲 ext4，不用再使用 filesystem 模組格式化分割區。

```
ceuser@deyu:~/ansible_pb$ ansible-doc parted| grep ^EX -A36
EXAMPLES:

- name: Create a new ext4 primary partition
  community.general.parted:
    device: /dev/sdb
    number: 1
    state: present
    fs_type: ext4

- name: Remove partition number 1
  community.general.parted:
    device: /dev/sdb
    number: 1
    state: absent

- name: Create a new primary partition with a size of 1GiB
  community.general.parted:
    device: /dev/sdb
    number: 1
    state: present
    part_end: 1GiB

- name: Create a new primary partition for LVM
  community.general.parted:
    device: /dev/sdb
    number: 2
    flags: [ lvm ]
    state: present
    part_start: 1GiB
```

```
- name: Create a new primary partition with a size of 1GiB at disk's end
  community.general.parted:
    device: /dev/sdb
    number: 3
    state: present
    fs_type: ext3
    part_start: -1GiB
```

2. filesystem 檔案系統管理，filesystem 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc filesystem | grep ^EX -A30
EXAMPLES:

- name: Create a ext2 filesystem on /dev/sdb1
  community.general.filesystem:
    fstype: ext2
    dev: /dev/sdb1

- name: Create a ext4 filesystem on /dev/sdb1 and check disk blocks
  community.general.filesystem:
    fstype: ext4
    dev: /dev/sdb1
    opts: -cc

- name: Blank filesystem signature on /dev/sdb1
  community.general.filesystem:
    dev: /dev/sdb1
    state: absent

- name: Create a filesystem on top of a regular file
  community.general.filesystem:
    dev: /path/to/disk.img
    fstype: vfat
```

3. 檔案系統掛載管理，mount 模組將掛載要求寫入開機自動掛載表 /etc/fstab 中，其中 state: mounted 指定寫入 /etc/fstab 後立即掛載。

```
ceuser@deyu:~/ansible_pb$ ansible-doc mount | grep ^EX -A86
EXAMPLES:

# Before 2.3, option 'name' was used instead of 'path'
- name: Mount DVD read-only
  ansible.posix.mount:
    path: /mnt/dvd
    src: /dev/sr0
    fstype: iso9660
    opts: ro,noauto
    state: present
```

```
- name: Mount up device by label
  ansible.posix.mount:
    path: /srv/disk
    src: LABEL=SOME_LABEL
    fstype: ext4
    state: present

- name: Mount up device by UUID
  ansible.posix.mount:
    path: /home
    src: UUID=b3e48f45-f933-4c8e-a700-22a159ec9077
    fstype: xfs
    opts: noatime
    state: present

- name: Unmount a mounted volume
  ansible.posix.mount:
    path: /tmp/mnt-pnt
    state: unmounted

- name: Remount a mounted volume
  ansible.posix.mount:
    path: /tmp/mnt-pnt
    state: remounted

# The following will not save changes to fstab, and only be temporary
    until
# a reboot, or until calling "state: unmounted" followed by "state:
    mounted"
# on the same "path"
- name: Remount a mounted volume and append exec to the existing options
  ansible.posix.mount:
    path: /tmp
    state: remounted
    opts: exec

- name: Mount and bind a volume
  ansible.posix.mount:
    path: /system/new_volume/boot
    src: /boot
    opts: bind
    state: mounted
    fstype: none

- name: Mount an NFS volume
  ansible.posix.mount:
    src: 192.168.1.100:/nfs/ssd/shared_data
    path: /mnt/shared_data
    opts: rw,sync,hard
    state: mounted
    fstype: nfs

- name: Mount NFS volumes with noauto according to boot option
  ansible.posix.mount:
```

```
    src: 192.168.1.100:/nfs/ssd/shared_data
    path: /mnt/shared_data
    opts: rw,sync,hard
    boot: false
    state: mounted
    fstype: nfs

- name: Mount ephemeral SMB volume
  ansible.posix.mount:
    src: //192.168.1.200/share
    path: /mnt/smb_share
    opts: "rw,vers=3,file_mode=0600,dir_mode=0700,dom={{ ad_domain }},
        username={{ ad_username }},password={{ ad_password }}"
    fstype: cifs
    state: ephemeral
```

16.2 parted 模組 - 新增分割區

1. parted 模組可分割硬碟，查看 parted 模組的範例，其中參數 part_start 分割起點預設是 0% 碟碟最前端，另一參數 part_end 分割終點預設是 100% 硬碟最後端。

```
ceuser@deyu:~/ansible_pb$ ansible-doc parted | grep LVM -A6
- name: Create a new primary partition for LVM
  community.general.parted:
    device: /dev/sdb
    number: 2
    flags: [ lvm ]
    state: present
    part_start: 1GiB
```

2. 查看 balancers 主機有 vda vdb vdc 三顆硬碟。

```
ceuser@deyu:~/ansible_pb$ ansible balancers -m shell -a "lsblk"
kvm15 | CHANGED | rc=0 >>
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
vda                 252:0    0    6G  0 disk
  vda1              252:1    0    1M  0 part
  vda2              252:2    0  400M  0 part /boot
  vda3              252:3    0  3.4G  0 part
    vg_kvm15-root   253:0    0  3.3G  0 lvm  /
    vg_kvm15-swap   253:1    0   96M  0 lvm  [SWAP]
vdb                 252:16   0  2.5G  0 disk
vdc                 252:32   0  1.5G  0 disk
```

3. 建立 partvg.yml 腳本，新增 balancers 主機硬碟 /dev/vdb1，分割終點是 1.2GMiB。

```
ceuser@deyu:~/ansible_pb$ vi partvg.yml
ceuser@deyu:~/ansible_pb$ cat partvg.yml
---
- hosts: balancers
  tasks:
  - name: Create a new primary partition with a size of 1.2GiB
    community.general.parted:
      device: /dev/vdb
      number: 1
      state: present
      part_end: 1.2GiB
      fs_type: ext4
```

4. ansible-playbook 執行 partvg.yml 腳本，成功分割 vdb1。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook partvg.yml

PLAY [balancers]
    ****************************************************************

TASK [Gathering Facts]
    ***********************************************************
ok: [kvm15]

TASK [Create a new primary partition with a size of 1.2GiB]
    ***********************
changed: [kvm15]

PLAY RECAP
    *********************************************************************

kvm15  : ok=2  changed=1  unreachable=0  failed=0  skipped=0  rescued=0
     ignored=0
```

5. 查看 balancers 主機硬碟 vdb 出現 vdb1 分割區，大小爲 1.2GiB。

```
ceuser@deyu:~/ansible_pb$ ansible balancers -m shell -a "fdisk -l /dev/
    vdb"
kvm15 | CHANGED | rc=0 >>
Disk /dev/vdb: 2.5 GiB, 2684354560 bytes, 5242880 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xce6507ef
```

```
10 Device     Boot Start     End Sectors  Size Id Type
   /dev/vdb1           1 2516581 2516581  1.2G 83 Linux
```

16.3 filesystem 模組 - 檔案系統格式化

1. 修改 parted.yml 爲 partedfs.yml，增加任務將 /dev/vdb1 格式化爲 ext4 檔案系統。

```
1  ceuser@deyu:~/ansible_pb$ vi partedfs.yml
   ---
3  - hosts: balancers
     tasks:
5    - name: Create a new primary partition with a size of 1.2GiB
       community.general.parted:
7        device: /dev/vdb
         number: 1
9        state: present
         part_end: 1.2GiB
11       fs_type: ext4

13   - name: Create a ext4
       community.general.filesystem:
15       fstype: ext4
         dev: /dev/vdb1
```

2. ansible-playbook 執行 partedfs.yml 腳本。

```
   ceuser@deyu:~/ansible_pb$ ansible-playbook partedfs.yml
2
   PLAY [balancers]
       ********************************************************************
4
   TASK [Gathering Facts]
       *********************************************************
6  ok: [kvm15]

8  TASK [Create a new primary partition with a size of 2GiB]
       **********************
   ok: [kvm15]
10
   TASK [Create a ext4]
       ************************************************************
12 changed: [kvm15]
```

```
14 PLAY RECAP
    *********************************************************************

kvm15 : ok=3  changed=1  unreachable=0 failed=0  skipped=0  rescued=0
    ignored=0
```

3. 查看 /dev/vdb1 已格式化爲 ext4 檔案系統。

```
1 ceuser@deyu:~/ansible_pb$ ansible balancers -m shell -a "fsck -N /dev/
    vdb1"
kvm15 | CHANGED | rc=0 >>
3 fsck from util-linux 2.40.2
[/usr/sbin/fsck.ext4 (1) -- /dev/vdb1] fsck.ext4 /dev/vdb1
```

16.4 mount 模組 - 掛載檔案系統

1. 修改 partedfs.yml 爲 partedfsmnt.yml，新增目錄 /newpart，在開機掛載表中新增 /dev/vdb1 掛載在 /newpart，且立即掛載。

```
ceuser@deyu:~/ansible_pb$ vi partedfsmnt.yml
2 ceuser@deyu:~/ansible_pb$ cat partedfsmnt.yml
---
4 - hosts: balancers
  tasks:
6   - name: Create a new primary partition with a size of 1.2GiB
      community.general.parted:
8       device: /dev/vdb
        number: 1
10      state: present
        part_end: 1.2GiB
12      fs_type: ext4

14  - name: Create a ext4
      community.general.filesystem:
16      fstype: ext4
        dev: /dev/vdb1
18
  - name: Create a directory if it does not exist
20    ansible.builtin.file:
        path: /newpart
22      state: directory

24  - name: Mount up device
      ansible.posix.mount:
26      path: /newpart
        src: /dev/vdb1
```

```
28        fstype: ext4
          state: mounted
```

2. ansible-playbook 成功執行 partvg.yml 腳本。

```
1  ceuser@deyu:~/ansible_pb$ ansible-playbook partedfsmnt.yml

3  PLAY [balancers]
       ****************************************************************

5  TASK [Gathering Facts]
       *********************************************************
   ok: [kvm15]
7
   TASK [Create a new primary partition with a size of 1.2GiB]
       ********************
9  ok: [kvm15]

11 TASK [Create a ext4]
       ************************************************************
   ok: [kvm15]
13
   TASK [Create a directory if it does not exist]
       *********************************
15 changed: [kvm15]

17 TASK [Mount up device]
       ***********************************************************
   changed: [kvm15]
19
   PLAY RECAP
       *********************************************************************

21 kvm15  : ok=5  changed=2  unreachable=0  failed=0  skipped=0  rescued=0
        ignored=0
```

3. 查看 kvm15 /etc/fstab 新增一行將 /dev/vdb1 掛載在 /newpart。

```
1  ceuser@deyu:~/ansible_pb$ ansible balancers -m shell -a "grep vdb1 /etc/
       fstab"
   kvm15 | CHANGED | rc=0 >>
3  /dev/vdb1 /newpart ext4 defaults 0 0
```

4. 查看 kvm15 掛載狀況，/dev/vdb1 已掛載在 /newpart，格式爲 ext4。

```
1  ceuser@deyu:~/ansible_pb$ ansible balancers -m shell -a "df -Th /dev/
       vdb1"
```

```
kvm15 | CHANGED | rc=0 >>
Filesystem     Type  Size  Used Avail Use% Mounted on
/dev/vdb1      ext4  1.2G   24K  1.1G   1% /newpart
```

16.5 解除掛載並移除分割區

1. 新增檔案系統並掛載的順序是：

   (a) parted 模組分割硬碟

   (b) filesystem 模組格式化分割區

   (c) file 模組建立掛載目錄

   (d) mount 模組設定開機掛載及立即掛載

2. 解除掛載並移除分割區的順序必須反向：

   (a) mount 模組解除掛載

   (b) file 模組刪除掛載目錄

   (c) parted 模組移除分割區

3. 撰寫 rmparted.yml mount 模組的 state 參數 absent 從 /etc/fstab 掛載表中移除並解除掛載 /newpart，若使 unmounted 則只會解除掛載，不會從 /etc/fstab 表中移除，file 模組移除目錄 /newpart，parted 模組移除 /dev/vdb1，

```
ceuser@deyu:~/ansible_pb$ vi rmparted.yml
---
- hosts: balancers
  tasks:
  - name: unMount device
    ansible.posix.mount:
      path: /newpart
      state: absent

  - name: Remove a directory
    ansible.builtin.file:
      path: /newpart
      state: absent

  - name: Remove a primary partition
    community.general.parted:
      device: /dev/vdb
      number: 1
      state: absent
```

4. ansible-playbook 成功執行 rmparted.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook rmparted.yml

PLAY [balancers]
    ********************************************************************

TASK [Gathering Facts]
    ***********************************************************
ok: [kvm15]

TASK [unMount device]
    ***********************************************************
changed: [kvm15]

TASK [Remove a directory]
    ***********************************************************
ok: [kvm15]

TASK [Remove a primary partition]
    *****************************************
changed: [kvm15]

PLAY RECAP
    ********************************************************************************

kvm15 : ok=4  changed=2  unreachable=0 failed=0  skipped=0  rescued=0
    ignored=0
```

5. 查看 kvm15 掛載狀況，/dev/vdb1 已不存在。

```
ceuser@deyu:~/ansible_pb$ ansible balancers -m shell -a "df -Th /dev/
    vdb1"
kvm15 | FAILED | rc=1 >>
df: /dev/vdb1: No such file or directorynon-zero return code
```

6. 查看 kvm15 /etc/fstab，/dev/vdb1 已不在掛載表中。

