Case Study for DevOps

Network Automation and Programmability

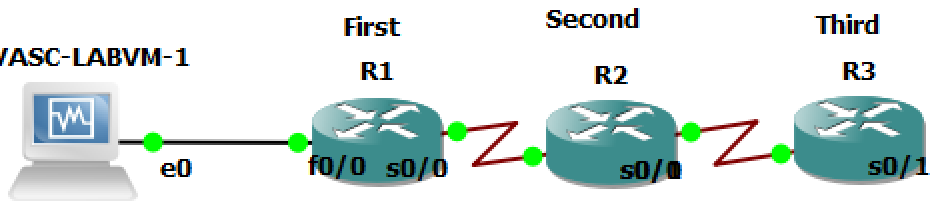
Intended Learning Outcomes:

- Configure and build a network using RIP routing protocol.
- Configure and implement OSPF routing protocol using Ansible in Cisco IOS device
- Configure OSPF routing protocol with NETCONF using Python ncclient in Cisco IOS-XE device.

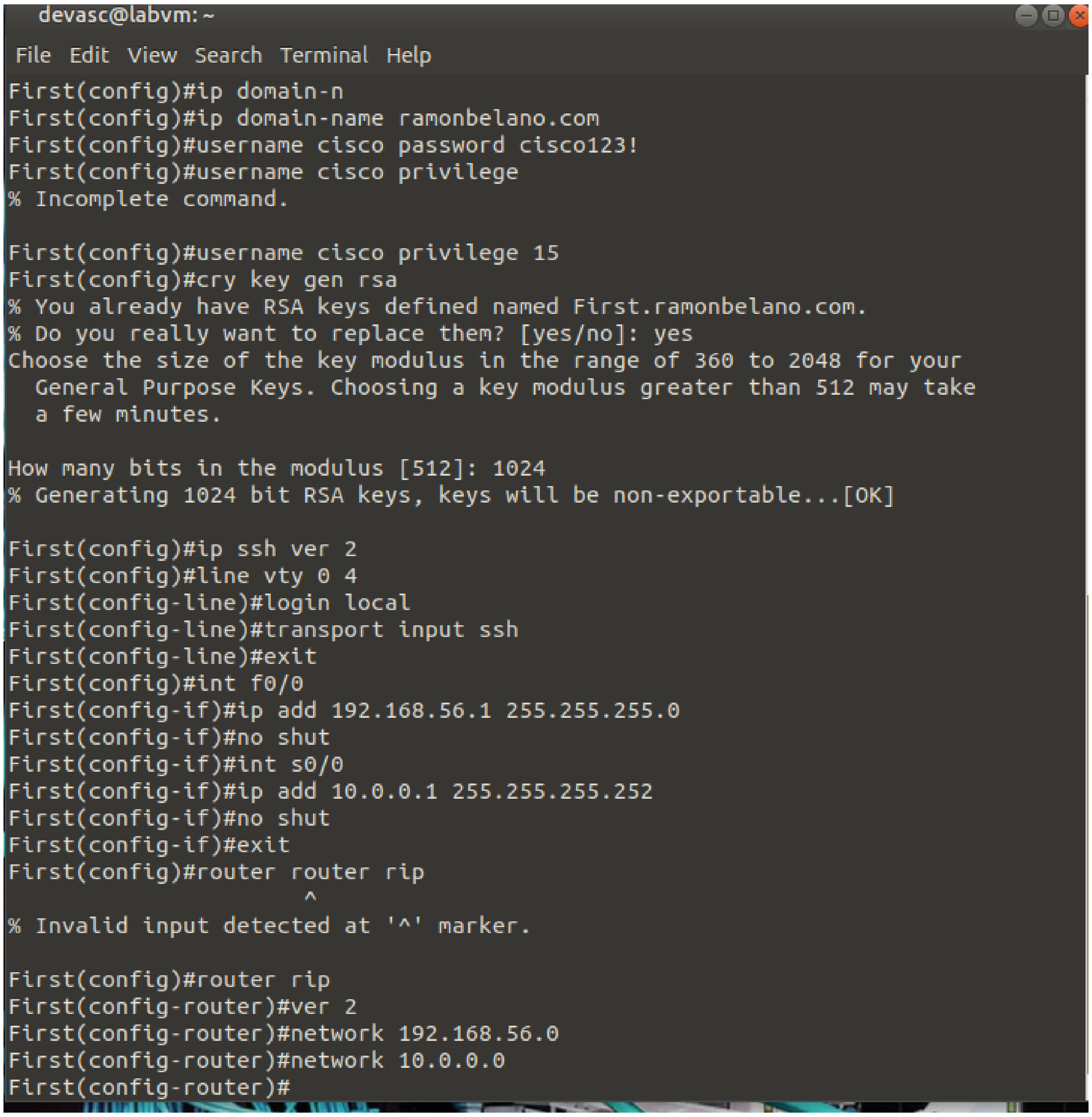
Resources:

- GNS3
- Virtual Box
- DEVASC-LABVM virtual machine
- CISCO 3750 IOS Image
-

Part 1: Network Automation using Ansible in CISCO IOS



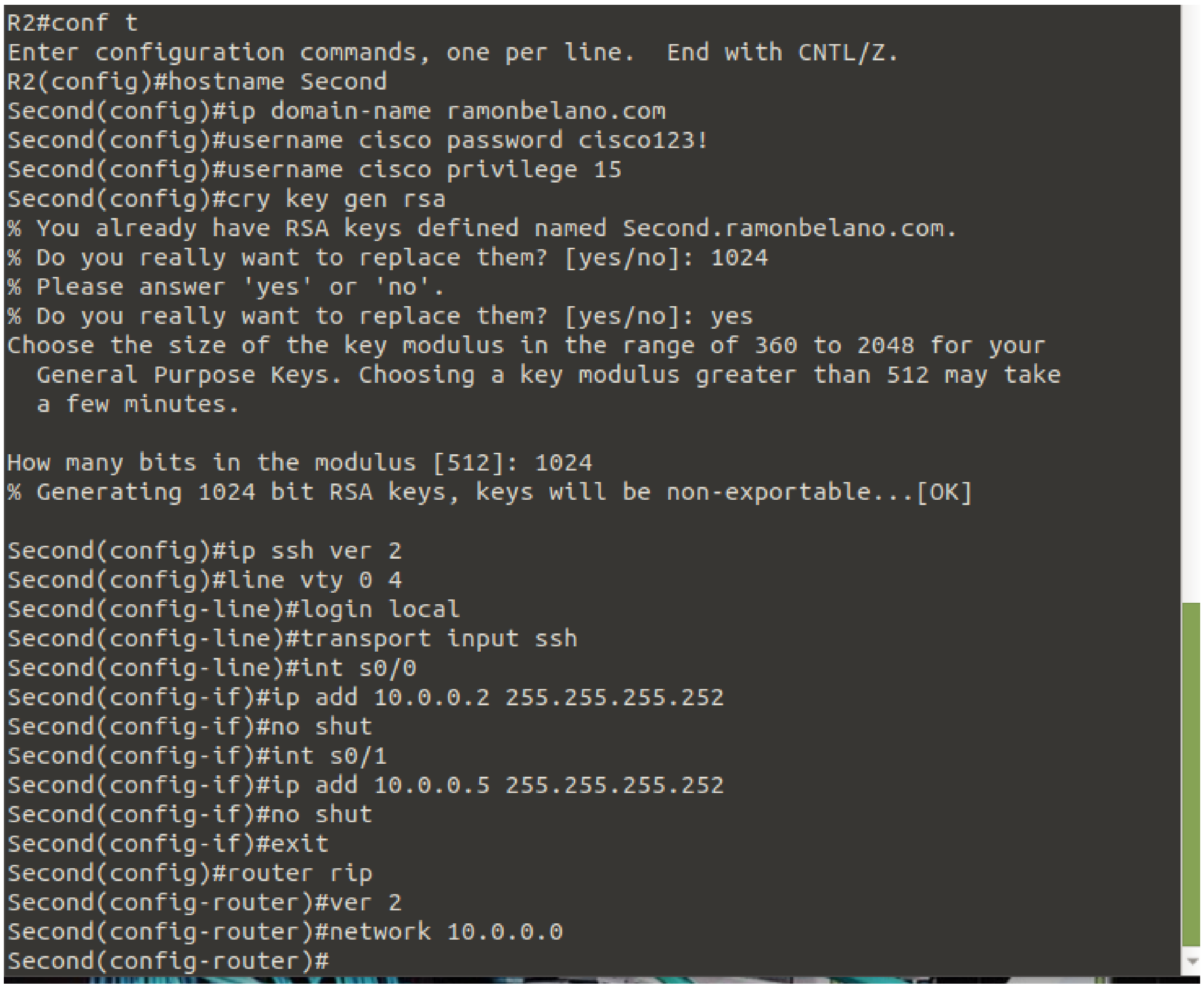
This image shows the topology built in gns3 and c3725



```
First(config)#ip domain-n
First(config)#ip domain-name ramonbelano.com
First(config)#username cisco password cisco123!
First(config)#username cisco privilege
% Incomplete command.

First(config)#username cisco privilege 15
First(config)#cry key gen rsa
% You already have RSA keys defined named First.ramonbelano.com.
% Do you really want to replace them? [yes/no]: yes
Choose the size of the key modulus in the range of 360 to 2048 for your
  General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.

How many bits in the modulus [512]: 1024
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]

First(config)#ip ssh ver 2
First(config)#line vty 0 4
First(config-line)#login local
First(config-line)#transport input ssh
First(config-line)#exit
First(config)#int f0/0
First(config-if)#ip add 192.168.56.1 255.255.255.0
First(config-if)#no shut
First(config-if)#int s0/0
First(config-if)#ip add 10.0.0.1 255.255.255.252
First(config-if)#no shut
First(config-if)#exit
First(config)#router router rip
                      ^
% Invalid input detected at '^' marker.

First(config)#router rip
First(config-router)#ver 2
First(config-router)#network 192.168.56.0
First(config-router)#network 10.0.0.0
First(config-router)#
```

