



- Lab: 環境準備
- Lab: Inventory Variables 實作練習
- Lab: Ansible Module 實作練習
- Lab: Ansible Plugin 實作練習
- Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習

- ✓ Lab: 環境準備
- Lab: Inventory Variables 實作練習
 - Single File
 - Split Variable Files
- Lab: Ansible Module 實作練習
- Lab: Ansible Plugin 實作練習
- Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習

藉由此實作來透過單一 Inventory 檔案設定變數, 並且透過 Playbooks 顯示變數值。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini inventory/1_single_file 3. cd inventory/1_single_file 4. 確認 hosts.ini 設定正確。
Step2	修改 Inventory 檔, 加上 Group 跟 Global 變數
	1. vim hosts.ini 2. 加入以下內容(紅色部分請根據使用的 Lab 主機調整): [all:vars] vm_datacenter_name=PDISC vm_folder_name=studentN [linux:vars] app_name=WEB system_services='["httpd","mysql"]' 1. 存檔

```
[speaker1@ansgbtn 1_single_file]$ pwd
/home/speaker1/ansible-concept-lab/inventory/1_single_file
[speaker1@ansgbtn 1_single_file]$
[speaker1@ansgbtn 1_single_file]$
[speaker1@ansgbtn 1_single_file]$ cat hosts.ini
[linux]
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```

藉由此實作來透過單一 Inventory 檔案設定變數, 並且透過 Playbooks 顯示變數值。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini inventory/1_single_file 3. cd inventory/1_single_file 4. 確認 hosts.ini 設定正確。
Step2	修改 Inventory 檔, 加上 Group 跟 Global 變數
	1. vim hosts.ini 2. 加入以下內容(紅色部分請根據使用的 Lab 主機調整): [all:vars] vm_datacenter_name=PDISC vm_folder_name=studentN [linux:vars] app_name=WEB system_services='["httpd","mysql"]' 1. 存檔

```
[speaker1@ansgbtn 1_single_file]$ cat hosts.ini
[linux]
student01 ansible_user=root ansible_connection=ssh machine_type=virtual

[all:vars]
vm_datacenter_name=PDISC
vm_folder_name=student01

[linux:vars]
app_name=WEB
system_services='["httpd","mysql"]'
```

藉由此實作來透過單一 Inventory 檔案設定變數, 並且透過 Playbooks 顯示變數值。

步驟	說明
Step3	執行 Playbooks 顯示變數值
1. ansible-playbook -i hosts.ini display_info.yml	

```
[speaker1@ansgbtn 1_single_file]$ ansible-playbook -i hosts.ini display_info.yml

PLAY [Get the informaion of machines] *****

TASK [Display machine information] *****
ok: [speaker01] => {
  "msg": [
    "Machine Name: speaker01",
    "Application Name: WEB",
    "Service Name: [\"httpd\", \"mysql\"]",
    "vCenter Datacenter: PDISC",
    "VM Folder Name: Speaker01"
  ]
}

PLAY RECAP *****
speaker01 : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

- ✓ Lab: 環境準備
- Lab: Inventory Variables 實作練習
 - Single File
 - Split Variable Files
- Lab: Ansible Module 實作練習
- Lab: Ansible Plugin 實作練習
- Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習

藉由此實作來將 Inventory 變數拆分成多個檔案檔案, 並且透過 Playbooks 顯示變數值。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini inventory/2_split_var_files 3. cd inventory/2_split_var_files 4. 確認 hosts.ini 設定正確。
Step2	修改 Inventory 檔, 並建立 host_vars。
	1. vim hosts.ini, 把所有 host var 都移除掉 (ansible_user=root ansible_connection=ssh machine_type=virtual)。 2. mkdir host_vars 3. vim host_vars/studentN.yml, 填入以內容 --- ansible_user: root ansible_connection: ssh machine_type: virtua 1. 存檔

```
[speaker1@ansgbtn 2_split_var_files]$ cat hosts.ini  
[linux]  
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual  
[speaker1@ansgbtn 2_split_var_files]$
```


藉由此實作來將 Inventory 變數拆分成多個檔案檔案, 並且透過 Playbooks 顯示變數值。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini inventory/2_split_var_files 3. cd inventory/2_split_var_files 4. 確認 hosts.ini 設定正確。
Step2	修改 Inventory 檔, 並建立 host_vars。
	1. vim hosts.ini, 把所有 host var 都移除掉 (ansible_user=root ansible_connection=ssh machine_type=virtual)。 2. mkdir host_vars 3. vim host_vars/studentN.yml, 填入以內容 --- ansible_user: root ansible_connection: ssh machine_type: virtual 1. 存檔

```
[speaker1@ansgbtn 2_split_var_files]$ cat host_vars/student01.yml
---

ansible_user: root
ansible_connection: ssh
machine_type: virtual[speaker1@ansgbtn 2_split_var_files]$
```

藉由此實作來將 Inventory 變數拆分成多個檔案檔案, 並且透過 Playbooks 顯示變數值。

步驟	說明
Step3	建立 group_vars。
<div>1. mkdir group_vars</div> <div>2. vim group_vars/all.yml, 填入以內容</div> <div>---</div> <div>vm_datacenter_name: PDISC</div> <div>vm_folder_name: studentN</div> <div>1. 存檔</div> <div>2. vim group_vars/linux.yml, 填入以內容</div> <div>--</div> <div>app_name: WEB</div> <div>system_services:</div> <div>- httpd</div> <div>- mysql</div> <div>1. 存檔</div>	
Step4	執行 Playbooks 顯示變數值
<div>1. ansible-playbook -i hosts.ini display_info.yml</div>	

```
[speaker1@ansgbtn group_vars]$ cat linux.yml
---

vm_datacenter_name: PDISC
vm_folder_name: studentN

[speaker1@ansgbtn group_vars]$ cat all.yml
---

app_name: WEB
system_services:
- httpd
- mysql
[speaker1@ansgbtn group_vars]$ cat all.yml
---

vm_datacenter_name: PDISC
vm_folder_name: student01[speaker1@ansgbtn group_vars]$
```