```
ceuser@deyu:~/ansible_pb$ ansible balancers -m shell -a "grep vdb1 /etc/
    fstab"
kvm15 | FAILED | rc=1 >>
non-zero return code
```

16.6 實機練習

1. 撰寫 partition.yml，在 balancers 主機進行以下任務：
   (a) 使用 parted 模組，在硬碟 /dev/vdX (看參數) 新增一個編號爲 1 的 primary 分割區。
   (b) 大小 partsize GiB
   (c) 格式化 parttype
   (d) /dev/sdX 永久掛載在 /newpart (寫入 /etc/fstab 表中)。

Chapter 17

LVM 檔案系統

17.1 LVM 相關模組

1. 查詢 LVM 相關模組，其中 lvg 是設置 vg 的模組，lvol 是設置 lv 的模組。

```
ceuser@deyu:~/ansible_pb$ ansible-doc -l | grep LVM
community.general.aix_filesystem                    Configure LVM
    and...
community.general.aix_lvg                           Manage LVM
    volume...
community.general.aix_lvol                          Configure AIX
    LVM...
community.general.lvg                               Configure LVM
    vol...
community.general.lvol                              Configure LVM
    log...
```

2. lvg 邏輯卷冊組管理模組，查看 lvg 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc lvg
```

3. lvol 邏輯卷冊管理模組，查看 lvol 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc lvol
```

17.2 lvg 模組

1. lvg 模組可管理邏輯卷冊組 (volume group)，查看 lvg 模組的範例。

```
ceuser@deyu:~/ansible_pb$ ansible-doc lvg | grep ^EX -A6
EXAMPLES:

- name: Create a volume group on top of /dev/sda1 with physical extent
    size = 32MB
  community.general.lvg:
    vg: vg.services
    pvs: /dev/sda1
    pesize: 32
```

2. 查看 dev 主機 kvm11 有一顆完全沒分割的 vdb 硬碟。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "lsblk"
kvm11 | CHANGED | rc=0 >>
NAME              MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
vda               252:0    0    6G  0 disk
  vda1            252:1    0    1M  0 part
  vda2            252:2    0  400M  0 part /boot
  vda3            252:3    0  3.4G  0 part
    vg_kvm11-root 253:0    0  3.3G  0 lvm  /
    vg_kvm11-swap 253:1    0   96M  0 lvm  [SWAP]
vdb               252:16   0  1.5G  0 disk
```

3. partvg.yml 腳本在 vdb 硬碟分割一個分割區，沒有指定大小、起始或結束位置，表示使用整顆 vdb 硬碟，接著在 vdb1 新增產生 vg1，PE 大小是 8M。

```
ceuser@deyu:~/ansible_pb$ vi partvg.yml
ceuser@deyu:~/ansible_pb$ cat partvg.yml
---
- hosts: dev
  tasks:
  - name: Create a new primary partition for LVM
    community.general.parted:
      device: /dev/vdb
      number: 1
      flags: [ lvm ]
      state: present

  - name: Create a volume group on top of /dev/vdb1
    community.general.lvg:
      vg: vg1
      pvs: /dev/vdb1
      pesize: 8
```

4. ansible-playbook 成功執行 partvg.yml 腳本。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook partvg.yml

PLAY [dev]
    ********************************************************************


TASK [Gathering Facts]
    ***********************************************************
ok: [kvm11]

TASK [Create a new primary partition for LVM]
    *********************************
changed: [kvm11]

TASK [Create a volume group on top of /dev/vdb1]
    ******************************
changed: [kvm11]

PLAY RECAP
    ********************************************************************

kvm11 : ok=3 changed=2  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

5. 查看 kvm11 新增名爲 vg1 的 VG。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "vgs"
kvm11 | CHANGED | rc=0 >>
  VG         #PV #LV #SN Attr   VSize   VFree
  fieldwork    1   0   0 wz--n- 296.00m 296.00m
  vg1          1   0   0 wz--n-   1.49g   1.49g
  vg_kvm11     1   2   0 wz--n-   3.41g      0
```

6. 查看 kvm11 新增名爲 vg1 的 VG。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "vgdisplay vg1"
kvm11 | CHANGED | rc=0 >>
  --- Volume group ---
  VG Name               vg1
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
```

```
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               1.49 GiB
  PE Size               8.00 MiB
  Total PE              191
  Alloc PE / Size       0 / 0
  Free  PE / Size       191 / 1.49 GiB
  VG UUID               tZTxdg-0zKV-VT5R-1OdZ-E9jc-R71I-29HsgS
```

17.3 刪除 VG

1. 查看 lvg 模組文件，刪除 VG 的範例。

```
ceuser@deyu:~/ansible_pb$ ansible-doc lvg | grep Remove -A3
- name: Remove a volume group with name vg.services
  community.general.lvg:
    vg: vg.services
    state: absent
```

2. 撰寫 rmvgs.yml，使用 lvg 模組刪除 VG vg1。

```
ceuser@deyu:~/ansible_pb$ cat rmvgs.yml
---
- hosts: dev
  tasks:
  - name: Remove a volume group with name vg1
    community.general.lvg:
      vg: vg1
      state: absent
```

3. ansible-playbook 成功執行 rmvgs.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook rmvgs.yml

PLAY [dev]
    ********************************************************************


TASK [Gathering Facts]
    *********************************************************
ok: [kvm11]
```

```
TASK [Remove a volume group with name vg1]
    ************************************
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm11 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

4. 查看 kvm11 名爲 vg1 的 VG 不存在了。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "vgs"
kvm11 | CHANGED | rc=0 >>
  VG       #PV #LV #SN Attr   VSize VFree
  vg_kvm11   1   2   0 wz--n- 3.41g    0
```

17.4 刪除分割區

1. 查看 parted 模組文件，刪除分割區的範例。

```
ceuser@deyu:~/ansible_pb$ ansible-doc parted | grep Remove -A5
- name: Remove partition number 1
  community.general.parted:
    device: /dev/sdb
    number: 1
    state: absent

--
- name: Remove all partitions from disk
  community.general.parted:
    device: /dev/sdb
    number: '{{ item.num }}'
    state: absent
  loop: '{{ sdb_info.partitions }}'
```

2. 撰寫 rmparts.yml，使用 parted 模組刪除 vdb1 分割區。

```
ceuser@deyu:~/ansible_pb$ cat rmparts.yml
---
- hosts: dev
  tasks:
  - name: Remove partitions from disk
    community.general.parted:
```

```
      device: /dev/vdb
      number: 1
      state: absent
```

3. ansible-playbook 執行 rmparts.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook rmparts.yml

PLAY [dev]
    ********************************************************************


TASK [Gathering Facts]
    ***********************************************************
ok: [kvm11]

TASK [Remove partitions from disk]
    *****************************************
changed: [kvm11]

PLAY RECAP
    ********************************************************************

kvm11 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

4. 再查看 kvm11 vdb 硬碟的 vdb1 ，已不存在。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "lsblk"
kvm11 | CHANGED | rc=0 >>
NAME              MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
vda               252:0    0    6G  0 disk
  vda1            252:1    0    1M  0 part
  vda2            252:2    0  400M  0 part /boot
 vda3            252:3    0  3.4G  0 part
   vg_kvm11-root 253:0    0  3.3G  0 lvm  /
   vg_kvm11-swap 253:1    0   96M  0 lvm  [SWAP]
vdb               252:16   0  1.5G  0 disk
```

17.5 lvol 模組

1. 查看 lvol 模組文件，產生 lv 的範例。

```
ceuser@deyu:~/ansible_pb$ ansible-doc lvol | grep ^EX -A6
```

```
EXAMPLES:

- name: Create a logical volume of 512m
  community.general.lvol:
    vg: firefly
    lv: test
    size: 512
```

2. 查詢主機 dev 中有 VG fieldwork。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "vgs"
kvm11 | CHANGED | rc=0 >>
  VG         #PV #LV #SN Attr   VSize   VFree
  fieldwork    1   0   0 wz--n- 296.00m 296.00m
  vg_kvm11     1   2   0 wz--n-   3.41g      0
```

3. 編輯 lvfs.yml 腳本，在主機 dev 的 fieldwork vg 新增 test lv，大小 100M。

```
ceuser@deyu:~/ansible_pb$ cat lvfs.yml
---
- hosts: dev
  tasks:
  - name: Create a logical volume of 100m
    community.general.lvol:
      vg: fieldwork
      lv: test
      size: 100
```

4. ansible-playbook 執行 lvfs.yml 腳本，成功產生 test 邏輯卷冊。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook lvfs.yml

PLAY [dev]
    ********************************************************************



TASK [Gathering Facts]
    *********************************************************
ok: [kvm11]

TASK [Create a logical volume of 100m]
    ****************************************
changed: [kvm11]

PLAY RECAP
    ********************************************************************
```

```
kvm11 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

5. 查看 kvm11 的 VG fieldwork 新增了 LV test。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "lvs"
kvm11 | CHANGED | rc=0 >>
  LV   VG       Attr     LSize Pool Origin Data%  Meta%  Move Log Cpy%
      Sync Convert
  test fieldwork -wi-a----- 100.00m
  root vg_kvm11  -wi-ao----   3.32g
  swap vg_kvm11  -wi-ao----  96.00m
```

17.6 filesystem 模組

1. 查看 filesystem 模組文件，格式化檔案系統的範例。

```
ceuser@deyu:~/ansible_pb$ ansible-doc filesystem | grep ^EX -A5
EXAMPLES:

- name: Create a ext2 filesystem on /dev/sdb1
  community.general.filesystem:
    fstype: ext2
    dev: /dev/sdb1
```

2. 編輯 lvfs.yml 腳本，新增格式化任務，將 /dev/fieldwork/test 邏輯卷冊格式化爲 ext4 檔案系統。

```
ceuser@deyu:~/ansible_pb$ vi lvfs1.yml
ceuser@deyu:~/ansible_pb$ cat lvfs1.yml
---
- hosts: dev
  tasks:
  - name: Create a logical volume of 100m
    community.general.lvol:
      vg: fieldwork
      lv: test
      size: 100

  - name: Create a ext4 filesystem on /dev/fieldwork/test
    community.general.filesystem:
      fstype: ext4
      dev: /dev/fieldwork/test
```

3. ansible-playbook 執行 lvfs.yml 腳本，成功格式化 test 邏輯卷冊。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook lvfs1.yml

PLAY [dev]
    ***********************************************************************


TASK [Gathering Facts]
    *********************************************************
ok: [kvm11]

TASK [Create a logical volume]
    ************************************************
ok: [kvm11]

TASK [Create a ext4]
    ************************************************************
changed: [kvm11]

PLAY RECAP
    ***********************************************************************

kvm11 : ok=3 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

4. 查看 kvm11 的 /dev/fieldwork/test，檔案格式是 ext4。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "fsck -N /dev/
    fieldwork/test"
kvm11 | CHANGED | rc=0 >>
fsck from util-linux 2.40.2
[/usr/sbin/fsck.ext4 (1) -- /dev/mapper/fieldwork-test] fsck.ext4 /dev/
    mapper/fieldwork-test
```

17.7 移除 lv

1. 寫 lvremove.yml 腳本移除先前新增的 lv /dev/fieldwork/test。

```
ceuser@deyu:~/ansible_pb$ vi lvremove.yml
ceuser@deyu:~/ansible_pb$ cat lvremove.yml
---
- hosts: dev
  tasks:
  - name: Remove the logical volume.
    community.general.lvol:
      vg: fieldwork
```

```
      lv: test
      state: absent
      force: true
```

2. ansible-playbook 執行 lvremove.yml 腳本，成功刪除 lv test。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook lvremove.yml

PLAY [dev]
    ********************************************************************


TASK [Gathering Facts]
    ***********************************************************
ok: [kvm11]

TASK [Remove a logical volume]
    *************************************************
changed: [kvm11]

PLAY RECAP
    ********************************************************************

kvm11 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. 查看 kvm11 沒有名爲 test 的 LV。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "lvs"
kvm11 | CHANGED | rc=0 >>
  LV   VG      Attr    LSize Pool Origin Data%  Meta%  Move Log Cpy%Sync
       Convert
  root vg_kvm11 -wi-ao----  3.32g
  swap vg_kvm11 -wi-ao---- 96.00m
```

17.8 實機練習

1. 撰寫 lvmfs1.yml，在 balancers 主機進行以下任務：

   (a) 使用 parted 模組，使用整顆硬碟 /dev/vdc 新增一個分割區。

   (b) 使用 lvg 模組在新增的分割區產生名爲 lvm1vg 的 vg，pesize 爲 4M。

   (c) 使用 lvol 模組在 lvm1vg vg，新增一名爲 lvm1lv，大小爲 50M 的 lv，如果已存在則移除再產生。

(d) 使用 filesystem 模組，將 lv /dev/lvm1vg/lvm1lv 格式化爲 ext3。

(e) 做錯或參數變動時，使用模組移除分割區、vg, lv。

Chapter 18

Block 錯誤處理

18.1 Block-rescue-always

1. Playbooks 中的區塊 (block) 經常配合 when 條件在某些狀況下執行，本章介紹當 block 執行任務出現錯誤時，執行 rescue 任務，而 always 則不管有無錯誤都執行。

```
- name: Attempt and graceful roll back demo
  block:
    - ansible.builtin.debug:
        msg: 'I execute normally'
    - name: i force a failure
      command: /bin/false
    - ansible.builtin.debug:
        msg: 'I never execute, due to the above task failing, :-('
  rescue:
    - ansible.builtin.debug:
        msg: 'I caught an error'
    - name: i force a failure in middle of recovery! >:-)
      command: /bin/false
    - ansible.builtin.debug:
        msg: 'I also never execute :-('
  always:
    - ansible.builtin.debug:
        msg: "This always executes"
```

2. debug 模組的 msg 列印客製化訊息，如果省略則預設列印”Hello World!”，查看 debug 模組範例。

```
ceuser@deyu:~/ansible_pb$ ansible-doc debug | grep ^EX -A4
EXAMPLES:

- name: Print the gateway for each host when defined
  ansible.builtin.debug:
```

```
6       msg: System {{ inventory_hostname }} has gateway {{
            ansible_default_ipv4.gateway }}
```