Figure 1

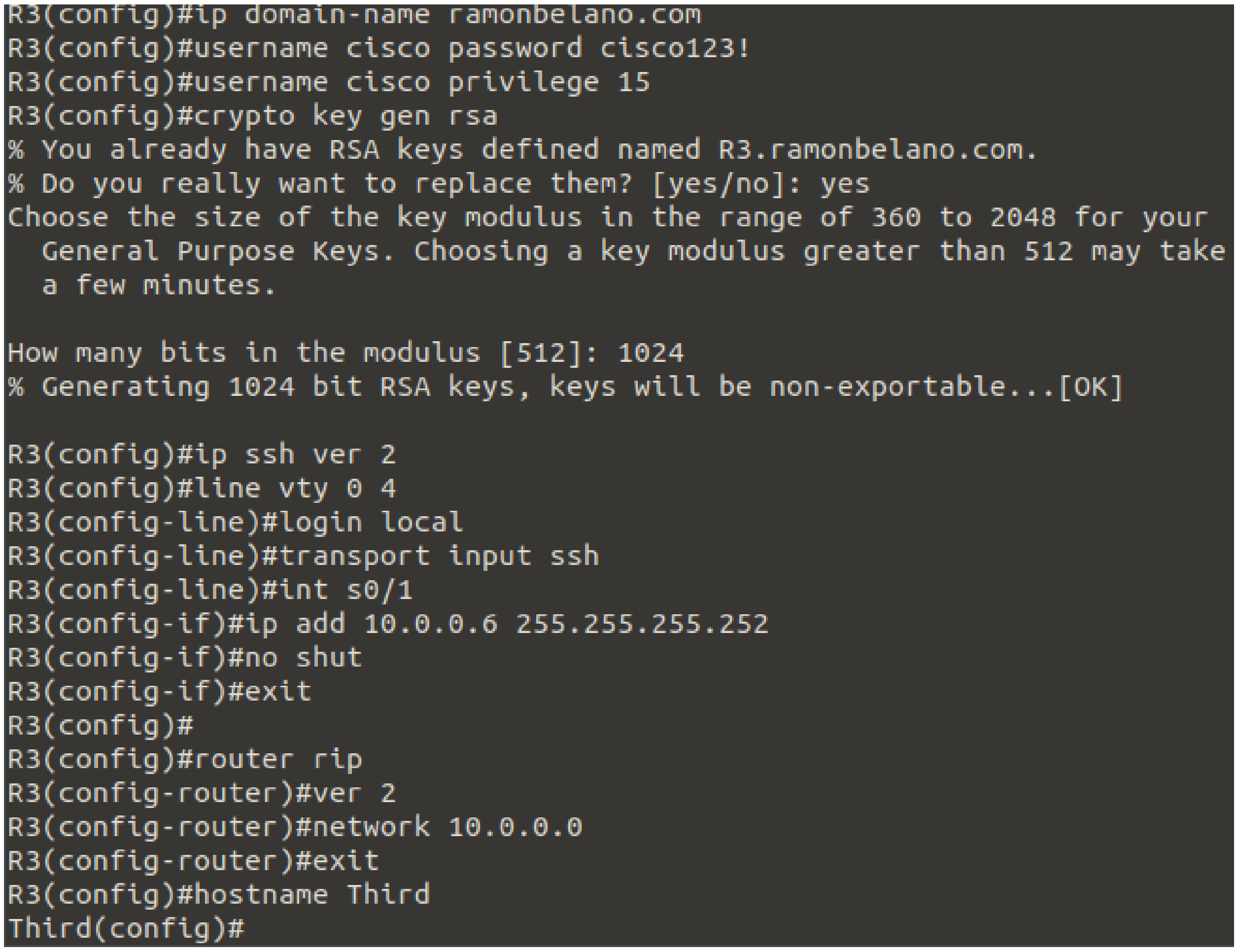
This image shows the configuration of the R1 router. First IP domain-name, username, password, privilege, crypto key gen RSA, IP ssh ver, line vty, login local, and transport input ssh. Second, add IP addresses in inter f0/0 and s0/0 and put no shutdown to work your interface. Third, add router rip ver 2 to connect another router.



```
R2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)#hostname Second
Second(config)#ip domain-name ramonbelano.com
Second(config)#username cisco password cisco123!
Second(config)#username cisco privilege 15
Second(config)#cry key gen rsa
% You already have RSA keys defined named Second.ramonbelano.com.
% Do you really want to replace them? [yes/no]: 1024
% Please answer 'yes' or 'no'.
% Do you really want to replace them? [yes/no]: yes
Choose the size of the key modulus in the range of 360 to 2048 for your
  General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.

How many bits in the modulus [512]: 1024
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]

Second(config)#ip ssh ver 2
Second(config)#line vty 0 4
Second(config-line)#login local
Second(config-line)#transport input ssh
Second(config-line)#int s0/0
Second(config-if)#ip add 10.0.0.2 255.255.255.252
Second(config-if)#no shut
Second(config-if)#int s0/1
Second(config-if)#ip add 10.0.0.5 255.255.255.252
Second(config-if)#no shut
Second(config-if)#exit
Second(config)#router rip
Second(config-router)#ver 2
Second(config-router)#network 10.0.0.0
Second(config-router)#
```