藉由此實作來將 Inventory 變數拆分成多個檔案檔案, 並且透過 Playbooks 顯示變數值。

步驟	說明
Step3	建立 group_vars。
<div>1. mkdir group_vars</div> <div>2. vim group_vars/all.yml, 填入以內容</div> <div>---</div> <div>vm_datacenter_name: PDISC</div> <div>vm_folder_name: studentN</div> <div>1. 存檔</div> <div>2. vim group_vars/linux.yml, 填入以內容</div> <div>--</div> <div>app_name: WEB</div> <div>system_services:</div> <div>- httpd</div> <div>- mysql</div> <div>1. 存檔</div>	
Step4	執行 Playbooks 顯示變數值
<div>1. ansible-playbook -i hosts.ini display_info.yml</div>	

```
[speaker1@ansgbtn 2_split_var_files]$ ansible-playbook -i hosts.ini display_info.yml

PLAY [Get the informaion of machines] *****

TASK [Display machine information] *****
ok: [speaker01] => {
  "msg": [
    "Machine Name: speaker01",
    "Application Name: WEB",
    "Service Name: ['httpd', 'mysql']",
    "vCenter Datacenter: PDISC",
    "VM Folder Name: student01"
  ]
}

PLAY RECAP *****
speaker01 : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```



- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- Lab: Ansible Module 實作練習
 - Common Modules Reference
 - Deploy WebApp
 - Backup/Restore File
- Lab: Ansible Plugin 實作練習
- Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習

```
ansible [目的主機群組] -m [Module 名稱] -a “使用 Module 執行的動作”
```

[yum module](#), 安裝 httpd 套件

```
$ ansible linux -m yum -a "name=chrony state=present"
```

[template module](#), 從 control node 部署模板到 managed node

```
$ ansible linux -m template -a "src=./templates/index.html.j2 dest=/var/www/html/index.html"
```

[file module](#), 在 managed node 上建立目錄或檔案

```
$ ansible linux -m file -a "path=/tmp/backup state=directory"
```

[copy module](#), 在 managed node 上, 把檔案從來源複製到目的端

```
$ ansible linux -m copy -a "src=/var/www/html/index.html dest=/tmp/backup remote_src=yes"
```

[command module](#), 在 managed node 上執行指令

```
$ ansible linux -m command -a "uname -n"
```

```
ansible [目的主機群組] -m [Module 名稱] -a “使用 Module 執行的動作”
```

[systemd module](#), 管理 managed node 上 systemd service

```
$ ansible linux -m systemd -a "name=chronyd state=restarted"
```

[firewalld module](#), 管理 Linux 主機(RHEL or CentOS) 上防火牆設定

```
$ ansible linux -m firewalld -a "service=http permanent=yes state=enabled"
```

- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- Lab: Ansible Module 實作練習
 - Common Modules Reference
 - Deploy WebApp
 - Backup/Restore File
- Lab: Ansible Plugin 實作練習
- Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習

藉由此實作使用 Ansible Module 部署簡單的 WebApp:

部署 WebApp 步驟:

1. 使用 yum 安裝 httpd package
2. 將 index.html template 複製到 /var/www/html/index.html 裡
3. 透過 systemd 重啟 httpd 服務
4. 防火牆 firewalld 設定允許 http 服務
5. firewalld reload (`firewall-cmd --reload`)

移除 WebApp 步驟

1. 透過 systemd 關閉 httpd 服務
2. 使用 yum 移除 httpd package

藉由此實作使用 Ansible Module 部署簡單的 WebApp。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini module/1_webapp/inventory 3. cd module/1_webapp/ 4. 確認 hosts.ini 設定正確。
Step2	執行 ad-hoc 指令部署 WebApp
	1. ansible linux -m yum -a "name=httpd state=present" 2. ansible linux -m template -a "src=templates/index.html.j2 dest=/var/www/html/index.html" 3. ansible linux -m systemd -a "name=httpd enabled=yes state=started" 4. ansible linux -m firewalld -a "service=http permanent=yes state=enabled" 5. ansible linux -m command -a "firewall-cmd --reload"
Step3	測試是否能連到 WebApp
	1. curl studentN

```
[linux]
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```



藉由此實作使用 Ansible Module 部署簡單的 WebApp。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini module/1_webapp/inventory 3. cd module/1_webapp/ 4. 確認 hosts.ini 設定正確。
Step2	執行 ad-hoc 指令部署 WebApp
	1. ansible linux -m yum -a "name=httpd state=present" 2. ansible linux -m template -a "src=templates/index.html.j2 dest=/var/www/html/index.html" 3. ansible linux -m systemd -a "name=httpd enabled=yes state=started" 4. ansible linux -m firewalld -a "service=http permanent=yes state=enabled" 5. ansible linux -m command -a "firewall-cmd --reload"
Step3	測試是否能連到 WebApp
	1. curl studentN

```
[speaker1@ansgbtn 1_webapp]$ ansible linux -m yum -a "name=httpd state=present"
speaker01 | CHANGED => {
  "ansible_facts": {
    "pkg_mgr": "dnf"
  },
  "changed": true,
  "msg": "",
  "rc": 0,
  "results": [
    "Installed: apr-util-1.6.1-6.el8.x86_64",
    "Installed: apr-1.6.3-12.el8.x86_64",
    "Installed: apr-util-openssl-1.6.1-6.el8.x86_64",
    "Installed: httpd-2.4.37-51.module+el8.7.0+18499+2e106f0b.5.x86_64",
    "Installed: httpd-filesystem-2.4.37-51.module+el8.7.0+18499+2e106f0b.5.noarch",
    "Installed: redhat-logos-httpd-84.5-1.el8.noarch",
    "Installed: mod_http2-1.15.7-5.module+el8.7.0+18499+2e106f0b.4.x86_64",
    "Installed: httpd-tools-2.4.37-51.module+el8.7.0+18499+2e106f0b.5.x86_64",
    "Installed: apr-util-bdb-1.6.1-6.el8.x86_64"
  ]
}
```