18.2 變數查詢

1. rescue 是在 block 執行出錯時才執行，判斷是否執行出錯，要先得知執行內容有哪些參數，使用 setup 模組過濾變數字串來查詢。例如：查詢主機 dev 的 ansible_lvm 相關參數如如下：

```
ceuser@deyu:~/ansible_pb$ ansible dev -m setup -a "filter=ansible_lvm"
kvm11 | SUCCESS => {
    "ansible_facts": {
        "ansible_lvm": {
            "lvs": {
                "root": {
                    "size_g": "3.32",
                    "vg": "vg_kvm11"
                },
                "swap": {
                    "size_g": "0.09",
                    "vg": "vg_kvm11"
                }
            },
            "pvs": {
                "/dev/vda3": {
                    "free_g": "0",
                    "size_g": "3.41",
                    "vg": "vg_kvm11"
                },
                "/dev/vda4": {
                    "free_g": "0.29",
                    "size_g": "0.29",
                    "vg": "fieldwork"
                },
                "/dev/vdb1": {
                    "free_g": "1.49",
                    "size_g": "1.49",
                    "vg": "vg1"
                }
            },
            "vgs": {
                "fieldwork": {
                    "free_g": "0.29",
                    "num_lvs": "0",
                    "num_pvs": "1",
                    "size_g": "0.29"
                },
                "vg_kvm11": {
                    "free_g": "0",
```

```
                    "num_lvs": "2",
                    "num_pvs": "1",
                    "size_g": "3.41"
                }
            }
        },
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

18.3 實例-正常任務

1. 腳本 blklvfs1.yml 使用 block - rescue - always 語法
   (a) when: 限定只有 fieldwork vg 存在時才執行。
   (b) 在 fieldwork 新增一個名爲 test，大小 200M 的 lv
   (c) rescue: 如果無法新增就改新增 100M 的 lv，且印出"Could not create logical volume of that size"。
   (d) always: 新增後將其格式化爲 ext4。

```
ceuser@deyu:~/ansible_pb$ vi blklvfs1.yml
ceuser@deyu:~/ansible_pb$ cat blklvfs1.yml
---
- hosts: dev
  tasks:
  - block:
    - name: Create a logical volume of 200m
      community.general.lvol:
        vg: fieldwork
        lv: test
        size: 200

    rescue:
    - ansible.builtin.debug:
        msg: Could not create logical volume of that size
    - name: Create a logical volume of 100m
      community.general.lvol:
        vg: fieldwork
        lv: test
        size: 100

    always:
    - name: Create a ext4
      community.general.filesystem:
        fstype: ext4
```

```
26          dev: /dev/fieldwork/test

28      when: ansible_lvm.vgs.fieldwork is defined
```

2. ansible-playbook 執行 lvfs1.yml 腳本，正常新增 200M lv test，且成功格式化。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook blklvfs1.yml
2
PLAY [dev]
    ********************************************************************
4
TASK [Gathering Facts]
    ***********************************************************
6 ok: [kvm11]
8 TASK [Create a logical volume of 200m]
    ****************************************
changed: [kvm11]
10
TASK [Create a ext4]
    ************************************************************
12 changed: [kvm11]
14 PLAY RECAP
    ********************************************************************

kvm11 : ok=3 changed=2  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. 查看 kvm11 fieldwork VG 內有一名爲 test 的 LV，大小爲 200M，且檔案格式爲 ext4。

```
1 ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "lvs"
kvm11 | CHANGED | rc=0 >>
3   LV   VG    Attr    LSize   Pool Origin Data%  Meta%  Move Log Cpy%Sync
        Convert
    test fieldwork -wi-a----- 200.00m
5   root vg_kvm11  -wi-ao----   3.32g
    swap vg_kvm11  -wi-ao----  96.00m
```

4. /dev/fieldwork/test 檔案格式爲 ext4。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "lsblk -f /dev/
    fieldwork/test"
```

```
kvm11 | CHANGED | rc=0 >>
NAME           FSTYPE FSVER LABEL UUID                 FSAVAIL FSUSE%
    MOUNTPOINTS
fieldwork-test ext4   1.0         b1b267cc-0814-4d80-98b0-3f3e7b62dcc7
```

18.4 實例-rescue

1. 將 blklvfs1.yml 腳本的 hosts 改爲 test，並存成 blklvfs2.yml。

```
ceuser@deyu:~/ansible_pb$ vi blklvfs2.yml
ceuser@deyu:~/ansible_pb$ grep hosts blklvfs2.yml
- hosts: test
```

2. ansible-playbook 執行 lvfs2.yml 腳本，因 kvm12 的 vg 不足 200M，所以無法新增，所以執行 rescue 任務，列印客製化訊息"Could not create logical volume of that size"，並改新增 100M 的 lv。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook blklvfs2.yml

PLAY [test]
    ********************************************************************

TASK [Gathering Facts]
    *********************************************************
ok: [kvm12]

TASK [Create a logical volume of 200m]
    ****************************************
fatal: [kvm12]: FAILED! => {"changed": false, "err": "  Volume group \"
    fieldwork
\" has insufficient free space (37 extents): 50 required.\n", "msg": "
    Creating
logical volume 'test' failed", "rc": 5}

TASK [ansible.builtin.debug]
    ***************************************************
ok: [kvm12] => {
    "msg": "Could not create logical volume of that size"
}

TASK [Create a logical volume of 100m]
    ****************************************
changed: [kvm12]

TASK [Create a ext4]
    ***********************************************************
```

```
changed: [kvm12]

PLAY RECAP
    ********************************************************************

kvm12 : ok=4 changed=2  unreachable=0  failed=0  skipped=0  rescued=1
    ignored=0
```

3. 查看 kvm11 fieldwork VG 內有一名爲 test 的 LV，大小爲 200M，且檔案格式爲 ext4。

```
ceuser@deyu:~/ansible_pb$ ansible test -m shell -a "lvs"
kvm12 | CHANGED | rc=0 >>
  LV   VG    Attr    LSize   Pool Origin Data%  Meta%  Move Log Cpy%Sync
       Convert
  test fieldwork -wi-a----- 100.00m
  root vg_kvm12  -wi-ao----   3.32g
  swap vg_kvm12  -wi-ao----  96.00m
```

4. /dev/fieldwork/test 檔案格式爲 ext4。

```
ceuser@deyu:~/ansible_pb$ ansible test -m shell -a "lsblk -f /dev/
    fieldwork/test"
kvm12 | CHANGED | rc=0 >>
NAME           FSTYPE FSVER LABEL UUID                 FSAVAIL FSUSE%
    MOUNTPOINTS
fieldwork-test ext4   1.0         3a214e0d-5589-48dd-93e6-084cce3770f1
```

18.5 實例-debug

1. 將 blklvfs2.yml 腳本的 hosts 改爲 blancers，並存成 blklvfs3.yml。

```
ceuser@deyu:~/ansible_pb$ vi blklvfs3.yml
ceuser@deyu:~/ansible_pb$ grep hosts blklvfs3.yml
- hosts: balancers
```

2. blklvfs3.yml，增加 debug 除錯，當 vg fieldwork 不存在時印出客製化訊息”Volume group does not exist”。

```
ceuser@deyu:~/ansible_pb$ cat blklvfs3.yml
```

```
---
- hosts: balancers
  tasks:
  ......................
  - ansible.builtin.debug:
      msg: Volume group does not exist
    when: ansible_lvm.vgs.fieldwork is undefined
```

3. ansible-playbook 執行 blklvfs3.yml 腳本，因 kvm15 的 vg fieldwork 不存在，所以執行 debug 任務，列印客製化訊息"Volume group does not exist"。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook blklvfs3.yml

PLAY [balancers]
    ****************************************************************

TASK [Gathering Facts]
    ************************************************************
ok: [kvm15]

TASK [Create a logical volume of 200m]
    ***************************************
skipping: [kvm15]

TASK [Create a ext4]
    ************************************************************
skipping: [kvm15]

TASK [ansible.builtin.debug]
    ***************************************************
ok: [kvm15] => {
    "msg": "Volume group does not exist"
}

PLAY RECAP
    ********************************************************************************

kvm15 : ok=2 changed=0  unreachable=0  failed=0  skipped=2  rescued=0
    ignored=0
```

18.6 實機練習

1. 修改 partition.yml 爲 partition1.yml 腳本，增加 block...rescue...always 語法：

(a) 如果 /dev/vdX1 無法新增，輸出以下錯誤訊息，並改以 partsize1 大小分割。

```
Could not create partition of that size
```

(b) 如果 /dev/vdY 不存在，則輸出以下錯誤訊息：

```
1 Disk /dev/vdY does not exist
```

2. 使用 lvol 模組，先移除所有主機 VG fieldwork 的 LV。

3. 撰寫 lvfs.yml 腳本，對所有主機完成以下任務：

(a) when: 限定只有 fieldwork vg 存在時，才執行新增 lv。

(b) 在 fieldwork 新增一個名爲 data，大小 240M 的 lv。

(c) rescue: 如果無法新增就改新增 120M 的 lv，且 debug 印出"Could not create LV data of that size"。

(d) always: 新增後將其格式化爲 ext3。

(e) when: 限定只有 fieldwork vg 不存在時，才執行 debug 除錯，印出客製化訊息"VG fieldwork does not exist"。

Chapter 19

Group 模組

19.1 前言

1. Group 模組可管理群組帳號。
2. group 模組類似 Linux 指令 groupadd, groupdel, groupmod 等群組帳號管理功能。

19.2 模組文件

1. 查詢 group 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc group
```

2. 查詢 group 模組文件的範例，新增群組帳號 somegroup。

```
ceuser@deyu:~/ansible_pb$ ansible-doc group | grep ^EX -A5
EXAMPLES:

- name: Ensure group "somegroup" exists
  ansible.builtin.group:
    name: somegroup
    state: present
```

19.3 建立群組帳號

1. 撰寫 group.yml 腳本，增加一群組帳號 sharegrp。

```
ceuser@deyu:~/ansible_pb$ vi group.yml
ceuser@deyu:~/ansible_pb$ cat group.yml
---
- hosts: dev
  tasks:
  - name: Ensure group "sharegrp" exists
    ansible.builtin.group:
      name: sharegrp
      state: present
```

2. ansible-playbook 執行腳本 group.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook group.yml

PLAY [dev]
    ********************************************************************


TASK [Gathering Facts]
    *********************************************************
ok: [kvm11]

TASK [Ensure group "sharegrp" exists]
    ****************************************
changed: [kvm11]

PLAY RECAP
    ********************************************************************

kvm11 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. ansible 使用 shell 模組查看主機 dev 的群組帳號 sharegrp 已產生，gid 是 1001。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "getent group sharegrp
    "
kvm11 | CHANGED | rc=0 >>
sharegrp:x:1002:
```

19.4 實機練習

1. 撰寫 yaml 腳本 group.yml，要求如下：

(a) 對 dev 主機執行任務。

(b) 使用 group 模組，增加一群組帳號，帳號名稱爲 sharegrp。

2. 撰寫 yaml 腳本 group1.yml，要求如下：

(a) 對 dev 主機執行任務。

(b) 使用 group 模組，增加一群組帳號，帳號名稱爲 webtest。

3. 撰寫 yaml 腳本 group2.yml，要求如下：

(a) 對 test 主機執行任務。

(b) 使用 group 模組，增加一群組帳號，帳號名稱由命令列外加變數 gname 決定。

Chapter 20

File 模組

20.1 前言

1. File 模組可設定或移除檔案屬性、連結或目錄。
2. file 模組可在遠端建立和刪除檔案 (file)、目錄 (directory)、軟連結 (sym-links) 。
3. 類似 Linux 指令 chown, chmod, ln, mkdir 和 touch 等檔案管理功能。

20.2 模組文件

1. 查詢 file 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc file
```

2. 查詢 file 模組文件的範例，變更檔案屬性。

```
ceuser@deyu:~/ansible_pb$ ansible-doc file | grep ^EX -A7
EXAMPLES:

- name: Change file ownership, group and permissions
  ansible.builtin.file:
    path: /etc/foo.conf
    owner: foo
    group: foo
    mode: '0644'
```

3. 設定已在在檔案 /work 不安全屬性 1777。

```
- name: Give insecure permissions to an existing file
  ansible.builtin.file:
    path: /work
    owner: root
    group: root
    mode: '1777'
```

4. 建立軟連結

```
- name: Create a symbolic link
  ansible.builtin.file:
    src: /file/to/link/to
    dest: /path/to/symlink
    owner: foo
    group: foo
    state: link
```

5. 建立兩個硬連結

```
- name: Create two hard links
  ansible.builtin.file:
    src: '/tmp/{{ item.src }}'
    dest: '{{ item.dest }}'
    state: link
  loop:
    - { src: x, dest: y }
    - { src: z, dest: k }
```

6. touch 一個檔案，並使用符號方式設定屬性。

```
- name: Touch a file, using symbolic modes to set the permissions (
    equivalent to 0644)
  ansible.builtin.file:
    path: /etc/foo.conf
    state: touch
    mode: u=rw,g=r,o=r
```

7. touch 一個檔案，並增加或移除屬性。

```
- name: Touch the same file, but add/remove some permissions
  ansible.builtin.file:
    path: /etc/foo.conf
    state: touch
```

```
5     mode: u+rw,g-wx,o-rwx
```

8. touch 一個檔案，不變更修改及存取的時間。

```
- name: Touch again the same file, but dont change times this makes the
    task idempotent
  ansible.builtin.file:
    path: /etc/foo.conf
    state: touch
    mode: u+rw,g-wx,o-rwx
    modification_time: preserve
    access_time: preserve
```

9. 如果目錄不存在，則建立目錄。

