



Ansible 自动化概述



ansible



Ansible 是一个流行的开源自动化工具，用于配置管理、应用程序部署、任务自动化和 IT 编排。它使用简单的 YAML 文件（称为 Playbooks）来定义自动化任务，并通过 SSH 或 API 与目标系统交互，无需安装额外的代理。

Ansible 的主要特点

1. 无代理架构：

- Ansible 通过 SSH 或 API 与远程设备通信，无需在目标系统上安装任何客户端或守护进程。

2. 简单直观：

- 使用 YAML 编写 Playbooks，语法简单易读，适合初学者和复杂场景的自动化。

3. 跨平台支持：

- 支持管理多种操作系统（如 Linux、Windows、网络设备等）以及云服务（如 AWS、Azure、GCP）。

4. 模块化设计：

- 提供大量内置模块用于任务执行（如文件操作、软件安装、网络配置等），也支持自定义模块。

5. 幂等性：

- 确保任务多次运行的结果一致，不会对系统产生意外影响。



Ansible 的核心概念

1. Inventory :

- 定义目标主机的列表，可以是简单的 INI 文件，也可以是动态 Inventory 脚本。

2. Playbook :

- 自动化任务的配置文件，使用 YAML 格式，定义任务的执行顺序和配置细节。

3. Module :

- 预定义的功能单元，执行具体任务（如安装软件包、管理用户、配置服务）。

4. Role :

- 将 Playbooks、变量、模板等逻辑结构化组织，便于重用和共享。

5. Task :

- Playbook 中的具体操作，调用模块实现单个任务。

6. Handler :

- 只有在被触发时才会运行的任务，用于处理条件操作（如重启服务）。

Ansible 的应用场景

1. 配置管理 :

- 安装软件包、管理用户、设置防火墙规则等。

2. 应用部署 :

- 自动化发布流程，部署多层应用程序。

3. IT 编排 :

- 自动化多个系统间的复杂工作流，例如数据中心管理。



4. 网络自动化：

- 配置网络设备，支持多种供应商（如 Cisco、Juniper、Arista）。

示例：一个简单的 Playbook

```
---  
- name: 安装和启动 Apache 服务  
  hosts: webservers  
  become: yes  
  
  tasks:  
    - name: 安装 Apache  
      apt:  
        name: apache2  
        state: present  
  
    - name: 确保 Apache 服务已启动  
      service:  
        name: apache2  
        state: started  
        enabled: yes
```

常用命令

1. 运行 Playbook：

```
ansible-playbook playbook.yml
```

2. 测试连接：

```
ansible all -m ping -i inventory
```

3. 列出主机：



```
ansible-inventory --list
```

如果需要深入学习，可以从官方文档或社区资源中获取更多示例和实践！



我有學過linux指令，以繁體中文詳細說明怎麼使用ansible、ansible的語法



以下是詳細的 Ansible 使用方式與語法教學，幫助你快速掌握其核心功能。

一、安裝 Ansible

在 Linux 系統中，你可以透過以下指令安裝 Ansible（以 Ubuntu 為例）：

```
sudo apt update  
sudo apt install ansible -y
```

驗證安裝是否成功：

```
ansible --version
```

二、建立 Inventory 檔案

Inventory 檔案用來定義管理的主機列表，通常位於 `~/etc/ansible/hosts`，格式如下：



```
[webservers]
192.168.1.10
192.168.1.11

[databases]
192.168.1.20
```

- ``webservers`` 與 ``databases`` 是主機群組名稱。
- 每行寫一台主機，可以是 IP 或域名。

如果需要自訂檔案位置，例如 ``my_inventory``，在執行指令時可指定：

```
ansible -i my_inventory all -m ping
```

三、測試與基本指令

1. 測試主機連線

確認 Inventory 中的主機是否可連線：

```
ansible all -m ping
```

- ``all`` 表示所有主機。
- ``-m ping`` 使用 ``ping`` 模組測試。

2. 執行單一指令

在所有主機上執行 ``uname -a``：

```
ansible all -m shell -a "uname -a"
```

- ``-m shell`` 使用 Shell 模組。



- `-a` 指定要執行的指令。

3. 指定主機群組執行

只對 `webserver` 群組執行指令：

```
ansible webserver -m shell -a "uptime"
```

四、撰寫 Playbook

Playbook 是使用 YAML 格式的自動化腳本，描述多步驟的任務流程。

基本語法結構

以下是一個簡單的 Playbook：

```
---
- name: 安裝 Apache 並啟動服務
  hosts: webserver
  become: yes # 提升為 root 權限
  tasks:
    - name: 安裝 Apache
      apt:
        name: apache2
        state: present

    - name: 啟動 Apache 服務
      service:
        name: apache2
        state: started
        enabled: yes
```

Playbook 的主要語法元素



1. `hosts`

指定執行任務的主機或群組。

2. `become`

如果需要 root 權限，設為 `yes`。

3. `tasks`

定義執行的任務，每個任務都需要 `name` 和對應的模組。

4. 模組

常用模組包括：

- `apt`：管理 Debian 系統的套件。
- `yum`：管理 Red Hat 系統的套件。
- `service`：管理服務狀態。
- `copy`：傳輸檔案。
- `template`：使用 Jinja2 模板。

執行 Playbook

執行 Playbook 並應用到主機：

```
ansible-playbook playbook.yml
```

五、模組範例

1. 檔案管理

- 建立目錄：

```
- name: 建立目錄
  file:
    path: /var/www/html
```



```
state: directory
mode: '0755'
```

- 複製檔案：

```
- name: 複製檔案到主機
  copy:
    src: /local/path/to/file
    dest: /remote/path/to/file
```

2. 使用模板 使用 Jinja2 模板自動生成檔案：

```
- name: 部署配置檔案
  template:
    src: apache.conf.j2
    dest: /etc/apache2/sites-available/000-default.conf
```

3. 服務管理 啟動或停止服務：

```
- name: 確保服務已啟動
  service:
    name: apache2
    state: started
    enabled: yes
```

六、進階功能

1. 變數 (Variables) 變數可以在 Playbook 中定義，或者放在 Inventory 中。

```
vars:
  http_port: 8080
tasks:
  - name: 使用變數配置服務
```




```
lineinfile:
  path: /etc/apache2/ports.conf
  line: "Listen {{ http_port }}"
```

2. **Handler** 用於條件觸發操作（如服務重啟）：

```
tasks:
  - name: 更新配置檔案
    copy:
      src: new_config.conf
      dest: /etc/service/config.conf
    notify:
      - restart service

handlers:
  - name: restart service
    service:
      name: my_service
      state: restarted
```

3. **條件判斷** 根據條件執行任務：

```
tasks:
  - name: 僅在 CentOS 系統上執行
    yum:
      name: httpd
      state: present
    when: ansible_os_family == "RedHat"
```

七、安裝常見錯誤與解決

1. 主機無法連線

- 確認目標主機的 SSH 配置正確，並將 SSH 金鑰加入目標主機。



- 測試 SSH：

```
ssh user@192.168.1.10
```

2. 權限問題

- 在 Playbook 中加入 ``become: yes`` 提升權限。
- 確保執行者有 ``sudo`` 權限。

3. 模組缺失

- 安裝所需的 Ansible 模組（如 Python 模組）。

```
sudo apt install python3-apt
```

Ansible 是非常靈活的工具，適用於各種自動化場景，建議可以從官方文件與範例入手，針對你的實際需求進行練習！