藉由此實作使用 Ansible Module 部署簡單的 WebApp。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini module/1_webapp/inventory 3. cd module/1_webapp/ 4. 確認 hosts.ini 設定正確。
Step2	執行 ad-hoc 指令部署 WebApp
	1. ansible linux -m yum -a "name=httpd state=present" 2. ansible linux -m template -a "src=templates/index.html.j2 dest=/var/www/html/index.html" 3. ansible linux -m systemd -a "name=httpd enabled=yes state=started" 4. ansible linux -m firewalld -a "service=http permanent=yes state=enabled" 5. ansible linux -m command -a "firewall-cmd --reload"
Step3	測試是否能連到 WebApp
	1. curl studentN

```
[speaker1@ansgbtn 1_webapp]$ curl speaker01
THSRC Examples!!!

The following is custom message:

Hello THSRC!
```



藉由此實作使用 Ansible Module 部署簡單的 WebApp。

步驟	說明
Step4	執行 ad-hoc 指令移除 WebApp
<div>1. ansible linux -m systemd -a "name=httpd enabled=yes state=stopped"</div> <div>2. ansible linux -m yum -a 'name=httpd state=absent'</div>	

```
[speaker1@ansgbtn 1_webapp]$ ansible linux -m yum -a 'name=httpd state=absent'
speaker01 | CHANGED => {
  "changed": true,
  "msg": "",
  "rc": 0,
  "results": [
    "Removed: httpd-2.4.37-51.module+el8.7.0+18499+2e106f0b.5.x86_64",
    "Removed: mod_http2-1.15.7-5.module+el8.7.0+18499+2e106f0b.4.x86_64"
  ]
}
```

- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- Lab: Ansible Module 實作練習
 - Common Modules Reference
 - Deploy WebApp
 - Backup/Restore File
- Lab: Ansible Plugin 實作練習
- Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習

藉由此實作使用 Ansible Module 在 Managed Node 上實現備份還原檔案功能。

備份檔案步驟:

1. 在 Managed Node 上建立備份用目錄
2. 從來源目錄把檔案複製到備份用目錄

還原檔案步驟

1. 將檔案從備份用目錄複製回來源目錄

* 此實作需要先部署 httpd 至 Managed Node (可參考前一小節實作)

藉由此實作使用 Ansible Module 在 Managed Node 上實現備份還原檔案功能。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini module/2_backup/inventory 3. cd module/2_backup/ 4. 確認 hosts.ini 設定正確。
Step2	執行 ad-hoc 指令備份檔案
	1. ansible linux -m file -a "path={{ backup_files.dest }} state=directory" 2. ansible linux -m copy -a "src={{ backup_files.src }} dest={{ backup_files.dest }} remote_src=yes"
Step3	執行 ad-hoc 指令還原檔案
	1. ansible linux -m copy -a "src={{ backup_files.dest }}/{{ backup_files.src win_basename trim }} dest={{ backup_files.src dirname }} remote_src=true"

```
[linux]
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```

藉由此實作使用 Ansible Module 在 Managed Node 上實現備份還原檔案功能。

步驟	說明
Step1	複製 Inventory 到實作目錄
	<div>1. cd ansible-concepts</div> <div>2. cp hosts.ini module/2_backup/inventory</div> <div>3. cd module/2_backup/</div> <div>4. 確認 hosts.ini 設定正確。</div>
Step2	執行 ad-hoc 指令備份檔案
	<div>1. ansible linux -m file -a "path={{ backup_files.dest }} state=directory"</div> <div>2. ansible linux -m copy -a "src={{ backup_files.src }} dest={{ backup_files.dest }} remote_src=yes"</div>
Step3	執行 ad-hoc 指令還原檔案
	<div>1. ansible linux -m copy -a "src={{ backup_files.dest }}/{{ backup_files.src win_basename trim }} dest={{ backup_files.src dirname }} remote_src=true"</div>

```
[speaker1@ansgbtn 2_backup]$ ansible linux -m file -a "path={{ backup_files.dest }} state=directory"
speaker01 | CHANGED => {
  "changed": true,
  "gid": 0,
  "group": "root",
  "mode": "0755",
  "owner": "root",
  "path": "/tmp/backup",
  "secontext": "unconfined_u:object_r:user_tmp_t:s0",
  "size": 6,
  "state": "directory",
  "uid": 0
}
[speaker1@ansgbtn 2_backup]$ ansible linux -m copy -a "src={{ backup_files.src }} dest={{ backup_files.dest }} remote_src=yes"
speaker01 | CHANGED => {
  "changed": true,
  "checksum": "65abe83aa6d09c36dd39cbc857c52b55ff39a26a",
  "dest": "/tmp/backup/index.html",
  "gid": 0,
  "group": "root",
  "md5sum": "a3636f12bbc1b3e2b9e8f4d917886460",
  "mode": "0644",
  "owner": "root",
  "secontext": "system_u:object_r:httpd_sys_content_t:s0",
  "size": 66,
  "src": "/var/www/html/index.html",
  "state": "file",
  "uid": 0
}
```

於 Managed Node 上確認是否備份成功

```
[root@speaker01 ~]# cat /tmp/backup/index.html
THSRC Examples!!!