Figure 2

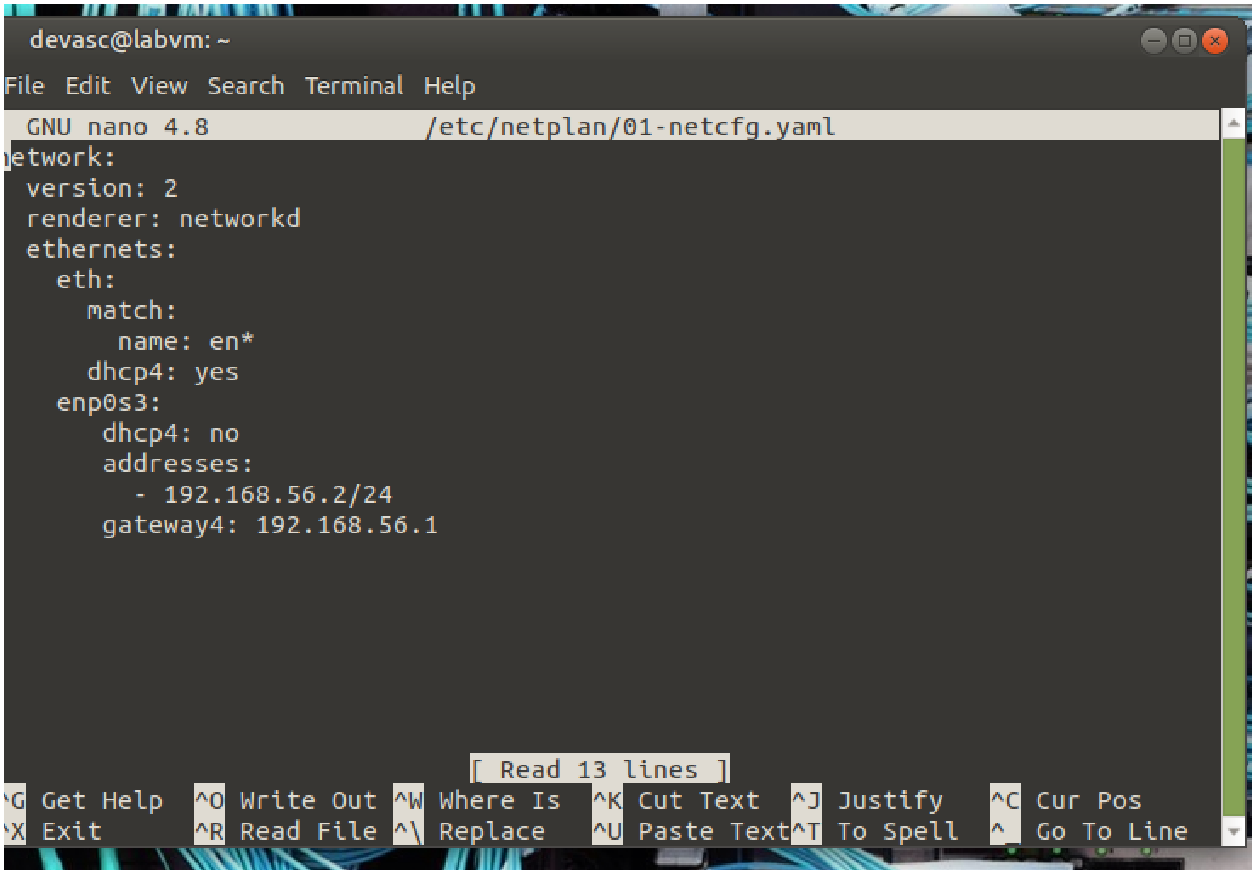
This image shows the configuration of the R2 router. First IP domain-name, username, password, privilege, crypto key gen RSA, IP ssh ver, line vty, login local, and transport input ssh. Second, also add IP address on the interface of s0/0 and s0/1 and no shutdown command to work your interface—third set router rip ver 2.



```
R3(config)#ip domain-name ramonbelano.com
R3(config)#username cisco password cisco123!
R3(config)#username cisco privilege 15
R3(config)#crypto key gen rsa
% You already have RSA keys defined named R3.ramonbelano.com.
% Do you really want to replace them? [yes/no]: yes
Choose the size of the key modulus in the range of 360 to 2048 for your
  General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.

How many bits in the modulus [512]: 1024
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]

R3(config)#ip ssh ver 2
R3(config)#line vty 0 4
R3(config-line)#login local
R3(config-line)#transport input ssh
R3(config-line)#int s0/1
R3(config-if)#ip add 10.0.0.6 255.255.255.252
R3(config-if)#no shut
R3(config-if)#exit
R3(config)#
R3(config)#router rip
R3(config-router)#ver 2
R3(config-router)#network 10.0.0.0
R3(config-router)#exit
R3(config)#hostname Third
Third(config)#
```

