

Implementing a CI/CD Pipeline Using Ansible for Continuous Deployment

Table of Contents

1. [Introduction](#)
 2. [Problem Statement](#)
 3. [Prerequisites](#)
 - [Software Requirements](#)
 4. [Step-by-Step Guide to Implementing CI/CD with Ansible](#)
 - [Step 1: Set Up Inventory and Group Variables](#)
 - [Step 2: Create Playbooks for CI/CD Tasks](#)
 - [Task 1: Install Dependencies](#)
 - [Task 2: Deploy Application](#)
 - [Task 3: Configure Web Server](#)
 - [Step 3: Run the CI/CD Pipeline](#)
 5. [Verifying the Deployment](#)
 6. [Supported References](#)
-

Introduction

In modern software development, Continuous Integration and Continuous Deployment (CI/CD) pipelines automate the process of deploying applications. This guide demonstrates how to implement a CI/CD pipeline using Ansible, leveraging its capabilities to automate tasks across multiple environments.

Problem Statement

Manually deploying applications can be error-prone and time-consuming. A well-defined CI/CD pipeline allows for consistent deployments, faster release cycles, and automated testing, ensuring that applications are always in a deployable state.

Prerequisites

Completion of all previous lab guides (up to Lab Guide-09) is required before proceeding with Lab Guide-10.

Software Requirements

- **Ansible 2.9+:** Installed on your control node (WSL for Windows users).
 - **WSL (Windows Subsystem for Linux):** If using Windows as your control node.
 - **Access to a Web Server:** This can be a local machine or a remote server.
 - **Git:** If using a Git repository for the application code.
-

Step-by-Step Guide to Implementing CI/CD with Ansible

Step 1: Set Up Inventory and Group Variables

1. Create Playbook Directory:

- Create a directory for your playbooks:

```
mkdir ~/ansible_cicd  
cd ~/ansible_cicd
```

2. Create Inventory File:

- Create a file named `inventory.ini`:

```
nano inventory.ini
```

```
[web_servers]  
<ip-address>  
  
[web_servers:vars]  
ansible_user=ansible_user  
ansible_password=P@ssw0rd  
ansible_connection=winrm  
ansible_winrm_transport=basic  
ansible_port=5985  
ansible_winrm_server_cert_validation=ignore
```

```
[web_servers]  
192.168.0.102  
  
[web_servers:vars]  
ansible_user=ansible_user  
ansible_password=P@ssw0rd  
ansible_connection=winrm  
ansible_winrm_transport=basic  
ansible_port=5985  
ansible_winrm_server_cert_validation=ignore
```

3. Create Group Variables:

- Create a directory for group variables and add a file named `group_vars/web_servers.yml`:

```
mkdir group_vars
```

```
nano group_vars/web_servers.yml
```

```
---  
app_name: "hello_world"  
app_directory: "/var/www/html/{{ app_name }}"
```

```
---  
app_name: "hello_world"  
app_directory: "/var/www/html/{{ app_name }}"
```

Step 2: Create Playbooks for CI/CD Tasks

1. Create `install_dependencies.yml` Playbook:

```
---  
- name: Install Dependencies  
  hosts: web_servers  
  tasks:  
    - name: Ensure Chocolatey is installed  
      win_chocolatey:  
        name: chocolatey  
        state: present  
  
    - name: Install OpenSSL  
      win_chocolatey:  
        name: openssl  
        state: present  
  
    - name: Install Visual C++ Redistributable  
      win_chocolatey:  
        name: vc_redist2015  
        state: present  
  
    - name: Install required packages  
      win_chocolatey:  
        name: git  
        state: present  
        ignore_dependencies: yes
```

```
---
- name: Install Dependencies
  hosts: web_servers
  tasks:
    - name: Ensure Chocolatey is installed
      win_chocolatey:
        name: chocolatey
        state: present

    - name: Install OpenSSL
      win_chocolatey:
        name: openssl
        state: present

    - name: Install Visual C++ Redistributable
      win_chocolatey:
        name: vc_redist2015
        state: present

    - name: Install required packages
      win_chocolatey:
        name: git
        state: present
        ignore_dependencies: yes
```

