Lab: Automated LAMP Setup Using Ansible

Introduction:

A LAMP (**Linux, Apache, MySQL, PHP/Perl/Python**) stack is a bundle of four different software technologies that developers use to build websites and web applications. LAMP is an acronym for the operating system, Linux; the web server, Apache; the database server, MySQL; and the programming language, PHP.

All four of these technologies are open source, which means they are community maintained and freely available for anyone to use. Developers use LAMP stacks to create, host, and maintain web content. It is a popular solution that powers many of the websites you commonly use today.

Objective:

- **Deploying LAMP Stack using Ansible Playbook**

1. Deploying LAMP Stack using Ansible Playbook

1.1 Let's deploy a LAMP stack and publish your website using Ansible. Here we are going to write the complete Ansible Playbook to Install Apache, MYSQL database, PHP on CentOs machine and deploy your webpage.

1.2 Let's create **lamp.yml** as usual the first step is to target the host.

```
1  ---
2  - name: "Deploying LAMP Stack"
3    hosts: webservers
4    become: yes
5    tasks:
```

Task1---> Let's install latest package.

```
6      - name: "Install Latest Packages"
7        ansible.builtin.yum:
8          name:
9            - httpd
10           - mariadb-server
11           - php
12           - php-mysqlnd
13           - firewalld
14         state: latest
```

Task-2---> Let's add the content to index.html.

```
16     - name: "Copy contents"
17       ansible.builtin.copy:
18         content: "Test LAMP Stack"
19         dest: /var/www/html/index.html
```

Task-3---> Let's Enable and Start the httpd Service

```
20      - name: "Enable and Start HTTPD Service"
21        ansible.builtin.service:
22          name: httpd
23          state: started
24          enabled: yes
```

Task-4---> Let's Enable and Start the mariadb Service.

```
26      - name: "Enable and Start Mariadb Service"
27        ansible.builtin.service:
28          name: mariadb
29          state: started
30          enabled: yes
```

Task-5---> Let's start the firewalld service.

```
32      - name: "Enable and Start Firewalld Service"
33        ansible.builtin.service:
34          name: firewalld
35          state: started
36          enabled: yes
```

Task-6---> Let's add an exception to the httpd service in the firewalld.

```
37      - name: "Add exception in the firewall"
38        ansible.posix.firewalld:
39          service: http
40          permanent: yes
41          state: enabled
42          immediate: yes
```

Task-7---> Let's hit the test page.

```
44  - name: Test page
45    hosts: localhost
46    tasks:
47      - name: acces page
48        uri:
49          url: http://eoc-node3
50          status_code: 200
```

1.3 Let's view the **lamp.yml** manifest.

```
# cat -n lamp.yml
```

Output:

```
---
- name: "Deploying LAMP Stack"
  hosts: webservers
  become: yes
  tasks:
    - name: "Install Latest Packages"
      ansible.builtin.yum:
        name:
          - httpd
          - mariadb-server
          - php
          - php-mysqlnd
          - firewalld
        state: latest

    - name: "Copy contents"
      ansible.builtin.copy:
        content: "Test LAMP Stack"
        dest: /var/www/html/index.html
    - name: "Enable and Start HTTPD Service"
      ansible.builtin.service:
        name: httpd
        state: started
        enabled: yes

    - name: "Enable and Start MYSQL Service"
      ansible.builtin.service:
        name: mariadb
        state: started
        enabled: yes

    - name: "Enable and Start Firewalld Service"
      ansible.builtin.service:
        name: firewalld
        state: started
        enabled: yes
    - name: "Add exception in the firewall"
      ansible.posix.firewalld:
        service: http
        permanent: yes
        state: enabled
        immediate: yes

- name: Test page
  hosts: localhost
  tasks:
    - name: acces page
      uri:
        url: http://eoc-node3
        status_code: 200
```