```
- name: Create a directory if it does not exist
  ansible.builtin.file:
    path: /etc/some_directory
    state: directory
    mode: '0755'
```

10. 更新修改及存取的時間。

```
- name: Update modification and access time of given file
  ansible.builtin.file:
    path: /etc/some_file
    state: file
    modification_time: now
    access_time: now
```

11. 設定存取時間。

```
- name: Set access time based on seconds from epoch value
  ansible.builtin.file:
    path: /etc/another_file
    state: file
    access_time: '{{ "%Y%m%d%H%M.%S" | strftime(stat_var.stat.atime) }}'
```

12. 變更目錄 (含次目錄及檔案) 的擁有者及群組。

```
- name: Recursively change ownership of a directory
  ansible.builtin.file:
    path: /etc/foo
    state: directory
    recurse: yes
    owner: foo
    group: foo
```

13. 刪除檔案。

```
- name: Remove file (delete file)
  ansible.builtin.file:
    path: /etc/foo.txt
    state: absent
```

14. 刪除目錄 (含次目錄及檔案) 。

```
- name: Recursively remove directory
  ansible.builtin.file:
    path: /etc/foo
    state: absent
```

20.3 建立目錄

1. 撰寫 webtest.yml 腳本，建立目錄 /webtest，/webtest 屬於 webtest 群組，屬性 2775 (owner=rwx, group=rwx, other=rx, setGID)。

```
ceuser@deyu:~/ansible_pb$ vi webtest.yml
ceuser@deyu:~/ansible_pb$ cat webtest.yml
---
- name: Web test
  hosts: dev
  tasks:
  - ansible.builtin.file:
      path: /webtest
      state: directory
      group: webtest
      mode: '2775'
```

2. ansible-playbook 執行腳本 webtest.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook webtest.yml

PLAY [Web test]
    ***************************************************************

TASK [Gathering Facts]
    ***********************************************************
ok: [kvm11]

TASK [ansible.builtin.file]
    *************************************************
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm11: ok=2  changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. ansible 使用 shell 模組查看主機 test 的 /webtest 目錄屬性，完全符合 webtest.yml 腳本的要求。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "ls -ld /webtest"
kvm11 | CHANGED | rc=0 >>
drwxrwsr-x. 2 root webtest 6 Apr 24 18:03 /webtest
```

20.4 建立軟連結

1. 修改 webtest.yml 腳本，存成 webtest1.yml，使用 apache 角色架設網頁伺服器，新增一軟連結 /var/www/html/webtest，連結到目錄 /webtest。

```
ceuser@deyu:~/ansible_pb$ vi webtest1.yml
ceuser@deyu:~/ansible_pb$ cat webtest1.yml
---
- name: Web test
  hosts: dev
  roles:
    - apache
  tasks:
  - ansible.builtin.file:
      path: /webtest
      state: directory
      group: webtest
      mode: '2775'
```

```
15   - ansible.builtin.file:
         src: /webtest
17       dest: /var/www/html/webtest
         state: link
```

2. ansible-playbook 執行腳本 webtest1.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook webtest1.yml
2 PLAY [Web test]
    ********************************************************************

4 TASK [Gathering Facts]
    *********************************************************
ok: [kvm11]
6
TASK [apache : install the latest version of Apache]
    ***************************
8 ok: [kvm11]

10 TASK [apache : Start service httpd]
    ******************************************
changed: [kvm11]
12
TASK [apache : Enable service firewalld]
    ***************************************
14 ok: [kvm11]

16 TASK [apache : firewalld add http service]
    ***********************************
changed: [kvm11]
18
TASK [apache : Template a file]
    ************************************************
20 changed: [kvm11]

22 TASK [ansible.builtin.file]
    ****************************************************
ok: [kvm11]
24
TASK [ansible.builtin.file]
    ****************************************************
26 changed: [kvm11]

28 PLAY RECAP
    ********************************************************************

kvm11 : ok=8 changed=4  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. ansible 使用 shell 模組查看主機 test 的 /var/www/html/webtest 目錄屬性，

完全符合 webtest.yml 腳本的要求。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "ls -ld /var/www/html/
    webtest"
kvm11 | CHANGED | rc=0 >>
lrwxrwxrwx. 1 root root 8 Apr 24 18:07 /var/www/html/webtest -> /webtest
```

20.5 新增含內容的檔案

1. file 模組的 touch 只能產生空白檔，使用 copy 模組可以複製檔案，或新增有內容的檔案，copy 模組文件的範例如下：

```
ceuser@deyu:~/ansible_pb$ ansible-doc copy | grep inline -A3
- name: Copy using inline content
  ansible.builtin.copy:
    content: '# This file was moved to /etc/other.conf'
    dest: /etc/mine.conf
```

2. 修改 webtest1.yml 腳本，存成 webtest2.yml，在目錄 /webtest 下新增一個內容爲"WebTest on hostname"。

```
ceuser@deyu:~/ansible_pb$ vi webtest2.yml
ceuser@deyu:~/ansible_pb$ cat webtest2.yml
---
- name: Web test
  hosts: dev
  roles:
    - apache
  tasks:
  - ansible.builtin.file:
      path: /webtest
      state: directory
      group: webtest
      mode: '2775'

  - ansible.builtin.file:
      src: /webtest
      dest: /var/www/html/webtest
      state: link

  - ansible.builtin.copy:
      content: "WebTest on {{ ansible_hostname }}\n"
      dest: /webtest/index.html
```

3. ansible-playbook 執行腳本 webtest2.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook webtest2.yml

PLAY [Web test]
    ****************************************************************

TASK [Gathering Facts]
    ************************************************************
ok: [kvm11]

TASK [apache : install the latest version of Apache]
    ***************************
ok: [kvm11]

TASK [apache : Start service httpd]
    *****************************************
ok: [kvm11]

TASK [apache : Enable service firewalld]
    ************************************
ok: [kvm11]

TASK [apache : firewalld add http service]
    ***********************************
ok: [kvm11]

TASK [apache : Template a file]
    ************************************************
ok: [kvm11]

TASK [ansible.builtin.file]
    ****************************************************
ok: [kvm11]

TASK [ansible.builtin.file]
    ****************************************************
ok: [kvm11]

TASK [ansible.builtin.copy]
    ****************************************************
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm11 : ok=9 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

4. ansible 使用 shell 模組查看主機 test 的 /webtest/index.html 內容，完全符合 webtest.yml 腳本的要求。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "cat /webtest/index.
    html"
kvm11 | CHANGED | rc=0 >>
WebTest on kvm11
```

5. curl 連線 http://kvm11/webtest/index.html，回應 403 Forbidden，原因爲 /webtest 目錄的 selinux fcontext 限制存取。

```
ceuser@deyu:~/ansible_pb$ curl http://kvm11/webtest/index.html
ceuser@deyu:~/ansible_pb$ curl http://kvm11/webtest/
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
</body></html>
```

6. 查看 kvm11 中目錄 /webtest 的 selinux fcontext type 是 `default_t`，不是網頁的 `httpd_sys_content_t`。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "ls -Z /webtest"
kvm11 | CHANGED | rc=0 >>
system_u:object_r:default_t:s0 index.html
```

20.6 實機練習

1. 撰寫 YAML 腳本 file1.yml，對主機 dev 執行任務，要求如下：
   (a) 新增檔案，檔案名稱由命令列外加變數 file1 決定
   (b) file1 擁有者及群組都是 ceuser，屬性 `rw-rw----`。
2. 撰寫 YAML 腳本 dir1.yml，對主機 dev 執行任務，要求如下：
   (a) 新增目錄，目錄名稱由命令列外加變數 dir1 決定
   (b) dir1 所屬群組是 sharegrp，屬性 `rwxr-xr-x`。
3. 撰寫 YAML 腳本 apache-dev.yml，對主機 dev 執行任務：使用 apache 角色，安裝 apache 網頁服務。

4. 撰寫 YAML 腳本 link1.yml，對主機 dev 執行任務，要求如下：
   (a) 新增一軟連結 /var/www/html/web1，連結到目錄 link1。
   (b) link1 目錄由命令列外加變數 link1 決定，且與 dir1.yml 外加變數 dir1 產生的目錄相同。
5. 撰寫 YAML 腳本 copy1.yml，對主機 dev 執行任務，要求如下：
   (a) 目錄 dir1 新增一檔案 home.html，內容爲 Hometest on IP。
   (b) dir1 爲命令列外加變數，IP 使用 ansible 變數取得主機 IP。
6. 撰寫 YAML 腳本 webtest.yml，對主機 dev 執行任務，要求如下：
   (a) 新增群組 webtest。
   (b) 新增目錄 /webtest，所屬群群組 webtest，屬性 `drwxrwsr-x`。
   (c) 建立連結 /var/www/html/webtest 連結到 /webtest。
   (d) 使用 copy 模組，目標檔案爲 /webtest/index.html，其內容爲”WebTest on hostname”，hostname 使用 `ansible_hostname` 變數取得。
7. 撰寫 YAML 腳本 webtest1.yml，對主機 dev 執行任務，要求如下：
   (a) 新增群組 webtest1。
   (b) 新增目錄 /webtest1，所屬群群組 webtest1，屬性 `drwxr-xr-x`。
   (c) 建立連結 /var/www/html/webtest1 連結到 /webtest1。
   (d) 使用 copy 模組，目標檔案爲 /webtest1/index1.html，其內容爲”WebTest1 on IP，IP 使用 ansible 變數取得主機 IP。

Chapter 21

Sefcontext 模組

21.1 前言

1. sefcontext 模組可管理檔案 selinux context。
2. sefcontext 模組類似 semanage fcontext 命令。

21.2 模組文件

1. 查詢 sefcontext 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc sefcontext
```

2. 修改 selinux fcontext 要使用命令 semanage，必須確定提供 semanage 命令的套件是否安裝，沒有的話要先安裝，安裝範例如下：

```
ceuser@deyu:~/ansible_pb$ ansible-doc dnf | grep ^EX -A5
EXAMPLES:

- name: Install the latest version of Apache
  ansible.builtin.dnf:
    name: httpd
    state: latest
```

3. 查詢 sefcontext 模組文件的範例，設定目標目錄 (含次錄) 的 setype 爲 httpd_git_rw_content_t。

```
ceuser@deyu:~/ansible_pb$ ansible-doc sefcontext | grep ^EX -A6
EXAMPLES:
```

```
3
- name: Allow apache to modify files in /srv/git_repos
5   community.general.sefcontext:
      target: '/srv/git_repos(/.*)?'
7     setype: httpd_git_rw_content_t
      state: present
```

4. 目標目錄 (含次錄) 的 setype，回復成設定值。

```
ceuser@deyu:~/ansible_pb$ ansible-doc sefcontext | grep 'Apply new
    SELinux' -A1
2 - name: Apply new SELinux file context to filesystem
    ansible.builtin.command: restorecon -irv /srv/git_repos
```

21.3 修改 setype

1. 撰寫 sefcontext.yml 腳本，semanage 及 restorecon 命令都必須存在，如果沒有要自行安裝，設定目錄 /webtest 及其次目錄的 fcontext 爲 httpd_sys_content_t，執行命令 restorecon 將 /webtest 目錄及其次目錄的 fcontext 還原成預設值。

```
1 ceuser@deyu:~/ansible_pb$ vi sefcontext.yml
  ceuser@deyu:~/ansible_pb$ cat sefcontext.yml
3 ---
  - name: Web test
5   hosts: dev
    tasks:
7   - name: Install the latest version of policycore.....
      ansible.builtin.dnf:
9       name: policycore......
        state: latest
11
    - name: Allow apache to modify files in /webtest
13    community.general.sefcontext:
        target: '/webtest(/.*)?'
15      setype: httpd_sys_content_t
        state: present
17
    - name: Apply new SELinux file context to filesystem
19    ansible.builtin.command: restorecon -irv /webtest
```

2. ansible-playbook 執行腳本 sefcontext.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook sefcontext.yml

PLAY [Web test]
    ***************************************************************

TASK [Gathering Facts]
    ***********************************************************
ok: [kvm11]

TASK [Install the latest version of policycore.....]
    ***************************
changed: [kvm11]

TASK [Allow apache to modify files in /webtest]
    *******************************
changed: [kvm11]

TASK [Apply new SELinux file context to filesystem]
    ***************************
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm11 : ok=4 changed=3  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. ansible 使用 shell 模組查看主機 test 的 /webtest 目錄，檔案的 index.html 的 fcontext 已改成 httpd_sys_content_t。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "ls -Z /webtest"
kvm11 | CHANGED | rc=0 >>
system_u:object_r:httpd_sys_content_t:s0 index.html
```

4. 使用 curl 連線 kvm11 的 /webtest 目錄，成功回應訊息。

```
ceuser@deyu:~/ansible_pb$ curl http://kvm11/webtest/
WebTest on kvm11
```

21.4 實機練習

1. 撰寫 sefcontext.yml 腳本，對 dev 主機要求如下：

(a) 安裝提供 semanage 命令的套件 查詢套件名稱

(b) 設定目錄 /webtest 及其次目錄的 fcontext 爲 `httpd_sys_content_t`。

(c) 設定目錄 /webtest1 及其次目錄的 fcontext 爲 `httpd_sys_content_t`。

(d) 執行命令 restorecon 將 /webtest 目錄及其次目錄的 fcontext 還原成預設值。

(e) 執行命令 restorecon 將 /webtest1 目錄及其次目錄的 fcontext 還原成預設值。

2. 修改 sefcontext.yml 爲 sefcontext1.yml，要求如下：

(a) 將目錄 /nfsdir 及其次目錄的 selinux 檔案 context，設定爲 nfs 分享目錄。

(b) 查詢 NFS 的 sefcontext type。

Chapter 22

Lineinfile 模組

22.1 前言

1. lineinfile 模組可配合正規表示法，修改、取代檔案的內容。
2. lineinfile 模組類似 Linux 的 sed 工具，以行爲單位做處理。
3. 如果要變更檔案中相似的多行，可使用 replace 模組。
4. 如果多行文字 (設定起點及終點) 做插入、更新、刪除，可用 blockinfile。

22.2 模組文件

1. 查詢 lineinfile 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc lineinfile
```

