Sample Ansible Playbook for configuring Interface of a Router

Case A: Simple Playbook

Ansible playbook is mainly divided into three section

- ✓ Hosts / target
- ✓ Vars
- ✓ Tasks

Let's create simple playbook for configuring interface of router

Hosts/Target: Here we include the information of hosts for which play will run. For Example:

```
- hosts: app
gather_facts: no
```

Vars : Includes variables which will be used by tasks while running playbook. It can be in the form of line, dictionaries ,items

Example 1:

```
username:                                                    cnlabs
cli:
   pass: 12345
routers_models:
 - { cisco: 2800, dell: 70, Intel: grey }
```

Example 2: As shown below example variable named network and iosv(directory) are defined under vars field

```
vars:
    network: 255.0.0.0

    iosv:
        host: "{{ inventory_hostname }}"
        username: cisco
        password: cisco
        transport: cli
```

Note: Defining variable is very useful when we have to use some values multiple times in playbook. In the case of IOS modules like ios_command and ios_config every task needs provider information like username , password , host etc. in this case we can simply call defined variable in var field.(shown next example)

Tasks: It consists of set of tasks to be performed on remote node that adds, remove and modify remote host configurations

Ex: in following

```
tasks:
 - name: configuring Interface GigabitEthernet 0/1
   ios_config:
     lines:
       - ip address 1.2.3.4  255.255.255.0
       - no shutdown
     parents: interface GigabitEthernet 0/1
     match: exact
     provider: "{{ iosv }}"
```

Note:

As shown above, tasks we are configuring IP address of GigabitEthernet 0/1 port of remote node which will comes under configuration of device so we used ios_config.

And ip address comes under Interface GigabitEthernet 0/1 , Parents field is used for specifying parent directory of ip address

Under provider “{{ iosv }}” calling all the variables defined under iosv in vars field

Complete Playbook will look like (routerinterface.yml):

Example:

```
---
- hosts: app
  become: yes
  gather_facts: no

   vars:
        iosv:
              host: "{{ inventory_hostname }}"
              username: cisco
              password: cisco
              transport: cli
  tasks:
    - name: configuring Interface GigabitEthernet 0/1
      ios_config:
        lines:
          - ip address 1.2.3.4  255.255.255.0
          - no shutdown
        parents: interface GigabitEthernet 0/1
        match: exact
```

```
    provider: "{{ iosv }}"
```

==========

Case B: Playbook using Ansible Roles

- Roles are a further level of abstraction that can be useful for organizing playbooks.
- As you add more and more functionality and flexibility to your playbooks, they can become unwieldy and difficult to maintain as a single file
- Roles allow you to create very minimal playbooks that then look to a directory structure to determine the actual configuration steps they need to perform.
- Organizing things into roles also allows you to reuse common configuration steps between different types of servers or devices.

Example project structure:

```
site.yml
webservers.yml
fooservers.yml
roles/
  webservers/
    files/
    templates/
    tasks/
```

handlers/
vars/
defaults/
meta/

This is what they are all for:

- **files**: This directory contains regular files that need to be transferred to the hosts you are configuring for this role. This may also include script files to run.
- **handlers**: All handlers that were in your playbook previously can now be added into this directory.
- **meta**: This directory can contain files that establish role dependencies. You can list roles that must be applied before the current role can work correctly.
- **templates**: You can place all files that use variables to substitute information during creation in this directory.
- **tasks**: This directory contains all of the tasks that would normally be in a playbook. These can reference files and templates contained in their respective directories without using a path.
- **vars**: Variables for the roles can be specified in this directory and used in your configuration files.

Within all of the directories but the "files" and "templates", if a file called main.yml exists, its contents will be automatically added to the playbook that calls the role.

Example: Lets create a playbook using roles that will add configuration to interface of router as we done earlier example.

- ✓ Considering Ansible/ is directory from where we will run playbook. Create directory named roles (if it does not exist)

Ubuntu@ubuntu/ansible# sudo mkdir roles
Ubuntu@ubuntu/ansible# cd roles

- ✓ Within this directory, we will define our roles. We will basically create a directory for each role that we will create. Since we are going to replicate our Routerinterface playbook, let's create an Routerinterface role:

Ubuntu@ubuntu/ansible/roles# mkdir Routerinterface
Ubuntu@ubuntu/ansible/roles# cd Routerinterface

- ✓ Within this directory, we create another set of directories that will help us separate the different sections of a normal playbook. For this example we need vars and tasks directory Create these directories now:

Ubuntu@ubuntu/ansible/roles/Routerinterface# mkdir vars
Ubuntu@ubuntu/ansible/roles/Routerinterface# mkdir tasks

Overall structure will look like:

Ansible/
    Ansible.cfg
    Hosts
    interfacerole.yml
    roles/
        Routerinterface /

```
tasks/
        main.yml
vars/
        main.yml
```

As mentioned earlier, playbook is divided in three section si.e. target, vars and tasks. Let's break routerinterface.yml playbook in three parts

So,

main.yml inside Ansible/Routerinterface/vars will look like

```
    iosv:
        host: "{{ inventory_hostname }}"
        username: cisco
        password: cisco
        transport: cli
```

note: alignment should be correct otherwise it will throw error

main.yml inside Ansible/Routerinterface/tasks will look like

```
- name: configuring Interface GigabitEthernet 0/1
```

```
    ios_config:
      lines:
        - ip address 1.2.3.4  255.255.255.0
        - no shutdown
      parents: interface GigabitEthernet 0/1
      match: exact
      provider: "{{ iosv }}"
```

Note: Here “{{ iosv }}” is calling the iosv variable which is defined in Ansible/Routerinterface/vars/main.yml

Finally,

Ansible/interfacerole.yml will look like:

```
---
- hosts: iosv1
  gather_facts: no
  connection: local
  roles:
  - Routerinterface
```

Running interfacerole.yml will it will include roles define under Routerinterface and execute for host in this case its iosv1

Further Scope:

Since our original playbook was very simple, so we modified according roles.

- ✓ Roles will be very useful while deploying different configurations for multiple hosts

  . roles/

  /dhcpserver

  /dhcpclient

- ✓ Example: For multiple roles such as dhcpserver, dhcpclient. We can write single playbook which will deploy configuration different hosts

```
---
- hosts: iosv1
  gather_facts: no
  connection: local
  roles:
  - dhcpserver
- hosts: iosv2
```

```
    gather_facts: no
    connection: local
    roles:
    - dhcpclient
```

It will deploy configurations two different host using single playbook.