

Ansible Hands-On Guide: Setting Up NGINX & Apache via Playbook

What is Ansible?

Ansible is an **open-source automation tool** used for:

- Configuration management
- Application deployment
- Task automation
- IT infrastructure provisioning

It helps system administrators automate repetitive tasks and manage multiple machines from a single server (called the **control node**).

Why Ansible?

- ◆ **Simple and powerful:** Uses easy-to-read YAML files
- ◆ **Agentless:** No software needed on target machines — just SSH
- ◆ **Idempotent:** Tasks run only if needed — safe to repeat
- ◆ **Fast and lightweight:** Doesn't use heavy resources
- ◆ **Open-source and widely supported**
- ◆ **Easy integration** with cloud, containers, DevOps pipelines

It reduces manual effort, increases reliability, and helps in managing many systems consistently.

How Ansible Works?

1. **Install Ansible** on one system (called the **control node**).
2. Prepare a list of remote machines (**managed nodes**) in an inventory file.
3. Use **SSH** to connect from the control node to the managed nodes.
4. Write **playbooks** in YAML to define what tasks should be done.
5. When a playbook is run, Ansible:
 - Connects to each managed node via SSH
 - Executes the defined tasks using **modules**
 - Ensures tasks are applied in the desired state (install packages, copy files, restart services, etc.)

Example:

- You want to install Apache on 10 servers → Write a playbook → Run it from the control node → Apache is installed on all 10 machines in one shot.

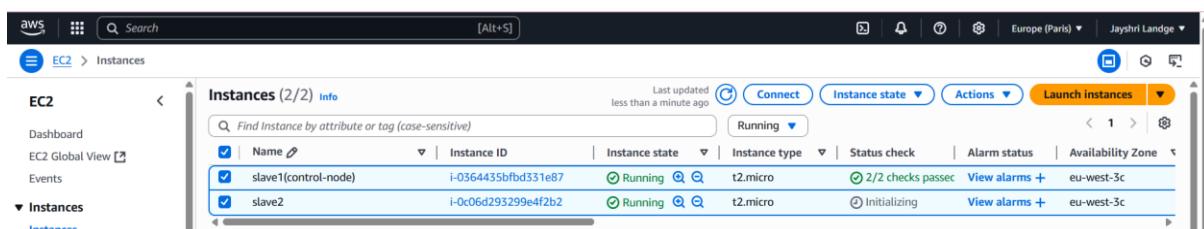
Prerequisites:

- 2 AWS EC2 Instances (Ubuntu-based)
 - Control Node (e.g., slave1)
 - Managed Node (e.g., slave2)
- SSH key pair access
- Ansible installed on control node only
- Python installed on managed node (slave2)

Step-by-Step Implementation

Step 1: Launch EC2 Instances

- Launch **two Ubuntu instances**:
 - slave1 → Control node
 - slave2 → Managed node
- Configure Security Group:
 - Open **Port 22 (SSH)**
 - Open **Port 80 (HTTP)**



Screenshot 1: AWS EC2 dashboard showing running instances.

Step 2: Access Your EC2 Instances

SSH into **Control Node (slave1)**:

```
ssh -i "your-key.pem" ubuntu@<slave1-public-ip>
```

```
ubuntu@ip-172-31-44-234:~$ ssh -i slave1-keypair.pem ubuntu@13.38.118.213
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1029-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Thu Jul 31 09:05:19 UTC 2025

System load: 0.0          Processes:           108
Usage of /: 25.7% of 6.71GB  Users logged in:   1
Memory usage: 21%          IPv4 address for enX0: 172.31.44.234
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Thu Jul 31 08:59:58 2025 from 223.185.43.169
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Screenshot 2: SSH-Connected-to-Slave1

Optional: SSH into **slave2** and install Python (required for Ansible):

```
ssh -i "your-key.pem" ubuntu@<slave2-public-ip>
```

```
ubuntu@ip-172-31-32-119:~$ ssh -i slave2-keypair.pem ubuntu@15.237.128.102
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1029-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Thu Jul 31 09:22:30 UTC 2025

System load: 0.0          Processes:           109
Usage of /: 25.9% of 6.71GB  Users logged in:   1
Memory usage: 21%          IPv4 address for enX0: 172.31.32.119
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Thu Jul 31 09:19:29 2025 from 223.185.43.169
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Screenshot 3: SSH-Connected-to-Slave2

```
sudo apt update
```

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

```
ubuntu@ip-172-31-32-119:~$ sudo apt install python3 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3 is already the newest version (3.12.3-0ubuntu2).
python3 set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 98 not upgraded.
ubuntu@ip-172-31-32-119:~$
```

Screenshot 4: Python installation on slave2.