Figure 3

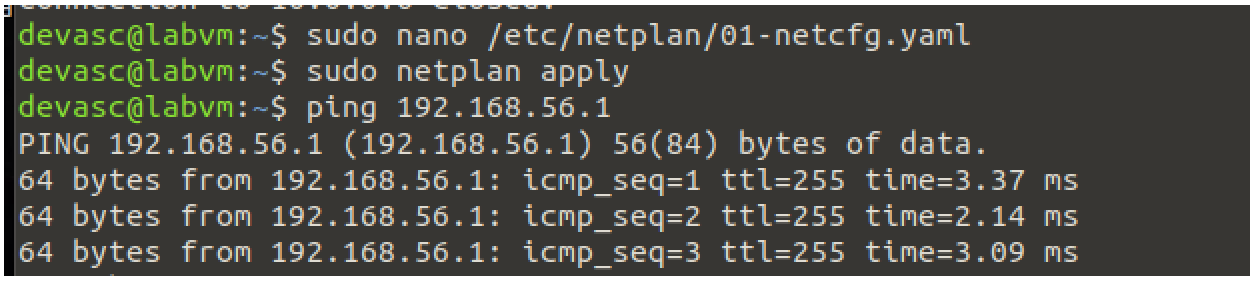
This image shows the configuration of the R3 router. First, Add IP domain-name, username, password, privilege, crypto key gen RSA, IP ssh ver, line vty, login local, and transport input ssh. Second, add an IP address on the interface of as0/1 and no shutdown command to work your interface. Third, add router rip ver 2.



```
devasc@labvm: ~
File  Edit  View  Search  Terminal  Help
 GNU nano 4.8                /etc/netplan/01-netcfg.yaml
network:
 version: 2
 renderer: networkd
 ethernets:
   eth:
     match:
       name: en*
     dhcp4: yes
   enp0s3:
      dhcp4: no
      addresses:
        - 192.168.56.2/24
      gateway4: 192.168.56.1

                              [ Read 13 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

Figure 4

In this figure, the user sets the netplan to add enpos3, add addresses, and gateway.

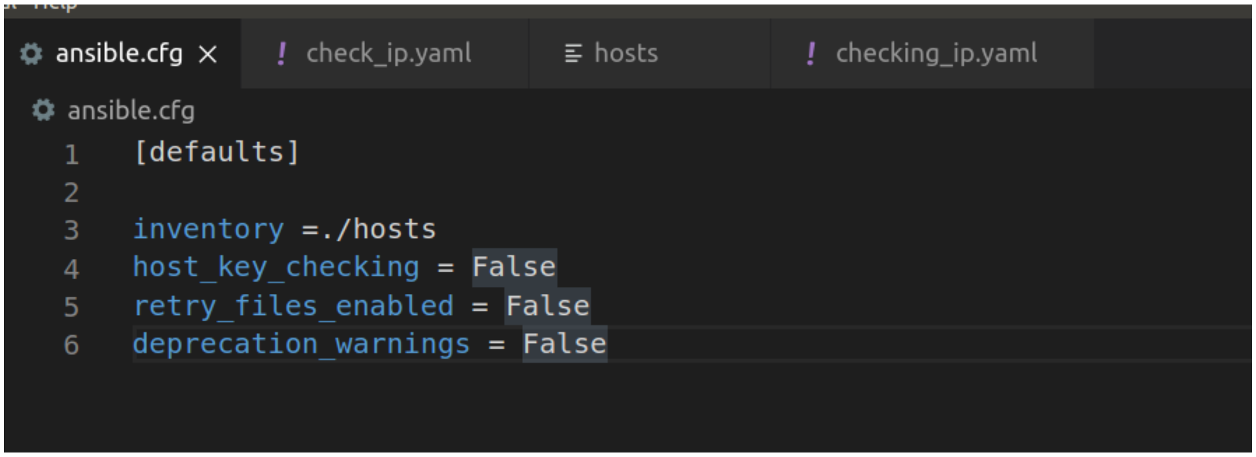