2. 查詢 lineinfile 模組文件的範例，修改 selinux 設定爲 enforcing。

```
1  ceuser@deyu:~/ansible_pb$ ansible-doc lineinfile | grep ^EX -A7
   EXAMPLES:
3
   # NOTE: Before 2.3, option 'dest', 'destfile' or 'name' was used instead
       of 'path'
5  - name: Ensure SELinux is set to enforcing mode
     ansible.builtin.lineinfile:
7      path: /etc/selinux/config
       regexp: '^SELINUX='
9      line: SELINUX=enforcing
```

3. 確認 sudoers 沒有設定 wheel 群組。

```
- name: Make sure group wheel is not in the sudoers configuration
  ansible.builtin.lineinfile:
    path: /etc/sudoers
    state: absent
    regexp: '^%wheel'
```

4. 設定 /etc/hosts 的屬性。

```
- name: Replace a localhost entry with our own
  ansible.builtin.lineinfile:
    path: /etc/hosts
    regexp: '^127\.0\.0\.1'
    line: 127.0.0.1 localhost
    owner: root
    group: root
    mode: '0644'
```

5. 取代 /etc/hosts 的 127.0.0.1 的紀錄爲只剩 localhost 一項。

```
- name: Replace a localhost entry searching for a literal string to
    avoid escaping
  ansible.builtin.lineinfile:
    path: /etc/hosts
    search_string: '127.0.0.1'
    line: 127.0.0.1 localhost
    owner: root
    group: root
    mode: '0644'
```

6. 確認 httpd 的 port 爲 8080。

```
- name: Ensure the default Apache port is 8080
  ansible.builtin.lineinfile:
    path: /etc/httpd/conf/httpd.conf
    regexp: '^Listen '
    insertafter: '^#Listen '
    line: Listen 8080
```

7. 修改 php 副檔名匹配模式。

```
- name: Ensure php extension matches new pattern
```

```
  ansible.builtin.lineinfile:
    path: /etc/httpd/conf/httpd.conf
    search_string: '<FilesMatch ".php[45]?$">'
    insertafter: '^\t<Location \/>\n'
    line: '        <FilesMatch ".php[34]?$">'
```

8. 確認增加 httpd 的註解行。

```
- name: Ensure we have our own comment added to /etc/services
  ansible.builtin.lineinfile:
    path: /etc/services
    regexp: '^# port for http'
    insertbefore: '^www.*80/tcp'
    line: '# port for http by default'
```

9. 如果檔案不存在，則新增檔案，加入一行。

```
- name: Add a line to a file if the file does not exist, without passing
    regexp
  ansible.builtin.lineinfile:
    path: /tmp/testfile
    line: 192.168.1.99 foo.lab.net foo
    create: yes
```

10. 確認 JBoss 記憶體設定符合要求。

```
# NOTE: Yaml requires escaping backslashes in double quotes but not in
    single quotes
- name: Ensure the JBoss memory settings are exactly as needed
  ansible.builtin.lineinfile:
    path: /opt/jboss-as/bin/standalone.conf
    regexp: '^(.*)Xms(\\d+)m(.*)$'
    line: '\1Xms${xms}m\3'
    backrefs: yes
```

11. 確認 ADMIN 是 sudoers。

```
# NOTE: Fully quoted because of the ': ' on the line. See the Gotchas in
    the YAML docs.
- name: Validate the sudoers file before saving
  ansible.builtin.lineinfile:
    path: /etc/sudoers
    state: present
    regexp: '^%ADMIN ALL='
```

```
    line: '%ADMIN ALL=(ALL) NOPASSWD: ALL'
    validate: /usr/sbin/visudo -cf %s
```

22.3 Facts 變數查詢

1. ansible 使用 setup 模組可以列出 ansible_facts 變數，增加 filter=.. 參數可以過濾變數。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m setup -a "filter=
    ansible_hostname"
kvm11 | SUCCESS => {
    "ansible_facts": {
        "ansible_hostname": "kvm11",
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

2. 過濾 ansible_mem* 跟記憶體有關的 ansible_facts 變數，得知總記憶體大小的變數爲 ansible_memory_mb.real.total 或 ansible_memtotal_mb。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m setup -a "filter=ansible_mem*"
kvm11 | SUCCESS => {
    "ansible_facts": {
        "ansible_memfree_mb": 1114,
        "ansible_memory_mb": {
            "nocache": {
                "free": 1453,
                "used": 252
            },
            "real": {
                "free": 1114,
                "total": 1705,
                "used": 591
            },
            "swap": {
                "cached": 0,
                "free": 95,
                "total": 95,
                "used": 0
            }
        },
        "ansible_memtotal_mb": 1705,
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
```

```
}
```

3. 查詢 bios 相關變數，版本變數爲 ansible_bios_version。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m setup -a "filter=ansible_bios_*
    "
kvm11 | SUCCESS => {
    "ansible_facts": {
        "ansible_bios_date": "04/01/2014",
        "ansible_bios_vendor": "SeaBIOS",
        "ansible_bios_version": "1.16.3-7.el10.alma.1",
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

4. 查詢 ansible_devices 變數，vda 大小變數爲 ansible_devices.vda.size。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m setup -a "filter=
    ansible_devices"
kvm11 | SUCCESS => {
    "ansible_facts": {
        "ansible_devices": {
            "dm-0": {
                "holders": [],
                "host": "",
                "links": {
                    "ids": [
                        "dm-name-vg_kvm11-root",
                        "dm-uuid-LVM-Vt5X..........Cay9sC5iT"
                    ],
                    "labels": [],
                    "masters": [],
                    "uuids": [
                        "b56f6615-6b57-4274-b59f-58ac3ecb2414"
                    ]
                },
                "model": null,
                "partitions": {},
                "removable": "0",
                "rotational": "1",
                "sas_address": null,
                "sas_device_handle": null,
                "scheduler_mode": "",
                "sectors": "6963200",
                "sectorsize": "512",
                "size": "3.32 GB",
                "support_discard": "512",
                "vendor": null,
```

```
                "virtual": 1
            },
            "dm-1": {
                "holders": [],
                "host": "",
                "links": {
                    "ids": [
                        "dm-name-vg_kvm11-swap",
                        "dm-uuid-LVM-Vt5X..............."
                    ],
                    "labels": [],
                    "masters": [],
                    "uuids": [
                        "23ace372-4bf3-4bf8-ad8b-d7a76616e459"
                    ]
                },
                "model": null,
                "partitions": {},
                "removable": "0",
                "rotational": "1",
                "sas_address": null,
                "sas_device_handle": null,
                "scheduler_mode": "",
                "sectors": "196608",
                "sectorsize": "512",
                "size": "96.00 MB",
                "support_discard": "512",
                "vendor": null,
                "virtual": 1
            },
            "dm-2": {
                "holders": [],
                "host": "",
                "links": {
                    "ids": [
                        "dm-name-fieldwork-test",
                        "dm-uuid-LVM-n8Y................"
                    ],
                    "labels": [],
                    "masters": [],
                    "uuids": [
                        "b1b267cc-0814-4d80-98b0-3f3e7b62dcc7"
                    ]
                },
                "model": null,
                "partitions": {},
                "removable": "0",
                "rotational": "1",
                "sas_address": null,
                "sas_device_handle": null,
                "scheduler_mode": "",
                "sectors": "409600",
                "sectorsize": "512",
                "size": "200.00 MB",
```

```
            "support_discard": "512",
            "vendor": null,
            "virtual": 1
        },
        "vda": {
            "holders": [],
            "host": "",
            "links": {
                "ids": [],
                "labels": [],
                "masters": [],
                "uuids": []
            },
            "model": null,
            "partitions": {
                "vda1": {
                    "holders": [],
                    "links": {
                        "ids": [],
                        "labels": [],
                        "masters": [],
                        "uuids": []
                    },
                    "sectors": "2048",
                    "sectorsize": 512,
                    "size": "1.00 MB",
                    "start": "2048",
                    "uuid": null
                },
                "vda2": {
                    "holders": [],
                    "links": {
                        "ids": [],
                        "labels": [],
                        "masters": [],
                        "uuids": [
                            "0af452ba-dcea-4255-8d81-1d48b7f82d8c"
                        ]
                    },
                    "sectors": "819200",
                    "sectorsize": 512,
                    "size": "400.00 MB",
                    "start": "4096",
                    "uuid": "0af452ba-dcea-4255-8d81-1d48b7f82d8c"
                },
                "vda3": {
                    "holders": [
                        "vg_kvm11-swap",
                        "vg_kvm11-root"
                    ],
                    "links": {
                        "ids": [
                            "lvm-pv-uuid-El2CWJ-3Y............."
                        ],
```

```
                            "labels": [],
                            "masters": [
                                "dm-0",
                                "dm-1"
                            ],
                            "uuids": []
                        },
                        "sectors": "7168000",
                        "sectorsize": 512,
                        "size": "3.42 GB",
                        "start": "823296",
                        "uuid": null
                    },
                    "vda4": {
                        "holders": [
                            "fieldwork-test"
                        ],
                        "links": {
                            "ids": [
                                "lvm-pv-uuid-kDnlEs-N9..........."
                            ],
                            "labels": [],
                            "masters": [
                                "dm-2"
                            ],
                            "uuids": []
                        },
                        "sectors": "614400",
                        "sectorsize": 512,
                        "size": "300.00 MB",
                        "start": "7991296",
                        "uuid": null
                    }
                },
                "removable": "0",
                "rotational": "1",
                "sas_address": null,
                "sas_device_handle": null,
                "scheduler_mode": "mq-deadline",
                "sectors": "12582912",
                "sectorsize": "512",
                "size": "6.00 GB",
                "support_discard": "512",
                "vendor": "0x1af4",
                "virtual": 1
            },
            "vdb": {
                "holders": [],
                "host": "",
                "links": {
                    "ids": [],
                    "labels": [],
                    "masters": [],
                    "uuids": []
```

```
                },
                "model": null,
                "partitions": {},
                "removable": "0",
                "rotational": "1",
                "sas_address": null,
                "sas_device_handle": null,
                "scheduler_mode": "mq-deadline",
                "sectors": "3145728",
                "sectorsize": "512",
                "size": "1.50 GB",
                "support_discard": "512",
                "vendor": "0x1af4",
                "virtual": 1
            }
        },
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```


22.4 修改檔案內容

1. 下載變數格式檔 report.orig。

```
ceuser@deyu:~/ansible_pb$ wget http://dywang.csie.cyut.edu.tw/materials/
   report.orig
```

2. 查看需要列出報告 report.orig 的變數。

```
ceuser@deyu:~/ansible_pb$ cat report.orig
# Variables report
HOST=inventoryhostname
MEMORY=memory_in_MB
VDA_SIZE=disk_vda_size
KERNEL=kernel_version
```

3. 撰寫 report.yml 腳本，使用正規表示法搜尋行首爲指定開鍵字的行，等號"="後的字串取代成指定的變數內容。

```
ceuser@deyu:~/ansible_pb$ vi report.yml
ceuser@deyu:~/ansible_pb$ cat report.yml
- name: Report variables
  hosts: test
```

```
  tasks:
6 - ansible.builtin.copy:
      src: report.orig
8     dest: /root/report.txt

10 - ansible.builtin.lineinfile:
      path: /root/report.txt
12    regexp: "^{{item.key}}="
      line: "{{item.key}}={{item.value}}"
14  loop:
    - { key: "HOST", value: "{{inventory_hostname}}" }
16  - { key: "MEMORY", value: "{{ansible_memtotal_mb}}" }
    - { key: "VDA_SIZE", value: "{{ansible_devices.vda.size}}" }
18  - { key: "KERNEL", value: "{{ansible_kernel}}" }
```

4. ansible-playbook 執行腳本 report.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook report.yml
2
PLAY [Report variables]
   ************************************************************
4
TASK [Gathering Facts]
   ************************************************************
6 ok: [kvm12]

8 TASK [ansible.builtin.copy]
   ************************************************
  changed: [kvm12]
10
  TASK [ansible.builtin.lineinfile]
   *******************************************
12 changed: [kvm12] => (item={'key': 'HOST', 'value': 'kvm12'})
  changed: [kvm12] => (item={'key': 'MEMORY', 'value': 1705})
14 changed: [kvm12] => (item={'key': 'VDA_SIZE', 'value': '6.00 GB'})
  changed: [kvm12] => (item={'key': 'KERNEL', 'value': '6.12.0-124.8.1.
   el10_1.x86_64'})
16
  PLAY RECAP
   *********************************************************************

18 kvm12 : ok=3 changed=2  unreachable=0  failed=0  skipped=0  rescued=0
   ignored=0
```

5. ansible 使用 shell 模組查看主機 test 的 /root/report.txt，檔案內容確實已改成變數查詢結果。

```
ceuser@deyu:~/ansible_pb$ ansible test -m shell -a "cat /root/report.txt
   "
```

```
kvm12 | CHANGED | rc=0 >>
# Variables report
HOST=kvm12
MEMORY=1705
VDA_SIZE=6.00 GB
KERNEL=6.12.0-124.8.1.el10_1.x86_64
```

22.5 變數不存在

1. 下載變數格式檔 report1.orig。

```
ceuser@deyu:~/ansible_pb$ wget http://dywang.csie.cyut.edu.tw/materials/
    report1.orig
```