The following is custom message:

Hello THSRC!
```



藉由此實作使用 Ansible Module 在 Managed Node 上實現備份還原檔案功能。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini module/2_backup/inventory 3. cd module/2_backup/ 4. 確認 hosts.ini 設定正確。
Step2	執行 ad-hoc 指令備份檔案
	1. ansible linux -m file -a "path={{ backup_files.dest }} state=directory" 2. ansible linux -m copy -a "src={{ backup_files.src }} dest={{ backup_files.dest }} remote_src=yes"
Step3	執行 ad-hoc 指令還原檔案
	1. ansible linux -m copy -a "src={{ backup_files.dest }}/{{ backup_files.src win_basename trim }} dest={{ backup_files.src dirname }} remote_src=true"

```
[speaker1@ansgbtn 2_backup]$ ansible linux -m copy -a "src={{ backup_files.dest }}/{{ backup_files.src|win_basename|trim }} dest={{ backup_files.src|dirname }} remote_src=true"
speaker01 | SUCCESS => {
  "changed": false,
  "checksum": "65abe83aa6d09c36dd39cbc857c52b55ff39a26a",
  "dest": "/var/www/html/index.html",
  "gid": 0,
  "group": "root",
  "md5sum": "a3636f12bbc1b3e2b9e8f4d917886460",
  "mode": "0644",
  "owner": "root",
  "secontext": "system_u:object_r:httpd_sys_content_t:s0",
  "size": 66,
  "src": "/tmp/backup/index.html",
  "state": "file",
  "uid": 0
}
```



- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- ✓ Lab: Ansible Module 實作練習
- Lab: Ansible Plugin 實作練習
 - Lookup Plugin
 - Callback Plugin
- Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習

藉由此實作使用 Ansible Lookup Plugin 抓取員工資料檔案並顯示出來。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini plugin/1_lookup/inventory 3. cd plugin/1_lookup/ 4. 確認 hosts.ini 設定正確。
Step2	執行 ad-hoc 指令顯示出員工資料
	1. ansible linux -m debug -a "msg={{ lookup('file', './file/employ_lists.json') }}"

```
[linux]
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```

藉由此實作使用 Ansible Lookup Plugin 抓取員工資料檔案並顯示出來。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini plugin/1_lookup/inventory 3. cd plugin/1_lookup/ 4. 確認 hosts.ini 設定正確。
Step2	執行 ad-hoc 指令顯示出員工資料
	1. ansible linux -m debug -a "msg={{ lookup('file', './file/employ_lists.json') }}"

```
[speaker1@ansgbtn 1_lookup]$ ansible linux -m debug -a "msg={{ lookup('file', '
speaker01 | SUCCESS=> {
  "msg": {
    "employ_list": [
      {
        "department": "Operation",
        "name": "Alice",
        "onboard_date": "2023/01/01"
      },
      {
        "department": "Sales",
        "name": "Bob",
        "onboard_date": "2021/12/25"
      },
      {
        "department": "Development",
        "name": "Charlie",
        "onboard_date": "2022/03/05",
        "salary": 75000
      }
    ]
  }
}
```

- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- ✓ Lab: Ansible Module 實作練習
- Lab: Ansible Plugin 實作練習
 - Lookup Plugin
 - Callback Plugin
- Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習

藉由此實作使用 Ansible Callback Plugin 取得各 Task 執行花費的時間。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini plugin/2_callback/inventory 3. cd plugin/2_callback/ 4. 確認 hosts.ini 設定正確。
Step2	修改 Ansible Configuration
	1. vim ansible.cfg, 把以下這行拿掉註解 callback_whitelist = timer, profile_tasks
Step3	執行部署 WebApp 腳本, 可以看到各 Task 執行花費時間
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml

```
[linux]
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```

藉由此實作使用 Ansible Callback Plugin 取得各 Task 執行花費的時間。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini plugin/2_callback/inventory 3. cd plugin/2_callback/ 4. 確認 hosts.ini 設定正確。
Step2	修改 Ansible Configuration
	1. vim ansible.cfg, 把以下這行拿掉註解 callback_whitelist = timer, profile_tasks
Step3	執行部署 WebApp 腳本, 可以看到各 Task 執行花費時間
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml

```
inventory ignore_extensions = ~, .orig, .bak, .ini, .c
callback_whitelist = timer, profile_tasks
#log_path=/var/log/ansible.log
```

藉由此實作使用 Ansible Callback Plugin 取得各 Task 執行花費的時間。

步驟	說明
Step1	複製 Inventory 到實作目錄
	<div>1. cd ansible-concepts</div> <div>2. cp hosts.ini plugin/2_callback/inventory</div> <div>3. cd plugin/2_callback/</div> <div>4. 確認 hosts.ini 設定正確。</div>
Step2	修改 Ansible Configuration
	<div>1. vim ansible.cfg, 把以下這行拿掉註解</div> <div>callback_whitelist = timer, profile_tasks</div>
Step3	執行部署 WebApp 腳本, 可以看到各 Task 執行花費時間
	<div>1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml</div>

```
[speaker1@ansible2_callback]$ ansible-playbook -i inventory/hosts.ini deploy-httpd.yml
PLAY [Deploy httpd Server] *****
Wednesday 03 May 2023  01:12:38 +0800 (0:00:00.049)    0:00:00.049 *****

TASK [Install httpd and start] *****
changed: [speaker01]
Wednesday 03 May 2023  01:12:55 +0800 (0:00:16.938)    0:00:16.988 *****

TASK [Customize index.html file] *****
ok: [speaker01]
Wednesday 03 May 2023  01:12:56 +0800 (0:00:01.779)    0:00:18.767 *****

TASK [Start httpd service] *****
changed: [speaker01]
Wednesday 03 May 2023  01:12:59 +0800 (0:00:02.334)    0:00:21.102 *****