```
devasc@labvm:~$ sudo nano /etc/netplan/01-netcfg.yaml
devasc@labvm:~$ sudo netplan apply
devasc@labvm:~$ ping 192.168.56.1
PING 192.168.56.1 (192.168.56.1) 56(84) bytes of data.
64 bytes from 192.168.56.1: icmp_seq=1 ttl=255 time=3.37 ms
64 bytes from 192.168.56.1: icmp_seq=2 ttl=255 time=2.14 ms
64 bytes from 192.168.56.1: icmp_seq=3 ttl=255 time=3.09 ms
```

Figure 5

In this figure, the user command to open netplan and apply it. Then when the user does the configuration, you can ping the gateway if it's running. If not, the user must have an error.

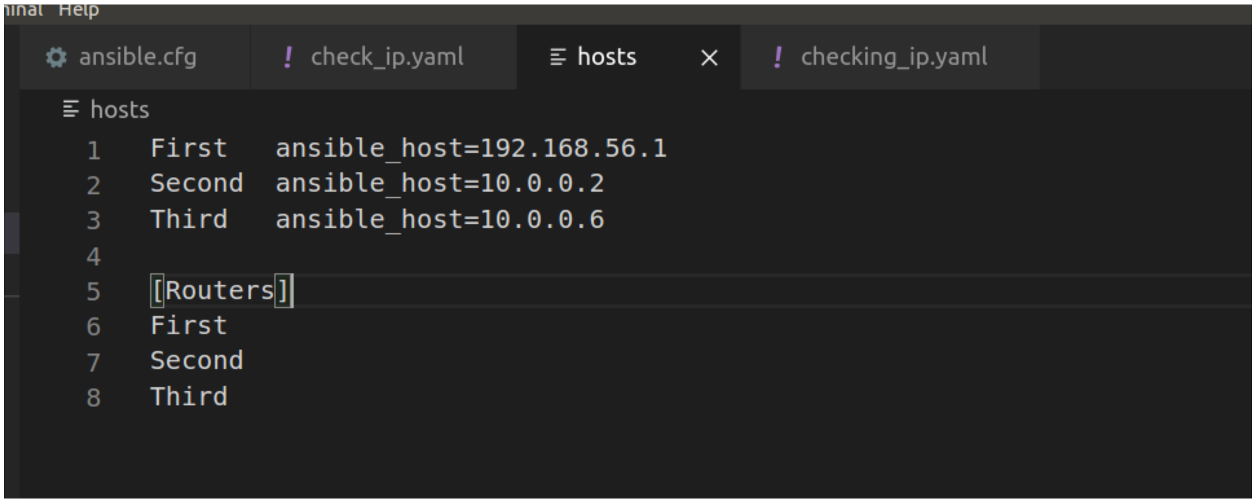
.



```
[defaults]

inventory =./hosts
host_key_checking = False
retry_files_enabled = False
deprecation_warnings = False
```

Figure 6

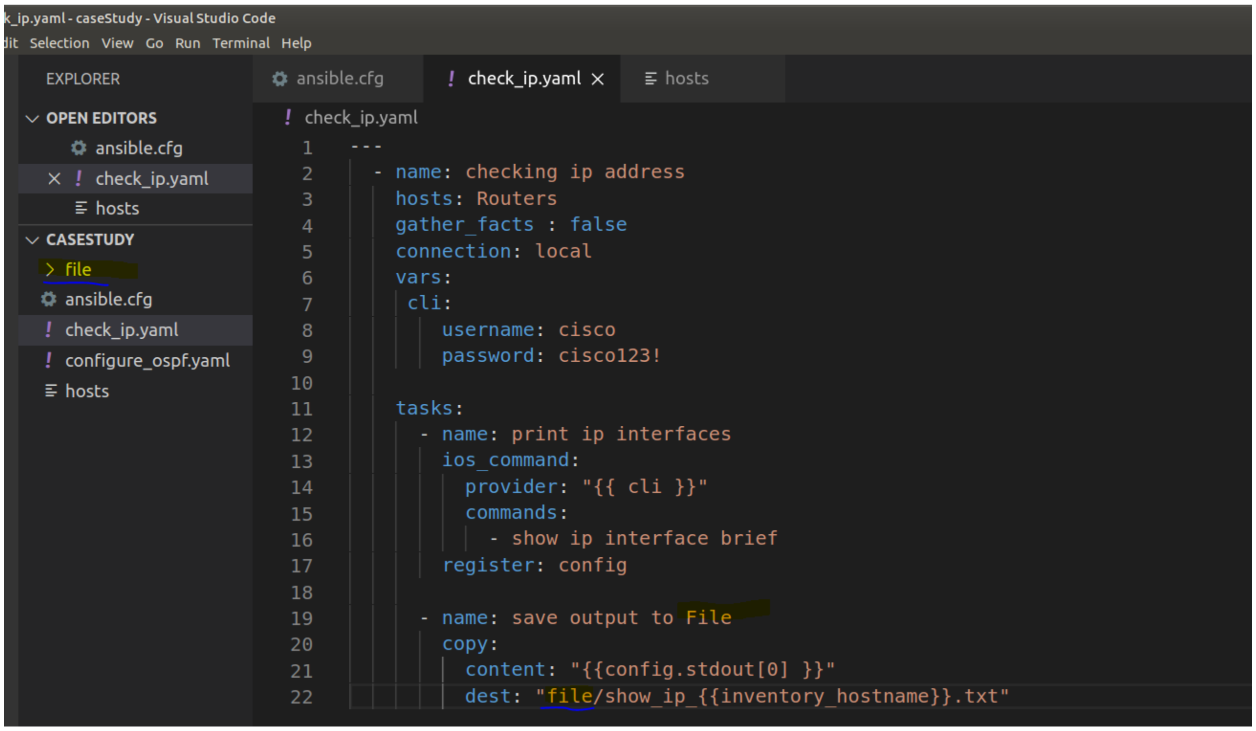
This figure shows the configuration of the ansible.cfg. First, set the defaults. Next is add and inventory, host key checking, retry file enabled, and deprecation.