2. 查看需要列出報告 report1.orig 的變數。

```
ceuser@deyu:~/ansible_pb$ cat report1.orig
# Variables report
HOST=inventoryhostname
BIOS=BIOS_version
VDA_SIZE=disk_vda_size
VDC_SIZE=disk_vdc_size
OS=os_family
```

3. 撰寫 report1.yml 腳本，因爲 kvm12 沒有 vdc 硬碟，ansible_devices.vdc.size 不會有定義，執行腳本會報錯無法完成。所以 value 設定使用 var|default('NONE')，當變數 var 不在在時，value 設定成預設值'NONE'，其他變數也都設定預設值'NONE'。

```
ceuser@deyu:~/ansible_pb$ vi report1.yml
ceuser@deyu:~/ansible_pb$ cat report1.yml
---
- name: Report variables
  hosts: test
  tasks:
  - ansible.builtin.copy:
      src: report1.orig
      dest: /root/report1.txt

  - ansible.builtin.lineinfile:
      path: /root/report1.txt
      regexp: "^{{item.key}}="
      line: "{{item.key}}={{item.value}}"
```

```
15      loop:
          - { key: "HOST", value: "{{inventory_hostname | default('NONE')}}"
              }
17        - { key: "BIOS", value: "{{ansible_bios_version | default('NONE')
            }}" }
          - { key: "VDA_SIZE", value: "{{ansible_devices.vda.size | default
            ('NONE')}}" }
19        - { key: "VDC_SIZE", value: "{{ansible_devices.vdc.size | default
            ('NONE')}}" }
          - { key: "OS", value: "{{ansible_os_family | default('NONE')}}" }
```

4. ansible-playbook 執行腳本 report1.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook report1.yml
2
PLAY [Report variables]
    ***********************************************************
4
TASK [Gathering Facts]
    ***********************************************************
6 ok: [kvm12]

8 TASK [ansible.builtin.copy]
    ***************************************************
changed: [kvm12]
10
TASK [ansible.builtin.lineinfile]
    *********************************************
12 changed: [kvm12] => (item={'key': 'HOST', 'value': 'kvm12'})
changed: [kvm12] => (item={'key': 'BIOS', 'value': '1.16.3-7.el10.alma.1
    '})
14 changed: [kvm12] => (item={'key': 'VDA_SIZE', 'value': '6.00 GB'})
changed: [kvm12] => (item={'key': 'VDC_SIZE', 'value': 'NONE'})
16 changed: [kvm12] => (item={'key': 'OS', 'value': 'RedHat'})

18 PLAY RECAP
    *********************************************************************

kvm12 : ok=3 changed=2  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

5. ansible 使用 shell 模組查看主機 test 的 /root/report1.txt，內容確實已改成變數查詢結果，且 VDC_SIZE=NONE。

```
1 ceuser@deyu:~/ansible_pb$ ansible test -m shell -a "cat /root/report1.
    txt"
kvm12 | CHANGED | rc=0 >>
3 # Variables report
HOST=kvm12
```

```
5 BIOS=1.16.3-7.el10.alma.1
  VDA_SIZE=6.00 GB
7 VDC_SIZE=NONE
  OS=RedHat
```

22.6 實機練習

1. 撰寫 linf1.yml，對主機 test 完成以下任務。
   (a) 使用 shell 模組，cat /etc/os-release 覆蓋導向到 /etc/dyw-release。
   (b) 修改 /etc/dyw-release 的 HOME_URL 爲" https://dywang.csie.cyut.edu.tw/" (包含雙引號)。
   (c) 確認 sudoers 沒有設定 wheel 群組。
   (d) 如果檔案 /tmp/linf1 不存在，則產生檔案並新增一行 linf1_test。
2. 使用 setup 模組查詢主機 test 的 ansible_facts 變數，並導向到目錄 /home/ceuser/$mysid 下指定檔案。
   (a) facts1.txt：沒有過濾。
   (b) facts2.txt：filter 過濾 ansible_default_ipv4。
   (c) facts3.txt：管線命令，grep 過濾 ipv4 及其後 4 行的結果。
3. 撰寫 report.yml，對 test 主機執行任務，要求如下：
   (a) 下載變數格式檔 report.orig。
   (b) 使用 copy 模組，將下載的 report.orig 複製到遠端的 /root/report.txt。
   (c) 使用 lineinfile 模組，以正規表示法搜尋行首爲指定開鍵字的行，等號"="後的字串取代成指定的變數內容。
   (d) 使用 loop 迴圈，針對檔案 /root/report.txt 中"=" 左邊關鍵字爲 item.key，找到正確的字串爲 item.value。
4. 修改 report.yml 爲
   (a) report.a.yml：目標檔案爲 /root/report.a.txt，增加一 lineinfile 任務，在 /root/report.a.txt 的 HOST 前一行增加一項，key 爲 DATE，value 爲ansible_date_time.date。
   (b) report.b.yml：修改 /etc/dyw-release 以下三項，並使用 loop 迴圈修改以下內容，value 都必須用雙引號括起來。

i. key: HOME_URL，vaule: https://dywang.csie.cyut.edu.tw/

ii. key: ID，vaule: 變數 ansible_distribution

iii. key: VERSION_ID，vaule: 變數 ansible_distribution_version

5. 撰寫 report1.yml，對 test 主機執行任務，要求如下：

   (a) 下載變數格式檔 report1.orig。

   (b) 任務如 report.yml，修改 report1.orig 每項 key 的 value。

   (c) value 由搜尋變數取得，當搜尋的變數不存在時，item.value 設定成預設值'NONE'。

6. 修改 report1.yml 爲

   (a) report1.a.yml：目標檔案爲 /root/report1.a.txt，增加一 lineinfile 任務，在 /root/report1.a.txt 的 HOST 後一行增加一項，key 爲 DATE，value 爲 ansible_date_time.date。

   (b) report1.b.yml：執行主機改爲 dev 及 test，目標檔案爲 /root/report1.b.txt，增加一 lineinfile 任務，在 /root/report1.b.txt 的 OS 後一行增加一項，key 爲 DATE，value 爲 ansible_date_time.date。

Chapter 23

Vault 加密

23.1 前言

1. 編寫 playbook 時可能涉及一些敏感資料，例如密碼，使用明文方式儲存敏感資料在 playbook 中是不能被接受的。
2. Vault 字面上意義是金庫、保險庫，在 Ansible 是指加密、解密檔案的技術。
3. Ansible 從 v1.5 開始支援此功能，預設使用 AES 加密技術。

23.2 ansible-vault 命令

1. ansible-vault 命令輔助說明，第一行列出的是所有可用的子命令。

```
ceuser@deyu:~/ansible_pb$ ansible-vault --help
usage: ansible-vault [-h] [--version] [-v]
                     {create,decrypt,edit,view,encrypt,encrypt_string,
                        rekey}
                     ...

encryption/decryption utility for Ansible data files

positional arguments:
  {create,decrypt,edit,view,encrypt,encrypt_string,rekey}
    create              Create new vault encrypted file
    decrypt             Decrypt vault encrypted file
    edit                Edit vault encrypted file
    view                View vault encrypted file
    encrypt             Encrypt YAML file
    encrypt_string      Encrypt a string
    rekey               Re-key a vault encrypted file

options:
```

```
  --version             show program's version number, config file
      location,
                        configured module search path, module location,
                        executable location and exit
  -h, --help            show this help message and exit
  -v, --verbose         Causes Ansible to print more debug messages.
      Adding
                        multiple -v will increase the verbosity, the
                            builtin
                        plugins currently evaluate up to -vvvvvv. A
                            reasonable
                        level to start is -vvv, connection debugging
                            might
                        require -vvvv. This argument may be specified
                            multiple
                        times.

See 'ansible-vault <command> --help' for more information on a specific
command.
```

23.3 建立加密檔案

1. create 建立加密檔 vfile.yml，輸入密碼後進入 vi 編輯檔案。

```
ceuser@deyu:~/ansible_pb$ ansible-vault create vfile.yml
New Vault password:
Confirm New Vault password:
```

2. 查看加密檔 vfile.yml，是一串亂數。

```
ceuser@deyu:~/ansible_pb$ cat vfile.yml
$ANSIBLE_VAULT;1.1;AES256
643863306564616137643566393136653661656230663064666633313239393631323761663534613
65353036646535366662306135366334383735636439363000
    a393030303131393837623532646330
393336323036373234373837316237323164666132613432343839633631333866626333613138393
37323433623233237370
    a393836383837653136373763663432393936613838353538346533306536
373966323238646336356632663639633232396539626236303965326465346431333
```

3. ansible-vault view 輸入密碼才能看到檔案 vfile.yml 的內容。

```
ceuser@deyu:~/ansible_pb$ ansible-vault view vfile.yml
Vault password:
vfile.yml vault test
```

4. edit 輸入密碼後，才可以編輯加密檔 vfile.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-vault edit vfile.yml
Vault password:
```

5. 再查看加密檔 vfile.yml，是一串亂數，但與之前的內容不一樣。

```
ceuser@deyu:~/ansible_pb$ cat vfile.yml
$ANSIBLE_VAULT;1.1;AES256
3065343366346634353762646261343335346634336336303839613637303637343930316634613
3

3065303663326561363833343562323633646630393437340
    a333833636262366332663162386330
6664386662646638663530336231656238386634613162346633653961313261383765633135303
3

3063633330613531630
    a65333165386566373532373035626339623637343265303036353634623
3
65333232343933666238383537336439303332623434353439636264323335373931
```

6. ansible-vault view 輸入密碼看到檔案 vfile.yml 的內容多了一行。

```
ceuser@deyu:~/ansible_pb$ ansible-vault view vfile.yml
Vault password:
vfile.yml vault test
1111
```

23.4 加解密現有檔案

1. 建立明文檔 vfile1.yml。

```
ceuser@deyu:~/ansible_pb$ echo 'vault file 1' > vfile1.yml

ceuser@deyu:~/ansible_pb$ cat vfile1.yml
vault file 1
```

2. encrypt 加密 vfile1.yml，輸入兩次密碼，成功加密。

```
ceuser@deyu:~/ansible_pb$ ansible-vault encrypt vfile1.yml
New Vault password:
Confirm New Vault password:
Encryption successful
```

3. 查看加密檔 vfile.yml，是一串亂數。

```
ceuser@deyu:~/ansible_pb$ cat vfile1.yml
$ANSIBLE_VAULT;1.1;AES256
30346561653064303436343036666238646432303264353035663362666534333839666265613639

3664303637646131323531356335343034623366653035650
    a32623462323531653739396363303033656238656231623239336335663335313632643532643037646162326637623163366161386464

3039336535626333340
    a656331646638303265333435363861656235303835306563313135356237
3737
```

4. view 輸入密碼後，才可以檢視加密檔 vfile1.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-vault view vfile1.yml
Vault password:
vault file 1
```

5. decrypt 解密 vfile1.yml，輸入密碼，成功解密。

```
ceuser@deyu:~/ansible_pb$ ansible-vault decrypt vfile1.yml
Vault password:
Decryption successful

ceuser@deyu:~/ansible_pb$ cat vfile1.yml
vault file 1
```

23.5 密碼檔解密 Vault

1. 密碼是敏感資料，不能直接寫在腳本，實作一加密密碼表 pwlist.yml。

```
ceuser@deyu:~/ansible_pb$ vi pwlist.yml
ceuser@deyu:~/ansible_pb$ cat pwlist.yml
devpw: '123123'
prodpw: 'qweqwe'
```

2. encrypt 加密 pwlist.yml，輸入密碼 pw4pwlist，成功加密。

```
ceuser@deyu:~/ansible_pb$ ansible-vault encrypt pwlist.yml
New Vault password:
Confirm New Vault password:
Encryption successful
```

3. 把 encrypt 加密密碼 pw4pwlist，儲存在 passowrd.txt，供後續解密用。

```
ceuser@deyu:~/ansible_pb$ echo 'pw4pwlist' > password.txt
ceuser@deyu:~/ansible_pb$ cat password.txt
pw4pwlist
```

4. 使用密碼檔解密 pwlist.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-vault view pwlist.yml --vault-password
    -file=password.txt
devpw: '123123'
prodpw: 'qweqwe'
```

23.6 rekey 重設密碼

1. 下載 vault 加密檔 rkvfile.yml。

```
ceuser@deyu:~/ansible_pb$ wget http://dywang.csie.cyut.edu.tw/materials/
    rkvfile.yml
```

2. 已知 rkvfile.yml 的加密密碼是'qweqwe'，使用 ansible-vault view 輸入密碼查看內容。

```
ceuser@deyu:~/ansible_pb$ ansible-vault view rkvfile.yml
Vault password:
```

```
3 ansible vault test
  edit
```

3. 已知 rkvfile.yml 的加密密碼是'qweqwe'，rekey 重設密碼爲'123qwe'。

```
ceuser@deyu:~/ansible_pb$ ansible-vault rekey rkvfile.yml
2 Vault password:
  New Vault password:
4 Confirm New Vault password:
  Rekey successful
```

4. 使用新密碼解密 rkvfile.yml。

