## Lab 17: Automate Installation of Web Server Using Ansible

### Case Study

The auto parts manufacturer TENSO has restructured its network operations management, which had become unwieldy and overly individualized due to the absence of centralized network infrastructure management. A network development methodology was introduced, enabling the automation of operations through the use of scripts. The implementation of this methodology facilitated rapid troubleshooting, regardless of the skill level of the personnel involved, minimized network downtime, and automated updates to network equipment across the factories, tasks that previously demanded a significant workload.

### Business Challenge

The IT administrator at TENSO intends to install web servers in multiple racks within the data center. However, this is a time-consuming task for an individual network engineer, as it requires installing a web server on each machine manually. To provide a scalable and efficient solution, the company has hired you, a certified DevNet Associate, to automate this process.

### Solution

This task will be automated using Ansible. The first step involves configuring Ansible to establish communication with the web server application. Your subsequent step will involve the development of a template that will automate the installation of the Apache web server. Additionally, a customized playbook will be developed to install Apache with specific configuration instructions tailored to the organization’s requirements.

The Apache HTTP Server, widely recognized for its extensive use, is a leading open-source solution for web server software. It functions as a server-side application that receives requests from clients, such as web browsers, and delivers the requested web pages or content. In essence, it serves as the foundational software that enables users to access and interact with websites.

Follow the steps to complete the lab:

1. Configure Ansible in a DEVASC VM.
2. Verify communications with the local web server.
3. Create Ansible playbooks to automate web server installation.
4. Add Apache web server configuration options to the Ansible playbook.

|  |  |  |  |
| --- | --- | --- | --- |
| **// Configure Ansible in a DEVASC VM**  1. Execute the following command: **sudo systemctl start ssh** to start the SSH service in DEVASC VM. SSH service is pre-installed in the DEVASC VM for your reference you use the below-provided commands to install SSH service in Ubuntu Linux.   |  | | --- | | sudo apt-get install openssh-server  sudo apt-get install sshpass |     2. Double-click on a **Visual Studio Code** icon to open it.    3. Click on **File.** Then click on **Open Folder…**    4. Then navigate to the **/labs/devnet-src/ansible** directory and click **OK** to open it.    5. The two subdirectories for the Ansible labs are now loaded in the Visual Studio Code **EXPLORER** pane. In this lab, we will work on the **ansible-apache** directory.    6. Click on the **hosts** file in the **ansible-apache** directory to open it.  7. Copy and paste the below-provided lines into the **hosts** file and press **Ctrl+S** to save the file.   |  | | --- | | [webservers]  192.0.2.3 ansible\_ssh\_user=devasc ansible\_ssh\_pass=Cisco123! |     8. The credentials **devasc** and **Cisco123!** are admin credentials for the DEVASC VM. The IPv4 address you will use for this lab is **192.0.2.3**. This is a static IPv4 address on the VM under the dummy0 interface, as shown in the output for the **ip addr** command**.**    9. In the **ansible-apache** subdirectory, click on the **cfg** file to open it. Remove the comment. Copy and paste the below-provided lines to the file and then press **Ctrl+S** to save it.   |  | | --- | | [defaults]  # Use local hosts file in this folder  inventory=./hosts  # Don't worry about RSA Fingerprints  host\_key\_checking = False  # Do not create retry files  retry\_files\_enabled = False | |

|  |
| --- |
| **// Verify Communications with the Local Web Server**  1. First execute following command: **cd labs/devnet-src/ansible/ansible-apache/** to go inside ansible-apache directory**.** Thenexecute the following command: **ansible webservers -m ping** to verify communications with the devices listed within the webservers group of your hosts inventory file.    2. Execute the following command: **ansible webservers -m command -a "/bin/echo hello world"** to verify communications with the devices listed within the **webservers** group of your **hosts** inventory file. In this example, you send the argument –**a “/bin/echo hello world”** to ask the local webserver to respond with “hello world”. |

|  |  |  |
| --- | --- | --- |
| **// Create Ansible Playbooks to Automate Web Server Installation**  1. In the Visual Studio Code, create a new file in the **ansible-apache** directory with the following name: **test\_apache\_playbook.yaml**.    2. Copy the paste the below-provided script in the **test\_apache\_playbook.yaml** file and then press **Ctrl+S** to save it.   |  | | --- | | ---  - hosts: webservers  tasks:  - name: run echo command  command: /bin/echo hello world |     The Ansible playbook is a YAML file. Make sure you use the proper YAML indentation. Every space and dash is significant. You may lose some formatting if you copy and paste the code.  3. Execute the following command: **ansible-playbook -v test\_apache\_playbook.yaml** to run ansible playbook.    4. In the Visual Studio Code, create a new file in the **ansible-apache** directory with the following name: **install\_apache\_playbook.yaml**.    5. Copy the paste the below-provided script in the **install\_apache\_playbook.yaml** file and then press **Ctrl+S** to save it.   |  | | --- | | ---  - hosts: webservers  become: yes  tasks:  - name: INSTALL APACHE2  apt: name=apache2 update\_cache=yes state=latest    - name: ENABLED MOD\_REWRITE  apache2\_module: name=rewrite state=present  notify:  - RESTART APACHE2    handlers:  - name: RESTART APACHE2  service: name=apache2 state=restarted |     6. Execute the following command: **ansible-playbook -v install\_apache\_playbook.yaml** to run ansible playbook. The first time Apache is installed on your VM; the task **INSTALL APACHE2** will take anywhere from 30 seconds to a few minutes depending on your internet speed.    7. Execute the following command: **sudo systemctl status apache2** to verify that Apache is now installed. We have successfully installed the Apache web server. Press **q** to quit. |

|  |  |
| --- | --- |
| **// Add Apache Web Server Configuration Options to the Ansible Playbook**  1. In the Visual Studio Code, create a new file in the **ansible-apache** directory with the following name: **install\_apache\_options\_playbook.yaml**.    2. Copy the paste the below-provided script in the **install\_apache\_options\_playbook.yaml** file and then press **Ctrl+S** to save it.   |  | | --- | | ---  - hosts: webservers  become: yes  tasks:  - name: INSTALL APACHE2  apt: name=apache2 update\_cache=yes state=latest    - name: ENABLED MOD\_REWRITE  apache2\_module: name=rewrite state=present  notify:  - RESTART APACHE2    - name: APACHE2 LISTEN ON PORT 8081  lineinfile: dest=/etc/apache2/ports.conf regexp="^Listen 80" line="Listen 8081" state=present  notify:  - RESTART APACHE2    - name: APACHE2 VIRTUALHOST ON PORT 8081  lineinfile: dest=/etc/apache2/sites-available/000-default.conf regexp="^<VirtualHost \\*:80>" line="<VirtualHost \*:8081>" state=present  notify:  - RESTART APACHE2    handlers:  - name: RESTART APACHE2  service: name=apache2 state=restarted |   This playlist is very similar to the previous one with the addition of two tasks that have the webservers listen on port 8081 instead of port 80.  The **lineinfile** module is used to replace existing lines in the /etc/apache2/ports.conf and /etc/apache2/sites-available/000-default.conf files.    3. Execute the following commands: **cat /etc/apache2/ports.conf** and **cat /etc/apache2/sites-available/000-default.conf** respectively to verify that the web server is currently running on port 80.    4. Execute the following command: **ansible-playbook install\_apache\_options\_playbook.yaml** to run the playbook.    5. Again execute the following commands: **cat /etc/apache2/ports.conf** and **cat /etc/apache2/sites-available/000-default.conf** respectively. Notice that the playbook modified these files to listen on port 8081. |