🧪Lab 6 — Introduction to Ansible Playbooks & Writing Your First Playbook

Author: *Sandeep Kumar Sharma*

🕑Learning Objectives

In this lab, you will learn: - What an Ansible playbook is - Why playbooks are used - Structure and format of a playbook - How to write a simple YAML playbook - How to install Apache on managed nodes using a playbook

---

👙Learning Outcomes

After completing this lab, you will: - Understand core playbook components - Write and execute basic playbooks - Know the difference between tasks, modules, variables, and hosts - Deploy software across multiple machines using a single YAML file

---

🧰What Is an Ansible Playbook?

A **playbook** is a YAML file that defines automation tasks to be executed on one or more managed nodes. Playbooks contain: - A list of plays - Each play contains multiple tasks - Each task uses an Ansible module to perform an action

Playbooks help to automate complex operations easily and repeatedly.

🕯️ Why Do We Use Playbooks?

Playbooks are used because: - Ad-hoc commands are good for quick tasks - But real automation requires repeatable, version-controlled instructions - Playbooks ensure consistency - Playbooks support variables, templates, handlers, loops, conditions, roles, etc.

Playbooks make automation predictable, reusable, and maintainable.

🧰Playbook File Format (YAML Basics)

Playbooks are written in **YAML**.

A YAML file must follow these rules: - Use spaces, not tabs - File must start with `---` - Use key-value format - Indentation defines structure

Example:

```
---
key: value
list:
  - item1
  - item2
```

🧱Structure of an Ansible Playbook

A typical playbook structure:

```
---
- name: Name of the play
  hosts: group_or_hosts
  become: yes/no
  vars:
    variable_name: value

  tasks:
    - name: Task description
      module_name:
        option1: value
        option2: value
```

Explanation of key fields:

- **name** — description of the play
- **hosts** — target machines (e.g., dev, webservers, all)
- **become** — whether to use sudo
- **vars** — variables
- **tasks** — steps to execute
- **module** — each task uses an Ansible module (like apt, yum, file, copy, service)

🧪Hands-On: Writing a Simple Playbook to Install Apache

This playbook will install Apache on: - *Ubuntu*: `apache2` - *Amazon Linux*: `httpd`

Create the playbook file:

```
nano install-apache.yml
```

Add:

```
---
- name: Install Apache Web Server
  hosts: dev
  become: yes
  gather_facts: yes

  tasks:
    - name: Install Apache on Ubuntu/Amazon Linux
      package:
        name: "{{ 'apache2' if ansible_facts['os_family'] == 'Debian' else
'httpd' }}"
        state: present

    - name: Start Apache Service
      service:
        name: "{{ 'apache2' if ansible_facts['os_family'] == 'Debian' else
'httpd' }}"
        state: started
        enabled: yes
```

📩Run the Playbook

Execute from master:

```
ansible-playbook install-apache.yml
```

Expected: - Apache installed - Service started and enabled

🧪 Verify Installation

Ubuntu:

```
systemctl status apache2
```

Amazon Linux:

```
systemctl status httpd
```

Or use Ansible:

```
ansible dev -m shell -a "systemctl status apache2 || systemctl status httpd"
```

👙 Lab Summary

In this lab, you learned: - What a playbook is - Why playbooks are important - The structure and format of a playbook - How to write and execute a simple playbook - How to install Apache on multiple machines automatically

Next Lab: 👉**Lab 7 — Using Variables in Playbooks (vars, vars_files, vars_prompt)**