## Automating Cisco Router Configurations with Ansible

### Overview

This documentation provides a step-by-step guide on how to automate Cisco router configurations using Ansible. Ansible is a powerful open-source automation tool that allows you to manage and automate configurations across your network infrastructure.

### Requirements

1. \*\*Ansible Installation\*\*: Ensure Ansible is installed on your control node where you will run your playbooks.

- \*\*Installation on Linux\*\*:

```bash

sudo apt update

sudo apt install ansible

```

- \*\*Installation on macOS\*\* (using Homebrew):

```bash

brew install ansible

```

- \*\*Installation on Windows\*\*:

- Ansible can be installed using WSL (Windows Subsystem for Linux) or in a Linux VM. Refer to Ansible documentation for detailed Windows installation instructions.

2. \*\*Cisco Router Configuration\*\*:

- Access to Cisco routers (either physical or virtual) with SSH connectivity enabled.

- SSH credentials (username and password) with administrative privileges on the routers.

3. \*\*Ansible Inventory File (`hosts`)\*\*:

- Create an Ansible inventory file (`hosts`) listing the IP addresses or hostnames of your Cisco routers.

- Example `hosts` file:

```

[cisco\_routers]

router1 ansible\_host=192.168.1.1

router2 ansible\_host=192.168.1.2

```

4. \*\*Ansible Playbook\*\*:

- Write an Ansible playbook (`cisco\_router.yaml`) that defines the tasks to configure Cisco router settings.

- Use Ansible modules such as `ios\_config`, `ios\_user`, etc., for managing configurations, users, and other network settings.

5. \*\*Variables File (`vars.yml`)\*\* (Optional but recommended):

- Store sensitive or reusable variables (like passwords) in a separate YAML file (`vars.yml`) and reference them in your playbook.

- Example `vars.yml`:

```yaml

router\_hostname: myrouter

router\_domain\_name: example.com

ssh\_version: 2

ssh\_timeout: 60

ssh\_retries: 5

admin\_password: "your\_admin\_password\_here"

```

### Writing the Ansible Playbook

Here's an example playbook (`cisco\_router.yaml`) to configure basic settings on Cisco routers:

```yaml

---

- name: Configure Cisco Router

hosts: cisco\_routers

gather\_facts: no # Disable gathering facts for network devices

tasks:

- name: Set hostname and domain name

ios\_config:

lines:

- hostname {{ router\_hostname }}

- ip domain-name {{ router\_domain\_name }}

- name: Configure SSH settings

ios\_config:

lines:

- ip ssh version {{ ssh\_version }}

- ip ssh time-out {{ ssh\_timeout }}

- ip ssh authentication-retries {{ ssh\_retries }}

- name: Create local user

ios\_user:

name: admin

privilege: 15

password: "{{ admin\_password }}"

state: present

# Additional tasks can be added for OSPF, VLANs, interfaces, etc.

```

### Running the Playbook

1. \*\*Execute the Playbook\*\*:

- Run the Ansible playbook using the `ansible-playbook` command:

```bash

ansible-playbook cisco\_router.yaml -i hosts --ask-pass --ask-become-pass

```

- Adjust command options (`--ask-pass`, `--ask-become-pass`) based on your authentication and privilege escalation requirements.

2. \*\*Verify Configuration\*\*:

- After running the playbook, verify configurations on Cisco routers using CLI commands (`show running-config`, etc.) or GUI tools.

### Tips for Success

- \*\*Testing\*\*: Test playbooks in a lab environment before deploying changes to production routers.

- \*\*Error Handling\*\*: Implement error handling and validation in playbooks to handle different scenarios gracefully.

- \*\*Documentation\*\*: Document playbooks, variables, and configurations for future reference and troubleshooting.

### Conclusion

By following this guide, you can effectively automate Cisco router configurations using Ansible, streamlining network management tasks and ensuring consistency across your infrastructure.

---

This documentation provides a comprehensive guide to get started with automating Cisco router configurations using Ansible. Adjust the playbook tasks and variables according to your specific network requirements and organizational policies. Happy automating!