2. Create `deploy_application.yml` Playbook:

```
---
- name: Deploy Application
  hosts: web_servers
  tasks:
    - name: Create application directory
      win_file:
        path: "{{ app_directory }}"
        state: directory

    - name: Copy HTML file
      win_copy:
        content: "<html><body><h1>Hello World!</h1></body></html>"
        dest: "{{ app_directory }}/index.html"
```

3. Create `configure_web_server.yml` Playbook:

```
---
- name: Configure Web Server
  hosts: web_servers
  tasks:
    - name: Open port 80 for HTTP traffic
      win_firewall_rule:
```

```

name: "Allow HTTP"
enable: yes
localport: '80'
protocol: TCP
action: allow
direction: in

```

Step 3: Run the CI/CD Pipeline

1. Execute the Playbooks:

- Run the playbooks in order to set up the CI/CD pipeline:

```

ansible-playbook -i inventory.ini install_dependencies.yml
ansible-playbook -i inventory.ini deploy_application.yml
ansible-playbook -i inventory.ini configure_web_server.yml

```

```

user1@Swayaan:~/ansible_cicd$ ansible-playbook -i inventory.ini install_dependencies.yml

PLAY [Install Dependencies] *****

TASK [Gathering Facts] *****
ok: [192.168.0.102]

TASK [Ensure Chocolatey is installed] *****
ok: [192.168.0.102]

TASK [Install OpenSSL] *****
ok: [192.168.0.102]

TASK [Install Visual C++ Redistributable] *****
ok: [192.168.0.102]

TASK [Install required packages] *****
changed: [192.168.0.102]

PLAY RECAP *****
192.168.0.102      : ok=5   changed=1   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0

user1@Swayaan:~/ansible_cicd$

```

```

user1@Swayaan:~/ansible_cicd$ ansible-playbook -i inventory.ini deploy_application.yml

PLAY [Deploy Application] *****

TASK [Gathering Facts] *****
ok: [192.168.0.102]

TASK [Create application directory] *****
changed: [192.168.0.102]

TASK [Copy HTML file] *****
changed: [192.168.0.102]

PLAY RECAP *****
192.168.0.102      : ok=3   changed=2   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0

```

```
user1@Swayaan:~/ansible_cicd$ ansible-playbook -i inventory.ini configure_web_server.yml

PLAY [Configure Web Server] *****

TASK [Gathering Facts] *****
ok: [192.168.0.102]

TASK [Open port 80 for HTTP traffic] *****
ok: [192.168.0.102]

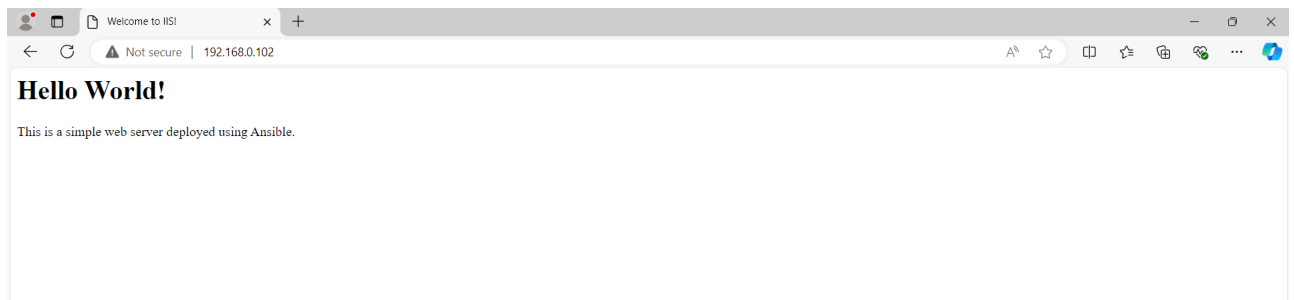
PLAY RECAP *****
192.168.0.102      : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Verifying the Deployment

1. Access the Application:

- Open a web browser and navigate to the server's IP address:

`http://<ip-address>`



You should see a "Hello World!" message if the deployment was successful.

Supported References

- [Ansible Documentation](#)
 - [Continuous Integration and Continuous Deployment](#)
 - [Ansible Galaxy](#)
-