1.4 Let's check the syntax of **lamp.yml**.

```
# ansible-playbook --syntax-check lamp.yml
```

Output:

```
[admin@eoc-controller ~]$ ansible-playbook --syntax-check lamp.yml

playbook: lamp.yml
```

1.5 Let's perform dry-run.

```
# ansible-playbook -C lamp.yml
```

Output:

```
[admin@eoc-controller ~]$ ansible-playbook -C lamp.yml

PLAY [Deploying LAMP Stack] ****************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [eoc-node3]

TASK [Install Latest Packages] *************************************************************************
changed: [eoc-node3]

TASK [Copy contents] ***********************************************************************************
changed: [eoc-node3]

TASK [Enable and Start HTTPD Service] ******************************************************************
fatal: [eoc-node3]: FAILED! => {"changed": false, "msg": "Could not find the requested service httpd: h
ost"}

PLAY RECAP *********************************************************************************************
eoc-node3                  : ok=3    changed=2    unreachable=0    failed=1    skipped=0    rescued=0
  ignored=0
```

Note: The playbook fails in the dryrun stage because dry run assumes or pretents that the package is installed and the service can be started, But in the backend no package is installed so only this check fails.

1.6 Let's run the **lamp.yml** playbook.

```
# ansible-playbook lamp.yml
```

Output:

```
[admin@eoc-controller ~]$ ansible-playbook lamp.yml

PLAY [Deploying LAMP Stack] ****************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [eoc-node3]

TASK [Install Latest Packages] *************************************************************************
changed: [eoc-node3]

TASK [Copy contents] ***********************************************************************************
changed: [eoc-node3]

TASK [Enable and Start HTTPD Service] ******************************************************************
changed: [eoc-node3]

TASK [Enable and Start MYSQL Service] ******************************************************************
changed: [eoc-node3]

TASK [Enable and Start Firewalld Service] **************************************************************
changed: [eoc-node3]

TASK [Add exception in the firewall] *******************************************************************
changed: [eoc-node3]

PLAY [Test page] ***************************************************************************************

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

TASK [acces page] **************************************************************************************
ok: [localhost]

PLAY RECAP *********************************************************************************************
eoc-node3                  : ok=7    changed=6    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
localhost                  : ok=2    changed=0    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
```

1.7 Let's verify the lamp deployment on the webservers using ad-hoc command.

```
# ansible webservers -m 'uri' -a "url=http://localhost
return_content=yes"
```

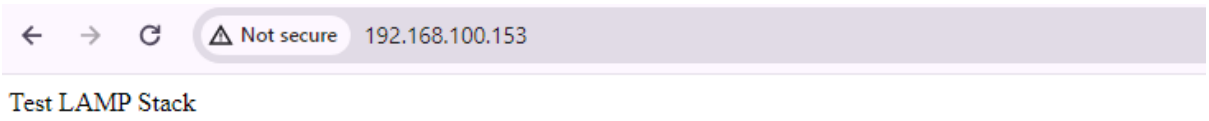
Output:

```
[admin@eoc-controller ~]$ ansible webservers -m 'uri' -a "url=http://localhost return_content=yes"
eoc-node3 | SUCCESS => {
    "accept_ranges": "bytes",
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "connection": "close",
    "content": "Test LAMP Stack",
    "content_length": "15",
    "content_type": "text/html; charset=UTF-8",
    "cookies": {},
    "cookies_string": "",
    "date": "Wed, 22 Nov 2023 17:27:30 GMT",
    "elapsed": 0,
    "etag": "\"f-60ac105d86894\"",
    "last_modified": "Wed, 22 Nov 2023 17:26:43 GMT",
    "msg": "OK (15 bytes)",
    "redirected": false,
    "server": "Apache/2.4.37 (CentOS Stream)",
    "status": 200,
    "url": "http://localhost"
}
```

1.8 Let's open the browser and verify the installation.

```
# http://192.168.100.153
```

Output:



Test LAMP Stack