Step 3: Install Ansible on Control Node

```
sudo apt update
```

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

Check version:

ansible --version

```
ubuntu@ip-172-31-44-234:~$ ansible --version
ansible [core 2.16.3]
  config file = None
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/ubuntu/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Feb 4 2025, 14:48:35) [GCC 13.3.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
```

Screenshot 5: Ansible version output.

Step 4: Enable Passwordless SSH Access

On **slave1**:

ssh-keygen -t rsa

```
ubuntu@ip-172-31-44-234:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:XVTjE7NkfanEeqzap10AYUKyHmqnPvwn0qp6YwiDFzY ubuntu@ip-172-31-44-234
The key's randomart image is:
+---[RSA 3072]----+
| ..o o o.B..|
| o o . * *o|
| o . . = .|
| E o o ..o + .|
| .. = + +S...o |
| + o = + o .. |
| .o.. = o . |
| . +. . ..o. |
| .+. .o. |
+---[SHA256]-----+
```

Screenshot 6: Generate-SSH-RSA-Key

cat ~/.ssh/id_rsa.pub

On **slave2**:

mkdir -p ~/.ssh

vim ~/.ssh/authorized_keys

Paste the key here

chmod 600 ~/.ssh/authorized_keys

On **slave1**:

Test login:

ssh ubuntu@<slave2-ip>

Should log in **without a password**.

```
ubuntu@ip-172-31-44-234:~$ ssh ubuntu@172.31.32.119
The authenticity of host '172.31.32.119 (172.31.32.119)' can't be established.
ED25519 key fingerprint is SHA256:TN26sgmnQDj/ZPGawx4e3S8ALEjxJ5pA6V+KkJqy9J8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.31.32.119' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1029-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Thu Jul 31 09:32:26 UTC 2025

System load: 0.0          Processes:           112
Usage of /: 29.2% of 6.71GB  Users logged in: 1
Memory usage: 22%          IPv4 address for enX0: 172.31.32.119
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.

103 updates can be applied immediately.
66 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Thu Jul 31 09:22:30 2025 from 15.237.128.102
ubuntu@ip-172-31-44-234:
```

Screenshot 7: Successful SSH login

Step 5: Configure Ansible Inventory

Edit the inventory file:

```
sudo vim /etc/ansible/hosts
```

Add:

```
slave1 ansible_host=<IP-of-slave1> ansible_user=ubuntu
```

```
slave2 ansible_host=<IP-of-slave2> ansible_user=ubuntu
```

Test connectivity:

```
ansible -m ping all
```

```
ubuntu@ip-172-31-32-119:~$ ansible -m ping all
The authenticity of host '13.38.118.213 (13.38.118.213)' can't be established.
ED25519 key fingerprint is SHA256:t/4Zw/0H41cXu45Dic8xhZhA3l6b6TSvJwsZpoRsTrQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? slave2 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
```

Screenshot 8: “pong” response

Step 6: Create Ansible Project Directory & Script

```
mkdir ~/ansible
```

```
cd ~/ansible
```

Create the **hello.sh** script:

```
nano hello.sh
```

Paste:

```
#!/bin/bash  
echo "Welcome" > /var/www/html/index.html
```

Make executable:

```
chmod +x hello.sh
```

```
ubuntu@ip-172-31-32-119:~/ansible$ nano hello.sh  
ubuntu@ip-172-31-32-119:~/ansible$ chmod +x hello.sh  
ubuntu@ip-172-31-32-119:~/ansible$ cat hello.sh  
#!/bin/bash  
echo "Welcome" > /var/www/html/index.html  
ubuntu@ip-172-31-32-119:~/ansible$
```