TASK [Allow http connection through firewall] *****
ok: [speaker01]

PLAY RECAP *****
speaker01 : ok=4  changed=2  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0

Wednesday 03 May 2023  01:13:00 +0800 (0:00:01.226)    0:00:22.328 *****

Install httpd and start ----- 16.94
Start httpd service ----- 2.33
Customize index.html file ----- 1.78
Allow http connection through firewall ----- 1.23
Playbook run took 0 days, 0 hours, 0 minutes, 22 seconds
```



- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- ✓ Lab: Ansible Module 實作練習
- ✓ Lab: Ansible Plugin 實作練習
- Lab: Ansible Vault 實作練習
 - Encrypt the variable file
 - Encrypt the specific string
- Lab: Ansible Troubleshooting 練習

藉由此實作將使用者檔案透過 Ansible Vault 加密, 接著驗證 Ansible 能讀取密文檔並建立新使用者。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini vault/1_users/inventory 3. cd vault/1_users 4. 確認 hosts.ini 設定正確。
Step2	加密使用者資料
	1. ansible-vault encrypt ./vars/userlist-plaintext.yml
Step3	執行 playbooks, 讓 Ansible 讀取密文檔建立新使用者
	1. ansible-playbook -i inventory/hosts.ini create-linuxuser.yml --ask-vault-pass

```
[linux]
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```

藉由此實作將使用者檔案透過 Ansible Vault 加密, 接著驗證 Ansible 能讀取密文檔並建立新使用者。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini vault/1_users/inventory 3. cd vault/1_users 4. 確認 hosts.ini 設定正確。
Step2	加密使用者資料
	1. ansible-vault encrypt ./vars/userlist-plaintext.yml
Step3	執行 playbooks, 讓 Ansible 讀取密文檔建立新使用者
	1. ansible-playbook -i inventory/hosts.ini create-linuxuser.yml --ask-vault-pass

```
[speaker1@ansgbtn 1_users]$ ansible-vault encrypt ./vars/userlist-plaintext.yml
New Vault password:
Confirm New Vault password:
Encryption successful
[speaker1@ansgbtn 1_users]$ cat ./vars/userlist-plaintext.yml
$ANSIBLE_VAULT;1.1;AES256
62376262386466346161373566313436373637616262653136376162313738383962616662366435
6365383562333865303361376466653935643261663861370a396462383231633564393166336465
35663337326631363563633863636239366336616161363162373964313263343031613036353235
6264393134363261630a623861623934393461396161303864343232396661336533613636346238
34636234633666353437656330303737303465393335376636303234376435306638333762613633
34363465333435303461343734616265346265623062653764353930393133643266656337373430
38343632333633356432313364363232386338623338393230306333313061356238663730376537
34353233353461386335343731323138316233363632396333646562363966616164663637643764
65383662626465396462353436393161373239343662633432376434666637313938373836616639
3733363623530646530323030393662313635633232356237616630613336343539663465326662
34336538616665353431663335373630626463353866396335313132363433316430626562326533
64336235643764626365353361616662396362633062613262616234653237353366646530653739
62396137386265343461376361346430616637643264636563333363653538393266
```

藉由此實作將使用者檔案透過 Ansible Vault 加密, 接著驗證 Ansible 能讀取密文檔並建立新使用者。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini vault/1_users/inventory 3. cd vault/1_users 4. 確認 hosts.ini 設定正確。
Step2	加密使用者資料
	1. ansible-vault encrypt ./vars/userlist-plaintext.yml
Step3	執行 playbooks, 讓 Ansible 讀取密文檔建立新使用者
	1. ansible-playbook -i inventory/hosts.ini create-linuxuser.yml --ask-vault-pass

```
[speaker1@ansgbtn 1_users]$ ansible-playbook -i inventory/hosts.ini create-linuxuser.yml --ask-vault-pass
Vault password:

PLAY [Create Linux User] *****

TASK [Create groups] *****
changed: [speaker01] => (item=None)
ok: [speaker01] => (item=None)
ok: [speaker01] => (item=None)
changed: [speaker01]

TASK [Create Users] *****
changed: [speaker01] => (item=None)
changed: [speaker01] => (item=None)
changed: [speaker01] => (item=None)
changed: [speaker01]

PLAY RECAP *****
speaker01 : ok=2 changed=2 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
```



- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- ✓ Lab: Ansible Module 實作練習
- ✓ Lab: Ansible Plugin 實作練習
- Lab: Ansible Vault 實作練習
 - Encrypt the variable file
 - **Encrypt the specific string**
- Lab: Ansible Troubleshooting 練習

藉由此實作將使用者密碼透過 Ansible Vault 加密(只加密密碼), 接著驗證 Ansible 能讀取密文檔並建立新使用者。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini vault/2_user-only-password/inventory 3. cd vault/2_user-only-password 4. 確認 hosts.ini 設定正確。
Step2	加密使用者密碼
	1. ansible-vault encrypt_string 'P@ssw0rd' --name password 2. ansible-vault encrypt_string '123456' --name password 3. ansible-vault encrypt_string '45678' --name password
Step3	修改使用者資料, 貼上剛剛產生的密文
	1. vim ./vars/userlist-plaintext.yml - username: "test1" password: !vault \$ANSIBLE_VAULT;1.1;AES2563530623965333131666 237613661353964396362646538373635666164633 96534636661323535393461366266626338633164 ...

```
[linux]  
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```



藉由此實作將使用者密碼透過 Ansible Vault 加密(只加密密碼), 接著驗證 Ansible 能讀取密文檔並建立新使用者。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini vault/2_user-only-password/inventory 3. cd vault/2_user-only-password 4. 確認 hosts.ini 設定正確。
Step2	加密使用者密碼
	1. ansible-vault encrypt_string '密碼1' --name password 2. ansible-vault encrypt_string '密碼2' --name password 3. ansible-vault encrypt_string '密碼3' --name password
Step3	修改使用者資料, 貼上剛剛產生的密文
	1. vim ./vars/userlist-plaintext.yml - username: "test1" password: !vault \$ANSIBLE_VAULT;1.1;AES2563530623965333131666 237613661353964396362646538373635666164633 96534636661323535393461366266626338633164 ...