```
1 ceuser@deyu:~/ansible_pb$ ansible-vault view rkvfile.yml
  Vault password:
3 ansible vault test
  edit
```

23.7 實機練習

1. 共同要求
   (a) ansible 工作目錄 /home/ceuser/$mysid
   (b) ansible-vault 有關的.yml 檔都不用增加註解。
   (c) 不要編輯重設密碼的加密檔。
2. create 一個 vault 加密檔 vfile.1.yml
   (a) 內容爲"vfile.1 test 202x-xx-xx"，其中日期爲今天。
   (b) 密碼爲 vpw1。
   (c) 密碼存到檔案 vfile.1.pw。
   (d) 使用密碼檔看是否可以解密？
3. 加密現有檔案 passwd：
   (a) 複製 /etc/passwd 到 /home/ceuser/$mysid 工作目錄。
   (b) 使用 ansible-vault 將檔案 passwd 加密，密碼爲 vpw2。

(c) 密碼存到檔案 passwd.pw。

(d) 使用密碼檔看是否可以解密？

4. rekey 只重設密碼，不要編輯變更內容，不用增加註解。

(a) 下載 vault 加密檔 rkvfile.yml。

(b) 下載 vault 加密檔 rkvfile1.yml。

(c) 下載 vault 加密檔 rkpwlist.yml。

(d) 以上三個檔加密密碼都是'qweqwe'，使用 ansible rekey 都重設密碼爲 vpw3。

(e) 將密碼 vpw3 儲存爲 rkpassword.txt。

(f) 使用密碼檔 rkpassword.txt 看是否可以解密？

Chapter 24

User 模組

24.1 前言

1. User 模組可管理用户帳號。
2. user 類似 Linux 指令 useradd, userdel, usermod 等帳號管理功能。

24.2 模組文件

1. 查詢 user 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc user
```

2. 查詢 user 模組文件的範例，新增帳號 johnd，且指定群組及 uid。

```
ceuser@deyu:~/ansible_pb$ ansible-doc user | grep ^EX -A50
EXAMPLES:

- name: Add the user 'johnd' with a specific uid and a primary group of
    'admin'
  ansible.builtin.user:
    name: johnd
    comment: John Doe
    uid: 1040
    group: admin
```

3. 新增帳號 james，且指定群組及 shell。

```
- name: Add the user 'james' with a bash shell, appending the group '
    admins'
```

```
and 'developers' to the user's groups
  ansible.builtin.user:
    name: james
    shell: /bin/bash
    groups: admins,developers
    append: yes
```

4. 刪除帳號 johnd。

```
- name: Remove the user 'johnd'
  ansible.builtin.user:
    name: johnd
    state: absent
    remove: yes
```

5. 新增帳號 jsmith，產生 ssh 金鑰。

```
- name: Create a 2048-bit SSH key for user jsmith in ~jsmith/.ssh/id_rsa
  ansible.builtin.user:
    name: jsmith
    generate_ssh_key: yes
    ssh_key_bits: 2048
    ssh_key_file: .ssh/id_rsa
```

6. 新增帳號 james18，設定帳户過期時間。

```
- name: Added a consultant whose account you want to expire
  ansible.builtin.user:
    name: james18
    shell: /bin/zsh
    groups: developers
    expires: 1422403387
```

7. 移除帳號 james18 的過期時間。

```
- name: Starting at Ansible 2.6, modify user, remove expiry time
  ansible.builtin.user:
    name: james18
    expires: -1
```

8. 設定 ram19 密碼最長（過期要變更）使用天數。

```
- name: Set maximum expiration date for password
  ansible.builtin.user:
    name: ram19
    password_expire_max: 10
```

9. 設定 pushkar15 密碼最短（不能變更）使用天數。

```
- name: Set minimum expiration date for password
  ansible.builtin.user:
    name: pushkar15
    password_expire_min: 5
```

24.3 建立用户

1. 撰寫 users.yml 腳本，新增用户 james 且附屬群組 sharegrp。

```
ceuser@deyu:~/ansible_pb$ vi users.yml
ceuser@deyu:~/ansible_pb$ cat users.yml
---
- hosts: dev
  tasks:
  - name: Ensure group "sharegrp" exists
    ansible.builtin.group:
      name: sharegrp
      state: present
  - name: Add user 'james'
    ansible.builtin.user:
      name: james
      groups: sharegrp
```

2. ansible-playbook 執行腳本 users.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook users.yml

PLAY [dev]
    ********************************************************************



TASK [Gathering Facts]
    *********************************************************
ok: [kvm11]

TASK [Ensure group "sharegrp" exists]
    ****************************************
```

```
ok: [kvm11]

TASK [Add user 'james']
    ***********************************************************
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm12: ok=3  changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. ansible 使用 shell 模組下 getnet passwd 命令查看主機 dev 的帳號 james。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "getent passwd james"
kvm11 | CHANGED | rc=0 >>
james:x:1001:1003::/home/james:/bin/bash
```

4. ansible 使用 shell 模組下 id 命令查看主機 dev 的帳號 james。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "id james"
kvm11 | CHANGED | rc=0 >>
uid=1001(james) gid=1003(james) groups=1003(james),1002(sharegrp)
```

24.4 設定帳號密碼

1. users1.yml 腳本，增加 password 密碼設定，明文密碼'123qwe' 要經過 password_hash 函式取得雜湊值，否則會將明文'123qwe' 存入 /etc/shadow 中，而無法使用。經過運算的結果要以雙大括號括起來取其結果，而整串文字要用雙引號括起來，如下顯示：

```
ceuser@deyu:~/ansible_pb$ vi users1.yml
ceuser@deyu:~/ansible_pb$ cat users1.yml
---
- hosts: dev
  tasks:
  - name: Ensure group "sharegrp" exists
    ansible.builtin.group:
      name: sharegrp
      state: present
  - name: Add user 'james'
    ansible.builtin.user:
      name: james
```

```
13        groups: sharegrp
          password: "{{ '123qwe' | password_hash('sha512') }}"
```

2. ansible-playbook 執行腳本 users.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook users1.yml
2
PLAY [dev]
    ********************************************************************


4
TASK [Gathering Facts]
    *********************************************************
6 ok: [kvm11]

8 TASK [Ensure group "sharegrp" exists]
    ***************************************
ok: [kvm11]
10
TASK [Add user 'james']
    ***********************************************************
12 [DEPRECATION WARNING]: Encryption using the Python crypt module is
    deprecated.
The Python crypt module is deprecated and will be removed from Python
    3.13.
14 Install the passlib library for continued encryption functionality. This
feature will be removed in version 2.17. Deprecation warnings can be
    disabled
16 by setting deprecation_warnings=False in ansible.cfg.
changed: [kvm11]
18
PLAY RECAP
    ********************************************************************


20 kvm11: ok=3  changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. 使用密碼'123qwe'，成功登入 james@kvm11。

```
ceuser@deyu:~/ansible_pb$ ssh james@kvm11
2 james@kvm11's password:
```

4. 登出 james@kvm11。

```
[james@kvm11 ~]$ exit
2 logout
```

```
Connection to kvm11 closed.
```

24.5 設定密碼過期

1. 帳號密碼過期設定，password_expire_max 設定密碼有效的最大天數。

```
ceuser@deyu:~/ansible_pb$ ansible-doc user | grep -A5 '^-
    password_expire_max'
- password_expire_max
        Maximum number of days between password change.
        Supported on Linux only.
        [Default: (null)]
        type: int
        added in: version 2.11 of ansible-core
```

2. users2.yml 腳本，增加設定密碼 10 天後過期。

```
ceuser@deyu:~/ansible_pb$ vi users2.yml
ceuser@deyu:~/ansible_pb$ cat users2.yml
---
- hosts: dev
  tasks:
  - name: Ensure group "sharegrp" exists
    ansible.builtin.group:
      name: sharegrp
      state: present
  - name: Remove user 'james'
    ansible.builtin.user:
      name: james
      state: absent
      remove: yes
  - name: Add user 'james'
    ansible.builtin.user:
      name: james
      groups: sharegrp
      password: "{{ '123qwe' | password_hash('sha512') }}"
      password_expire_max: 10
```

3. ansible-playbook 執行 users2.yml 腳本。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook user2.yml

PLAY [dev]
    ********************************************************************
```

```
4
TASK [Gathering Facts]
    ***********************************************************
6 ok: [kvm11]

8 TASK [Ensure group "sharegrp" exists]
    ****************************************
ok: [kvm11]
10
TASK [Remove user 'james']
    ***************************************************
12 changed: [kvm11]

14 TASK [Add user 'james']
    ***********************************************************
[DEPRECATION WARNING]: Encryption using the Python crypt module is
    deprecated.
16 The Python crypt module is deprecated and will be removed from Python
    3.13.
Install the passlib library for continued encryption functionality. This
18 feature will be removed in version 2.17. Deprecation warnings can be
    disabled
by setting deprecation_warnings=False in ansible.cfg.
20 changed: [kvm11]

22 PLAY RECAP
    *********************************************************************

kvm11 : ok=4 changed=2  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

4. 查看 kvm11 帳號 james，密碼於 10 天後 2023/06/06 過期。

```
1 ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "chage -l james"
kvm11 | CHANGED | rc=0 >>
3 Last password change                    : May 06, 2026
Password expires                        : May 16, 2026
5 Password inactive                       : never
Account expires                         : never
7 Minimum number of days between password change      : 0
Maximum number of days between password change      : 10
9 Number of days of warning before password expires   : 7
```

24.6 Vault 密碼檔

1. 修改 users.yml 腳本爲 `users_vault.yml`，密碼使用 vault 加密的 pwlist.yml 的密碼表，密碼變數爲 devpw：

```
ceuser@deyu:~/ansible_pb$ vi users_vault.yml
ceuser@deyu:~/ansible_pb$ cat users_vault.yml
---
- name: Create users with Valut
  hosts: dev
  vars_files: pwlist.yml
  tasks:
  - name: Ensure group "sharegrp" exists
    ansible.builtin.group:
      name: sharegrp
      state: present
  - name: Add user 'james'
    ansible.builtin.user:
      name: james
      groups: sharegrp
      password: "{{ devpw | password_hash('sha512') }}"
```

2. ansible-playbook 執行腳本 users_vault.yml，加選項 --ask-vault-pass 手動輸入密碼解密 pwlist.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook users_vault.yml --ask-vault-
    pass
Vault password:

PLAY [Create users with Valut]
    ***********************************************

TASK [Gathering Facts]
    ************************************************************
ok: [kvm11]

TASK [Ensure group "sharegrp" exists]
    ****************************************
ok: [kvm11]

TASK [Add user 'james']
    ************************************************************
changed: [kvm11]
Vault password:

PLAY [Create users with Valut]
    ***********************************************

TASK [Gathering Facts]
    ************************************************************
ok: [kvm11]

TASK [Ensure group "sharegrp" exists]
    ****************************************
ok: [kvm11]
```

```
TASK [Add user 'james']
    ***************************************************
[DEPRECATION WARNING]: Encryption using the Python crypt module is
    deprecated.
The Python crypt module is deprecated and will be removed from Python
    3.13.
Install the passlib library for continued encryption functionality. This
feature will be removed in version 2.17. Deprecation warnings can be
    disabled
by setting deprecation_warnings=False in ansible.cfg.
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm11 : ok=3 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0

PLAY RECAP
    *********************************************************************

kvm11: ok=3  changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. ansible-playbook 執行腳本 users_vault.yml，加選項 --vault-password-file 指定金鑰檔解密 pwlist.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook users_vault.yml --vault-
    password-file password.txt

PLAY [Create users with Valut]
    ************************************************

TASK [Gathering Facts]
    ***********************************************************
ok: [kvm11]

TASK [Ensure group "sharegrp" exists]
    ******************************************
ok: [kvm11]

TASK [Add user 'james']
    ***************************************************
[DEPRECATION WARNING]: Encryption using the Python crypt module is
    deprecated.
The Python crypt module is deprecated and will be removed from Python
    3.13.
Install the passlib library for continued encryption functionality. This
feature will be removed in version 2.17. Deprecation warnings can be
    disabled
by setting deprecation_warnings=False in ansible.cfg.
```

```
changed: [kvm11]

PLAY RECAP
    *********************************************************************

kvm11 : ok=3 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

4. 查看 devpw 變數，密碼爲'123123'。

```
ceuser@deyu:~/ansible_pb$ ansible-vault view pwlist.yml --vault-password
    -file=password.txt
devpw: '123123'
prodpw: 'qweqwe'
```

5. 使用密碼'123123'，成功登入 james@kvm11。

```
ceuser@deyu:~/ansible_pb$ ssh james@kvm11
james@kvm11's password:
```

6. 登出 james@kvm11。

```
[james@kvm11 ~]$ exit
logout
Connection to kvm11 closed.
```

24.7 建立不同帳號

1. 修改 users_vault.yml 腳本爲 user4host.yml，dev, test 產生的群組及帳號不同於 prod：

```
ceuser@deyu:~/ansible_pb$ vi user4hosts.yml
ceuser@deyu:~/ansible_pb$ cat user4hosts.yml
---
- name: Create users with Valut
  hosts: dev test prod
  vars_files: pwlist.yml
  tasks:
  - block:
    - name: Ensure group "devgrp" exists
      ansible.builtin.group:
```

```
11          name: devgrp
            state: present
13      - name: Add users
          ansible.builtin.user:
15          name: "{{ item }}"
            groups: devgrp
17          password: "{{ devpw | password_hash('sha512') }}"
          loop:
19          - peter
            - linda
21      when: '"dev" in group_names or "test" in group_names'

23    - block:
        - name: Ensure group "prodgrp" exists
25        ansible.builtin.group:
            name: prodgrp
27          state: present
        - name: Add user 'rita'
29        ansible.builtin.user:
            name: rita
31          groups: prodgrp
            password: "{{ prodpw | password_hash('sha512') }}"
33      when: '"prod" in group_names'
```