Screenshot 9: Script file & permissions

Step 7: Write the Ansible Playbook

```
nano play.yml  
  
---  
  
- hosts: slave1  
  
become: yes  
  
name: Install Nginx on Slave1  
  
tasks:  
  
- name: Install NGINX  
  
apt:  
  
name: nginx  
  
state: latest  
  
- hosts: slave2  
  
become: yes  
  
name: Install Apache & Deploy Script on Slave2  
  
tasks:  
  
- name: Install Apache Web Server  
  
apt:  
  
name: apache2
```

```
state: latest
- name: Add Hello World web page
  script: hello.sh
```

```
echo "Welcome" > /var/www/html/index.html
ubuntu@ip-172-31-32-119:~/ansible$ nano play.yml
ubuntu@ip-172-31-32-119:~/ansible$ cat play.yml
---
- hosts: slave1
  become: yes
  name: Install Nginx on Slave1
  tasks:
    - name: Install NGINX
      apt:
        name: nginx
        state: latest

- hosts: slave2
  become: yes
  name: Install Apache & Deploy Script on Slave2
  tasks:
    - name: Install Apache Web Server
      apt:
        name: apache2
        state: latest

    - name: Add Hello World web page
      script: hello.sh
```

Screenshot 10: YAML playbook

Step 8: Run the Playbook

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

```
ubuntu@ip-172-31-44-234:~/ansible$ ansible-playbook play.yml
PLAY [Install Nginx on Slave1] *****
TASK [Gathering Facts] *****
ok: [slave1]
TASK [Install NGINX] *****
Changed: [slave1]

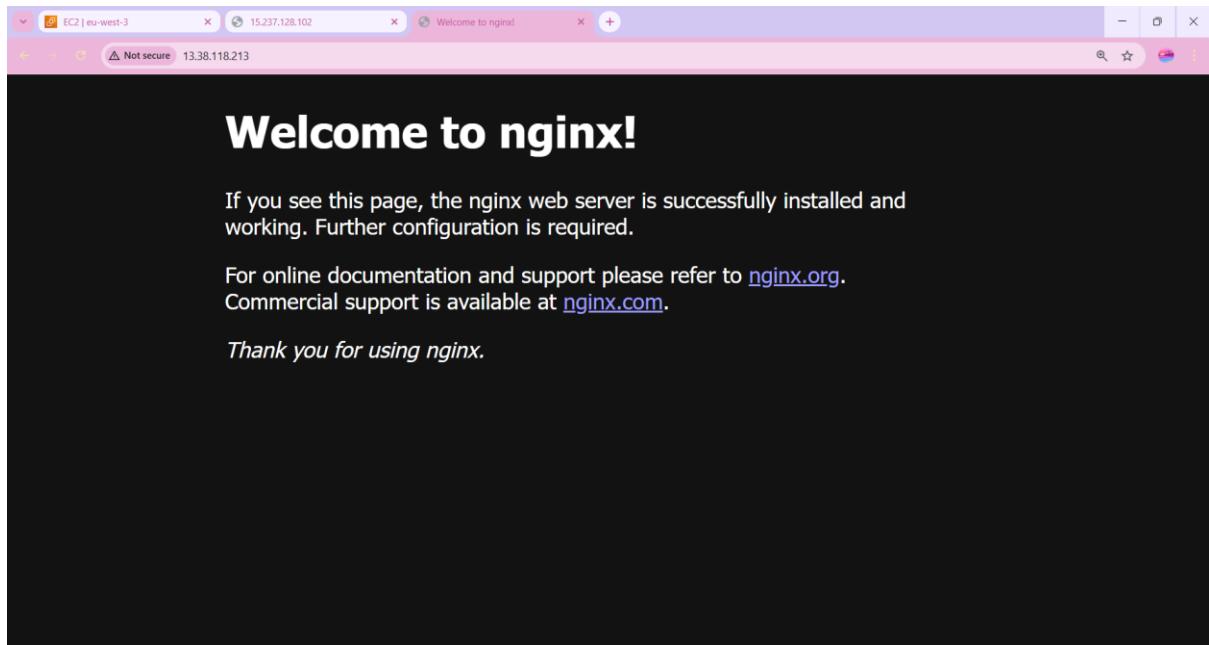