```
[speaker1@ansgbtn 2_user-only-password]$ ansible-vault encrypt_string '123456'  
New Vault password:  
Confirm New Vault password:  
password: !vault |  
$ANSIBLE_VAULT;1.1;AES256  
6336333565333165383862616134613235626534353132393862393038393132613837  
3838656434666563366332316530356134393463613163660a66636630623436653565  
3338643037626361313033613234656366323462316239363036303533313866613766  
6564333831336431350a65343238343038346534656462323335326432333062373634  
3433  
Encryption successful
```


藉由此實作將使用者密碼透過 Ansible Vault 加密(只加密密碼), 接著驗證 Ansible 能讀取密文檔並建立新使用者。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini vault/2_user-only-password/inventory 3. cd vault/2_user-only-password 4. 確認 hosts.ini 設定正確。
Step2	加密使用者密碼
	1. ansible-vault encrypt_string '密碼1' --name password 2. ansible-vault encrypt_string '密碼2' --name password 3. ansible-vault encrypt_string '密碼3' --name password
Step3	修改使用者資料, 貼上剛剛產生的密文
	1. vim ./vars/userlist-plaintext.yml - username: "test1" password: !vault \$ANSIBLE_VAULT;1.1;AES2563530623965333131666 237613661353964396362646538373635666164633 96534636661323535393461366266626338633164 ...

```
user_list:
- username: "test1"
  password: !vault |
    $ANSIBLE_VAULT;1.1;AES256
    35306239653331316662376136613539643963626465383736356661646339653463666132353539
    3461366266626338633164613633363136653931373132300a363230366163363162656666306533
    38646634333163346566353439353830393439383661363861373633363630303462626664646461
    6530353863383339630a646166643662613065663233646364316638363565363663393131656630
    3634
  group: thsrc
- username: "test2"
  password: !vault |
    $ANSIBLE_VAULT;1.1;AES256
    64343236616135613165333130393062346638323862633366363966643730623935636437373664
    6537643035383634613031323038653065306436333762390a646134386430376437333033396261
    36633133303638303463636630326563343766633934386663653037303335666561373163316631
    6365316139623437360a656661353966383430393637373264653065316139393230666265343432
    6162
  group: thsrc
- username: "test3"
  password: !vault |
    $ANSIBLE_VAULT;1.1;AES256
    3932663236626238626562313836353336334323832656330323433363643363626666356635383135
    313736306532613239663236666438323646336313830310a313661316538393937633533646663
    37633738343331313861653132373333333638636339356164343630353237346363306537646162
    3336373565656231310a666133643634323232643937333465323733306466666138373639623864
    3538
```



藉由此實作將使用者密碼透過 Ansible Vault 加密(只加密密碼), 接著驗證 Ansible 能讀取密文檔並建立新使用者。

步驟	說明
Step4	執行 playbooks, 讓 Ansible 讀取密文檔建立新使用者
1. ansible-playbook -i inventory/hosts.ini create-linuxuser.yml --ask-vault-pass	

```
Vault password:
PLAY [Create Linux User] *****

TASK [Create groups] *****
ok: [speaker01] => (item=None)
ok: [speaker01] => (item=None)
ok: [speaker01] => (item=None)
ok: [speaker01]

TASK [Create Users] *****
ok: [speaker01] => (item=None)
ok: [speaker01] => (item=None)
changed: [speaker01] => (item=None)
changed: [speaker01]

PLAY RECAP *****
speaker01 : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```



- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- ✓ Lab: Ansible Module 實作練習
- ✓ Lab: Ansible Plugin 實作練習
- ✓ Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習
 - Error Scenario 1
 - Error Scenario 2
 - Error Scenario 3

此實作的 Playbooks 為顯示主機資訊，執行後發生錯誤，請嘗試修復。

此實作共有 **1** 個錯誤需要修改。

```
[speaker1@ansgbtn 1_error]$ ansible-playbook -i inventory/hosts.ini display_info.yml

PLAY [Get the informaion of machines] *****

TASK [Display machine information] *****
fatal: [speaker01]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'app_name' is undefined\n\nThe error\n-concept-lab/troubleshooting/1_error/display_info.yml': line 7, column 5, but may\nbe elsewhere in the file depending on the exact syntax problem.\n\nTasks:\n - name: Display machine information\n   ^ here\n"}

PLAY RECAP *****
speaker01          : ok=0    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
```

此實作的 Playbooks, 執行後發生錯誤, 主要是少了 app_name 這個變數, 需要在 inventory variable 補上這個變數。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/1_error/inventory 3. cd troubleshooting/1_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 查看錯誤訊息。
	1. ansible-playbook -i inventory/hosts.ini display_info.yml
Step3	將缺少的變數補至 inventory variable
	1. vim inventory/group_vars/linux.yml, 加入以下這段 app_name: WEB
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini display_info.yml

```
[linux]
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```

此實作的 Playbooks, 執行後發生錯誤, 主要是少了 app_name 這個變數, 需要在 inventory variable 補上這個變數。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/1_error/inventory 3. cd troubleshooting/1_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 查看錯誤訊息。
	1. ansible-playbook -i inventory/hosts.ini display_info.yml
Step3	將缺少的變數補至 inventory variable
	1. vim inventory/group_vars/linux.yml, 加入以下這段 app_name: WEB
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini display_info.yml