2. ansible-playbook 執行腳本 user4host.yml，加選項 --ask-vault-pass 輸入密碼解密 pwlist.yml。

```
1   ceuser@deyu:~/ansible_pb$ ansible-playbook user4hosts.yml --ask-vault-
        pass
    Vault password:
3
    PLAY [Create users with Valut]
        ***********************************************
5
    TASK [Gathering Facts]
        *********************************************************
7   ok: [kvm11]
    ok: [kvm14]
9   ok: [kvm12]
    ok: [kvm13]
11
    TASK [Ensure group "devgrp" exists]
        ******************************************
13  skipping: [kvm13]
    skipping: [kvm14]
15  changed: [kvm11]
    changed: [kvm12]
17
    TASK [Add users]
        ************************************************************
19  [DEPRECATION WARNING]: Encryption using the Python crypt module is
```

```
    deprecated.
The Python crypt module is deprecated and will be removed from Python
    3.13.
Install the passlib library for continued encryption functionality. This
feature will be removed in version 2.17. Deprecation warnings can be
    disabled
by setting deprecation_warnings=False in ansible.cfg.
skipping: [kvm13] => (item=peter)
skipping: [kvm13] => (item=linda)
skipping: [kvm13]
skipping: [kvm14] => (item=peter)
skipping: [kvm14] => (item=linda)
skipping: [kvm14]
changed: [kvm11] => (item=peter)
changed: [kvm12] => (item=peter)
changed: [kvm11] => (item=linda)
changed: [kvm12] => (item=linda)

TASK [Ensure group "prodgrp" exists]
    ******************************************
skipping: [kvm11]
skipping: [kvm12]
changed: [kvm13]
changed: [kvm14]

TASK [Add user 'rita']
    ************************************************************
skipping: [kvm11]
skipping: [kvm12]
changed: [kvm14]
changed: [kvm13]

PLAY RECAP
    *********************************************************************

kvm11 : ok=3 changed=2  unreachable=0  failed=0  skipped=2  rescued=0
    ignored=0
kvm12 : ok=3 changed=2  unreachable=0  failed=0  skipped=2  rescued=0
    ignored=0
kvm13 : ok=3 changed=2  unreachable=0  failed=0  skipped=2  rescued=0
    ignored=0
kvm14 : ok=3 changed=2  unreachable=0  failed=0  skipped=2  rescued=0
    ignored=0
```

3. ansible-playbook 執行腳本 user4host.yml，加選項 `--vault-password-file` 指定金鑰檔解密 pwlist.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook user4hosts.yml --vault-
    password-file=password.txt

PLAY [Create users with Valut]
```

```
    ************************************************

TASK [Gathering Facts]
    ************************************************************
ok: [kvm11]
ok: [kvm14]
ok: [kvm13]
ok: [kvm12]

TASK [Ensure group "devgrp" exists]
    ******************************************
skipping: [kvm13]
skipping: [kvm14]
ok: [kvm11]
ok: [kvm12]

TASK [Add users]
    ********************************************************************
[DEPRECATION WARNING]: Encryption using the Python crypt module is
    deprecated.
The Python crypt module is deprecated and will be removed from Python
    3.13.
Install the passlib library for continued encryption functionality. This
feature will be removed in version 2.17. Deprecation warnings can be
    disabled
by setting deprecation_warnings=False in ansible.cfg.
skipping: [kvm13] => (item=peter)
skipping: [kvm13] => (item=linda)
skipping: [kvm13]
skipping: [kvm14] => (item=peter)
skipping: [kvm14] => (item=linda)
skipping: [kvm14]
changed: [kvm12] => (item=peter)
changed: [kvm11] => (item=peter)
changed: [kvm12] => (item=linda)
changed: [kvm11] => (item=linda)

TASK [Ensure group "prodgrp" exists]
    ******************************************
skipping: [kvm11]
skipping: [kvm12]
ok: [kvm13]
ok: [kvm14]

TASK [Add user 'rita']
    ************************************************************
skipping: [kvm11]
skipping: [kvm12]
changed: [kvm13]
changed: [kvm14]

PLAY RECAP
    *********************************************************************
```

```
47 kvm11 : ok=3 changed=1  unreachable=0  failed=0  skipped=2  rescued=0
       ignored=0
   kvm12 : ok=3 changed=1  unreachable=0  failed=0  skipped=2  rescued=0
       ignored=0
49 kvm13 : ok=3 changed=1  unreachable=0  failed=0  skipped=2  rescued=0
       ignored=0
   kvm14 : ok=3 changed=1  unreachable=0  failed=0  skipped=2  rescued=0
       ignored=0
```

24.8 測試帳號

1. 使用新增的帳號 peter 登入 kvm11 及 kvm12，輸入密碼'123123'，成功登入，查詢 id 也有附屬群組 devgrp。

```
   ceuser@deyu:~/ansible_pb$ ssh peter@kvm11
2  peter@kvm11's password:
   [peter@kvm11 ~]$ id
4  uid=1002(peter) gid=1005(peter) groups=1005(peter),1004(devgrp)
   context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
6  [peter@kvm11 ~]$ exit
   logout
8  Connection to kvm11 closed.

10 ceuser@deyu:~/ansible_pb$ ssh peter@kvm12
   peter@kvm12's password:
12 [peter@kvm12 ~]$ id
   uid=1001(peter) gid=1002(peter) groups=1002(peter),1001(devgrp)
14 context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
   [peter@kvm12 ~]$ exit
16 logout
   Connection to kvm12 closed.
```

2. 使用新增的帳號 rita 登入 kvm13 及 kvm14，輸入密碼'qweqwe'，成功登入，查詢 id 也有附屬群組 devgrp。

```
1  ceuser@deyu:~/ansible_pb$ ssh rita@kvm13
   rita@kvm13's password:
3  [rita@kvm13 ~]$ id
   uid=1001(rita) gid=1002(rita) groups=1002(rita),1001(prodgrp)
5  context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
   [rita@kvm13 ~]$ exit
7  logout
   Connection to kvm13 closed.
9
   ceuser@deyu:~/ansible_pb$ ssh rita@kvm14
11 rita@kvm14's password:
```

```
[rita@kvm14 ~]$ id
uid=1001(rita) gid=1002(rita) groups=1002(rita),1001(prodgrp)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[rita@kvm14 ~]$ exit
logout
Connection to kvm14 closed.
```

3. 查詢 dev 及 test 主機不存在 rita 帳號。

```
ceuser@deyu:~/ansible_pb$ ansible dev -m shell -a "id rita"
kvm11 | FAILED | rc=1 >>
id: ‘rita’ : no such usernon-zero return code

ceuser@deyu:~/ansible_pb$ ansible test -m shell -a "id rita"
kvm12 | FAILED | rc=1 >>
id: ‘rita’ : no such usernon-zero return code
```

4. 查詢 prod 主機不存在 peter 帳號。

```
ceuser@deyu:~/ansible_pb$ ansible prod -m shell -a "id peter"
kvm13 | FAILED | rc=1 >>
id: ‘peter’ : no such usernon-zero return code

kvm14 | FAILED | rc=1 >>
id: ‘peter’ : no such usernon-zero return code
```

24.9 實機練習

1. 撰寫 user1.yml，對主機 test 完成以下任務。
   (a) 增加帳號 deyu2，註解 deyu2 Chen，不能使用 ssh 登入，uid 1099。
   (b) 增加帳號 deyu3，註解 deyu3 Wang，ssh 登入使用 /bin/sh，附屬群組 wang。
   (c) 增加帳號 deyu4，註解 deyu4 Wang，ssh 登入使用 /bin/sh，附屬群組 wang。
   (d) 增加帳號 deyu5，註解 deyu5 Wang，ssh 登入使用 /bin/sh，附屬群組 wang。
   (e) 附屬群組爲 wang 的帳户，使用 loop 迴圈依所列順序新增，帳户名稱及註解 (comment) 變數分別爲 name 及 comm。

(f) 用户所屬群組不存在，則自行新增。

2. 修改 user1.yml 爲 user2.yml

(a) 設定群組 wang 的每個 user 的密碼爲：

```
deyu3: deyu3.pw
deyu4: deyu4.pw
deyu5: deyu5.pw
```

(b) 設定密碼 exd1 天後過期。

(c) loop 迴圈增加一變數 pw，其值爲密碼的 hash 結果。

3. 依以下要求修改 user2.yml，存成 user3.yml。

(a) 將 user2.yml 要設定的用户密碼寫入密碼檔 wang.pw。

(b) user3.yml 改由密碼檔 wang.pw 取得要設定的密碼，迴圈也不使用 pw 變數，直接以用户名稱在密碼檔中找到對應密碼。

(c) wang.pw 必須使用 ansible-vault 加密，解密密碼爲 123qwe。

4. 修改 user3.yml 爲 user4.yml，處理的主機包含 dev, test 及 prod。

(a) 增加帳號 deyu6，註解 deyu6 Lin，ssh 登入使用 /bin/sh，附屬群組 lin。

(b) 增加帳號 deyu7，註解 deyu7 Lin，ssh 登入使用 /bin/sh，附屬群組 lin。

(c) 密碼檔 wang.pw 增加密碼列表，lin 群組的每個 user 的密碼：

```
deyu6: deyu6.pw
deyu7: deyu7.pw
```

(d) 設定密碼 exd2 天後過期。

(e) 執行主機包含 dev, test 及 prod。

(f) 只有主機 dev 及 test 才產生群組 wang 的 user。

(g) 只有主機 prod 才產生群組 lin 的 user。

(h) dev, test 及 prod 主機皆新增 user deyu2。

(i) 用户所屬群組不存在，則自行新增。

Chapter 25

Cron 模組

25.1 前言

1. Cron 模組可設定例行性工作。
2. 類似 Linux 指令 crontab。

25.2 模組文件

1. 查詢 cron 模組文件。

```
ceuser@deyu:~/ansible_pb$ ansible-doc cron
```

2. 查詢 cron 模組文件的範例，2 點及 5 點執行”ls -alh > /dev/null”。

```
ceuser@deyu:~/ansible_pb$ ansible-doc cron | grep ^EX -A55
EXAMPLES:

- name: Ensure a job that runs at 2 and 5 exists. Creates an entry like
       "0 5,2 * * ls -alh > /dev/null"
  ansible.builtin.cron:
    name: "check dirs"
    minute: "0"
    hour: "5,2"
    job: "ls -alh > /dev/null"
```

3. 刪除名爲”an old job” 的例行性工作。

```
- name: 'Ensure an old job is no longer present. Removes any job that is
       prefixed by "#Ansible: an old job" from the crontab'
```

```
  ansible.builtin.cron:
4   name: "an old job"
    state: absent
```

4. 新增一個 cron job，在 reboot 時執行"/some/job.sh"。

```
1 - name: Creates an entry like "@reboot /some/job.sh"
  ansible.builtin.cron:
3   name: "a job for reboot"
    special_time: reboot
5   job: "/some/job.sh"
```

5. 在 crontab 的最上方新增一行"PATH=/opt/bin"。

```
1 - name: Creates an entry like "PATH=/opt/bin" on top of crontab
  ansible.builtin.cron:
3   name: PATH
    env: yes
5   job: /opt/bin
```

6. 在 PATH 宣告行後面新增一行"APP_HOME=/src/app"。

```
1 - name: Creates an entry like "APP_HOME=/srv/app" and insert it after
      PATH declaration
  ansible.builtin.cron:
3   name: APP_HOME
    env: yes
5   job: /srv/app
    insertafter: PATH
```

7. 在 /etc/cron.d 目錄下新增一個例行性工作檔ansible_yum-autoupdate，每星期二 12 點整執行 job。

```
- name: Creates a cron file under /etc/cron.d
2 ansible.builtin.cron:
    name: yum autoupdate
4   weekday: 2
    minute: 0
6   hour: 12
    user: root
8   job: "YUMINTERACTIVE=0 /usr/sbin/yum-autoupdate"
    cron_file: ansible_yum-autoupdate
```

8. 刪除 /etc/cron.d 目錄下的例行性工作檔ansible_yum-autoupdate。

```
- name: Removes a cron file from under /etc/cron.d
  ansible.builtin.cron:
    name: "yum autoupdate"
    cron_file: ansible_yum-autoupdate
    state: absent
```

9. 刪除 crontab 的環境變數"APP_HOME=/src/app"。

```
- name: Removes "APP_HOME" environment variable from crontab
  ansible.builtin.cron:
    name: APP_HOME
    env: yes
    state: absent
```

25.3 建立例行性命令

1. 撰寫 cron.yml 腳本，以 ceuser 用户建立例行命令。

```
ceuser@deyu:~/ansible_pb$ vi cron.yml
ceuser@deyu:~/ansible_pb$ cat cron.yml
---
- name: cron test
  hosts: test
  tasks:
    - ansible.builtin.cron:
        name: "create a cron job for ceuser"
        weekday: '2'
        minute: '0'
        hour: '12'
        user: ceuser
        job: 'echo "My CRON_TEST"'
```

2. ansible-playbook 執行腳本 cron.yml。

```
ceuser@deyu:~/ansible_pb$ ansible-playbook cron.yml

PLAY [cron test]
    ************************************************************

TASK [Gathering Facts]
    ************************************************************
ok: [kvm12]
```

```
7
TASK [cron]
    ***********************************************************************
9 changed: [kvm12]

11 PLAY RECAP
    ***********************************************************************

kvm12 : ok=2 changed=1  unreachable=0  failed=0  skipped=0  rescued=0
    ignored=0
```

3. ansible 使用 shell 模組查看主機 test 用户 ceuser 的例行性命令。

```
ceuser@deyu:~/ansible_pb$ ansible test -m shell -a "crontab -u ceuser -l
    "
2 kvm12 | CHANGED | rc=0 >>
#Ansible: create a cron job for ceuser
4 0 12 * * 2 echo "My CRON_TEST"
```

25.4 實機練習

1. 撰寫以下 YAML 腳本，對主機 test 執行任務，建立用户 cronuser 的例行性工作 cronbin (內容查看參數檔)。
   (a) cron1.yml：每 13 小時
   (b) cron2.yml：3-13 點的 14 分
   (c) cron3.yml：每月 10 及 19 日的 13 點 45 分
   (d) cron4.yml：每週六的 4 點每 25 分鐘
   (e) cron5.yml：每天 8 點 51 分