PLAY [Install Apache & Deploy Script on Slave2] *****
TASK [Gathering Facts] *****
ok: [slave2]
TASK [Install Apache Web Server] *****
changed: [slave2]
TASK [Add Hello World web page] *****
changed: [slave2]

PLAY RECAP *****
slave1           : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
slave2           : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

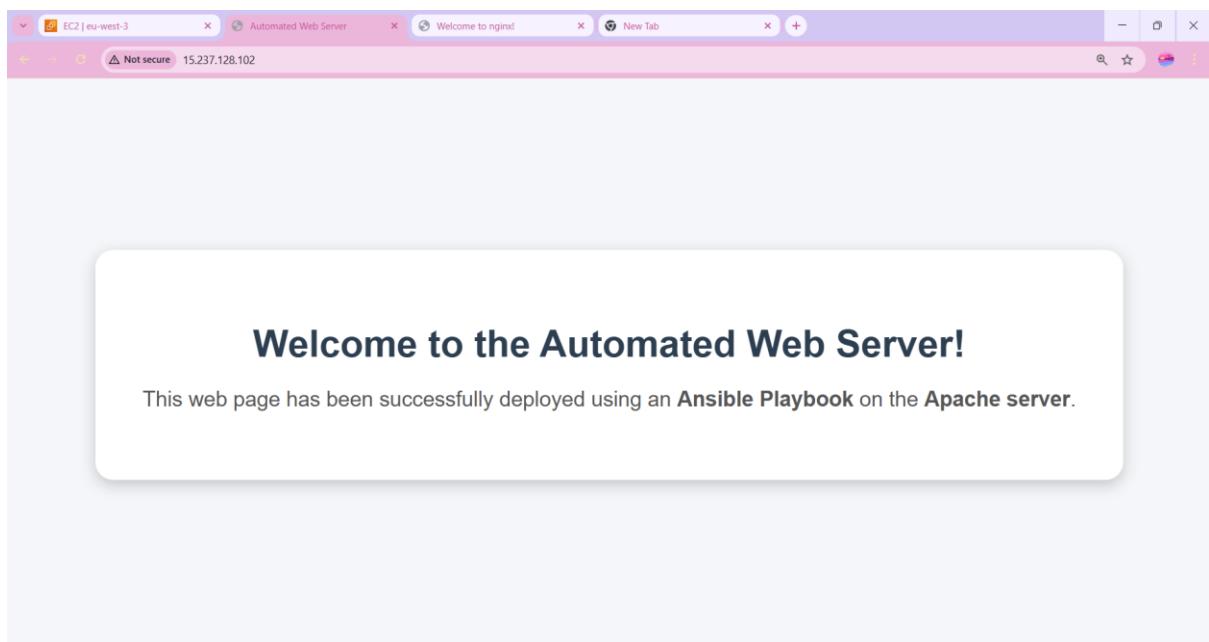
Screenshot 11: Playbook run showing NGINX & Apache installation.

Step 9: Validate in Browser

- Open <http://<slave1-public-ip>> → **Welcome to NGINX**
- Open <http://<slave2-public-ip>> → **Welcome to the automated Web Server**



Screenshot 12: Browser output of slave1



Screenshot 13: Browser output of slave2

Optional: Uninstall Apache

```
ssh ubuntu@<slave2-ip>
```

```
sudo apt-get purge apache2 -y
```

Conclusion

This hands-on lab showcased how **Ansible simplifies automation** by installing and configuring multiple web servers in a single step.

What I learned & achieved:

- Installed **NGINX on Slave1** and **Apache on Slave2** via playbook
- Used the **script module** to deploy a custom hello.sh page
- Set up **passwordless SSH** (agentless architecture in action)
- Strengthened my **Infrastructure as Code (IaC)** and **DevOps skills**

Ansible made it **fast, repeatable, and reliable** — a perfect real-world example of automation in action.