```
[speaker1@ansgbtn 1_error]$ ansible-playbook -i inventory/hosts.ini display_info.yml

PLAY [Get the informaion of machines] *****

TASK [Display machine information] *****
fatal: [speaker01]: FAILED! => {"msg": "The task includes an option with an undefined variable. The erro
-concept-lab/troubleshooting/1_error/display_info.yml": line 7, column 5, but may\nbe elsewhere in the f
tasks:\n  - name: Display machine information\n    ^ here\n"}

PLAY RECAP *****
speaker01                : ok=0    changed=0    unreachable=0    failed=1    skipped=0    rescued=0
```



此實作的 Playbooks, 執行後發生錯誤, 主要是少了 app_name 這個變數, 需要在 inventory variable 補上這個變數。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/1_error/inventory 3. cd troubleshooting/1_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 查看錯誤訊息。
	1. ansible-playbook -i inventory/hosts.ini display_info.yml
Step3	將缺少的變數補至 inventory variable
	1. vim inventory/group_vars/linux.yml, 加入以下這段 app_name: WEB
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini display_info.yml

```
app_name: WEB
system_services:
- httpd
- mysql
~
```

此實作的 Playbooks, 執行後發生錯誤, 主要是少了 app_name 這個變數, 需要在 inventory variable 補上這個變數。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/1_error/inventory 3. cd troubleshooting/1_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 查看錯誤訊息。
	1. ansible-playbook -i inventory/hosts.ini display_info.yml
Step3	將缺少的變數補至 inventory variable
	1. vim inventory/group_vars/linux.yml, 加入以下這段 app_name: WEB
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini display_info.yml

```
[speaker1@ansgbtn 1_error]$ ansible-playbook -i inventory/hosts.ini display_info.yml
PLAY [Get the informaion of machines] *****

TASK [Display machine information] *****
ok: [speaker01] => {
  "msg": [
    "Machine Name: speaker01",
    "Application Name: WEB",
    "Service Name: ['httpd', 'mysql']",
    "vCenter Datacenter: PDISC",
    "VM Folder Name: Student01"
  ]
}

PLAY RECAP *****
speaker01 : ok=1  changed=0  unreachable=0  failed=0  skipped=0  res
```



- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- ✓ Lab: Ansible Module 實作練習
- ✓ Lab: Ansible Plugin 實作練習
- ✓ Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習
 - Error Scenario 1
 - **Error Scenario 2**
 - Error Scenario 3

此實作的 Playbooks 為自動部署 WebAPP, 執行後發生錯誤, 請嘗試修復。

此實作共有 **3** 個錯誤需要修改。

```
[speaker1@ansgbtn 2_error]$ ansible-playbook -i inventory/hosts.ini deploy-httpd.yml
ERROR! couldn't resolve module/action 'templates'. This often indicates a misspelling, missing collection, or incorrect module path.

The error appears to be in '/home/speaker1/ansible-concept-lab/troubleshooting/2_error/deploy-httpd.yml': line 10, column 5, but may
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

- name: Customize index.html file
  ^ here
```

此實作的 Playbooks, 執行後發生錯誤, 主要錯誤如下

- template module 名稱拼錯
- httpd service 誤打成 apache2

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/2_error/inventory 3. cd troubleshooting/2_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 查看錯誤訊息。
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml
Step3	修復錯誤
	1. vim ideploy-httpd.yml, 修改以下部分 11 templates => template 18 apache2 => httpd 32 apache2 => httpd
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml

```
[linux]
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```



此實作的 Playbooks, 執行後發生錯誤, 主要錯誤如下

- template module 名稱拼錯
- httpd service 誤打成 apache2

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/2_error/inventory 3. cd troubleshooting/2_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 查看錯誤訊息。
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml
Step3	修復錯誤
	1. vim ideploy-httpd.yml, 修改以下部分 11 templates => template 18 apache2 => httpd 32 apache2 => httpd
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml

```
[speaker1@ansgbtn 2_error]$ ansible-playbook -i inventory/hosts.ini deploy-h
ERROR! couldn't resolve module/action 'templates'. This often indicates a mi

The error appears to be in '/home/speaker1/ansible-concept-lab/troubleshooti
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

- name: Customize index.html file
  ^ here
```

此實作的 Playbooks, 執行後發生錯誤, 主要錯誤如下

- template module 名稱拼錯
- httpd service 誤打成 apache2

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/2_error/inventory 3. cd troubleshooting/2_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 查看錯誤訊息。
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml
Step3	修復錯誤
	1. vim ideploy-httpd.yml, 修改以下部分 11 templates => template 18 apache2 => httpd 32 apache2 => httpd
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml

```
10 - name: Customize index.html file
11   template:
12     src: templates/index.html.j2
13     dest: /var/www/html/index.html
14     notify: Restart httpd service
15
16 - name: Start httpd service
17   systemd:
18     name: httpd
19     enabled: yes
20     state: started
21
```



此實作的 Playbooks, 執行後發生錯誤, 主要錯誤如下

- template module 名稱拼錯
- httpd service 誤打成 apache2

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/2_error/inventory 3. cd troubleshooting/2_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 查看錯誤訊息。
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml
Step3	修復錯誤
	1. vim ideploy-httpd.yml, 修改以下部分 11 templates => template 18 apache2 => httpd 32 apache2 => httpd
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini deploy-httpd.yml

```
[speaker1@ansgbtn 2_error]$ ansible-playbook -i inventory/hosts.ini deploy-httpd.yml

PLAY [Deploy httpd Server] *****

TASK [Install httpd and start] *****
ok: [speaker01]

TASK [Customize index.html file] *****
ok: [speaker01]

TASK [Start httpd service] *****
ok: [speaker01]

TASK [Allow http connection through firewall] *****
ok: [speaker01]

PLAY RECAP *****
speaker01 : ok=4    changed=0    unreachable=0    failed=0    skipped=0    resc...
```



- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- ✓ Lab: Ansible Module 實作練習
- ✓ Lab: Ansible Plugin 實作練習
- ✓ Lab: Ansible Vault 實作練習
- Lab: Ansible Troubleshooting 練習
 - Error Scenario 1
 - Error Scenario 2
 - Error Scenario 3

此實作的 Playbooks 為自動化備份檔案，執行後並沒有發生錯誤，但是沒有備份成功，請嘗試修復。

此實作共有 **1** 個錯誤需要修改。

*** 此實作需要先部署 httpd 至 Managed Node。**

```
[speaker1@ansgbtn 3_error]$ ansible-playbook -i inventory/hosts.ini backup.yml

PLAY [Backup the data on Managed Node] *****

TASK [Create backup dictionaries on control node] *****
changed: [speaker01]

TASK [Backup data on Managed Node] *****
changed: [speaker01]

PLAY RECAP *****
speaker01 : ok=2    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

```
[root@speaker01 ~]# ls /tmp/backup
ls: cannot access '/tmp/backup': No such file or directory
[root@speaker01 ~]#
```


此實作的 Playbooks, 問題主要是在變數覆蓋問題, 導致inventory variable 設定的備份參數 backup_files 被覆蓋, 需要將 playbooks 裡預設變數移除掉。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/3_error/inventory 3. cd troubleshooting/3_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 並使用 step by step 方式一步步檢查是哪個 task 有問題
	1. ansible-playbook -i inventory/hosts.ini backup.yml --step
Step3	檢查出來在第一個 Task 就出問題, 因此編輯 playbooks, 修復該 task 問題。
	1. vim backup.yml, 移除以下部分: 4 vars: 5 backup_files: 6 src: "/var/www/html/index.html" # folder name or file name 7 dest: "/mnt/backup" # folder name
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini backup.yml

```
[linux]
speaker01 ansible_user=root ansible_connection=ssh machine_type=virtual
```


此實作的 Playbooks, 問題主要是在變數覆蓋問題, 導致inventory variable 設定的備份參數 backup_files 被覆蓋, 需要將 playbooks 裡預設變數移除掉。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/3_error/inventory 3. cd troubleshooting/3_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 並使用 step by step 方式一步步檢查是哪個 task 有問題
	1. ansible-playbook -i inventory/hosts.ini backup.yml --step
Step3	檢查出來在第一個 Task 就出問題, 因此編輯 playbooks, 修復該 task 問題。
	1. vim backup.yml, 移除以下部分: 4 vars: 5 backup_files: 6 src: "/var/www/html/index.html" # folder name or file name 7 dest: "/mnt/backup" # folder name
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini backup.yml

```
[speaker1@ansgbtn 3_error]$ ansible-playbook -i inventory/hosts.ini backup.yml --step
PLAY [Backup the data on Managed Node] *****
Perform task: TASK: Create backup dictories on control node (N)o/(y)es/(c)ontinue: y
Perform task: TASK: Create backup dictories on control node (N)o/(y)es/(c)ontinue: *****
TASK [Create backup dictories on control node] *****
ok: [speaker01]
```



此實作的 Playbooks, 問題主要是在變數覆蓋問題, 導致inventory variable 設定的備份參數 backup_files 被覆蓋, 需要將 playbooks 裡預設變數移除掉。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/3_error/inventory 3. cd troubleshooting/3_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 並使用 step by step 方式一步步檢查是哪個 task 有問題
	1. ansible-playbook -i inventory/hosts.ini backup.yml --step
Step3	檢查出來在第一個 Task 就出問題, 因此編輯 playbooks, 修復該 task 問題。
	1. vim backup.yml, 移除以下部分: 4 vars: 5 backup_files: 6 src: "/var/www/html/index.html" # folder name or file name 7 dest: "/mnt/backup" # folder name
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini backup.yml

```
- name: Backup the data on Managed Node
  hosts: linux
  gather_facts: no
  tasks:
    - name: Create backup dictories on control node
      file:
        path: "{{ backup_files.dest }}"
        state: directory

    - name: Backup data on Managed Node
      copy:
        src: "{{ backup_files.src }}"
        dest: "{{ backup_files.dest }}"
        remote_src: true
```

此實作的 Playbooks, 問題主要是在變數覆蓋問題, 導致inventory variable 設定的備份參數 backup_files 被覆蓋, 需要將 playbooks 裡預設變數移除掉。

步驟	說明
Step1	複製 Inventory 到實作目錄
	1. cd ansible-concepts 2. cp hosts.ini troubleshooting/3_error/inventory 3. cd troubleshooting/3_error 4. 確認 hosts.ini 設定正確。
Step2	執行 playbooks 並使用 step by step 方式一步步檢查是哪個 task 有問題
	1. ansible-playbook -i inventory/hosts.ini backup.yml --step
Step3	檢查出來在第一個 Task 就出問題, 因此編輯 playbooks, 修復該 task 問題。
	1. vim backup.yml, 移除以下部分: 4 vars: 5 backup_files: 6 src: "/var/www/html/index.html" # folder name or file name 7 dest: "/mnt/backup" # folder name
Step4	再次執行 playbooks, 錯誤已消失
	1. ansible-playbook -i inventory/hosts.ini backup.yml

```
[root@speaker01 ~]# ls /tmp/backup
index.html
```

- ✓ Lab: 環境準備
- ✓ Lab: Inventory Variables 實作練習
- ✓ Lab: Ansible Module 實作練習
- ✓ Lab: Ansible Plugin 實作練習
- ✓ Lab: Ansible Vault 實作練習
- ✓ Lab: Ansible Troubleshooting 練習