```
First   ansible_host=192.168.56.1
Second  ansible_host=10.0.0.2
Third   ansible_host=10.0.0.6

[Routers]
First
Second
Third
```

Figure 7

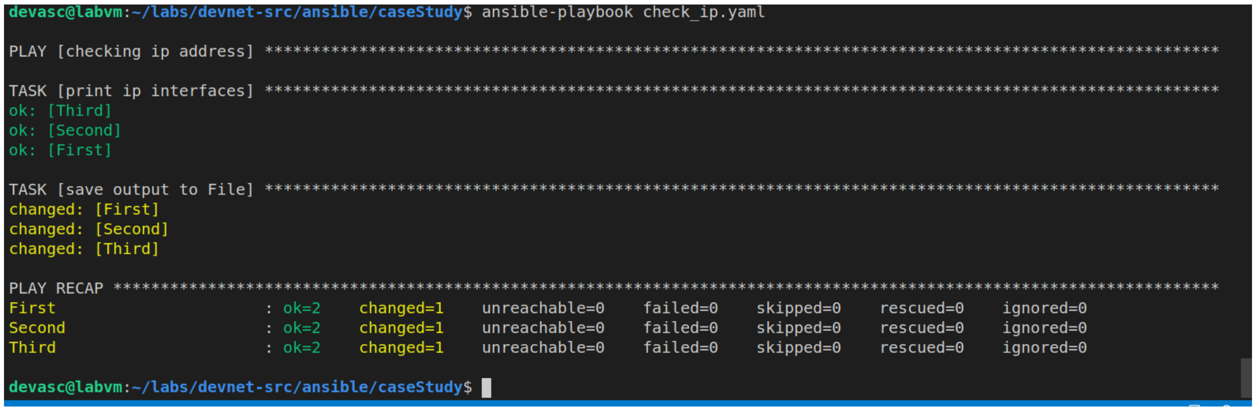
The image shows the configuration of hosts. Add first, second, third names based on your router, the add asible_hosts, and IP addresses. Next was to add routers, namely first, second, third.



```yaml
---
- name: checking ip address
  hosts: Routers
  gather_facts : false
  connection: local
  vars:
   cli:
     username: cisco
     password: cisco123!

  tasks:
    - name: print ip interfaces
      ios_command:
        provider: "{{ cli }}"
        commands:
          - show ip interface brief
      register: config

    - name: save output to File
      copy:
        content: "{{config.stdout[0] }}"
        dest: "file/show_ip_{{inventory_hostname}}.txt"
```

Figure 8

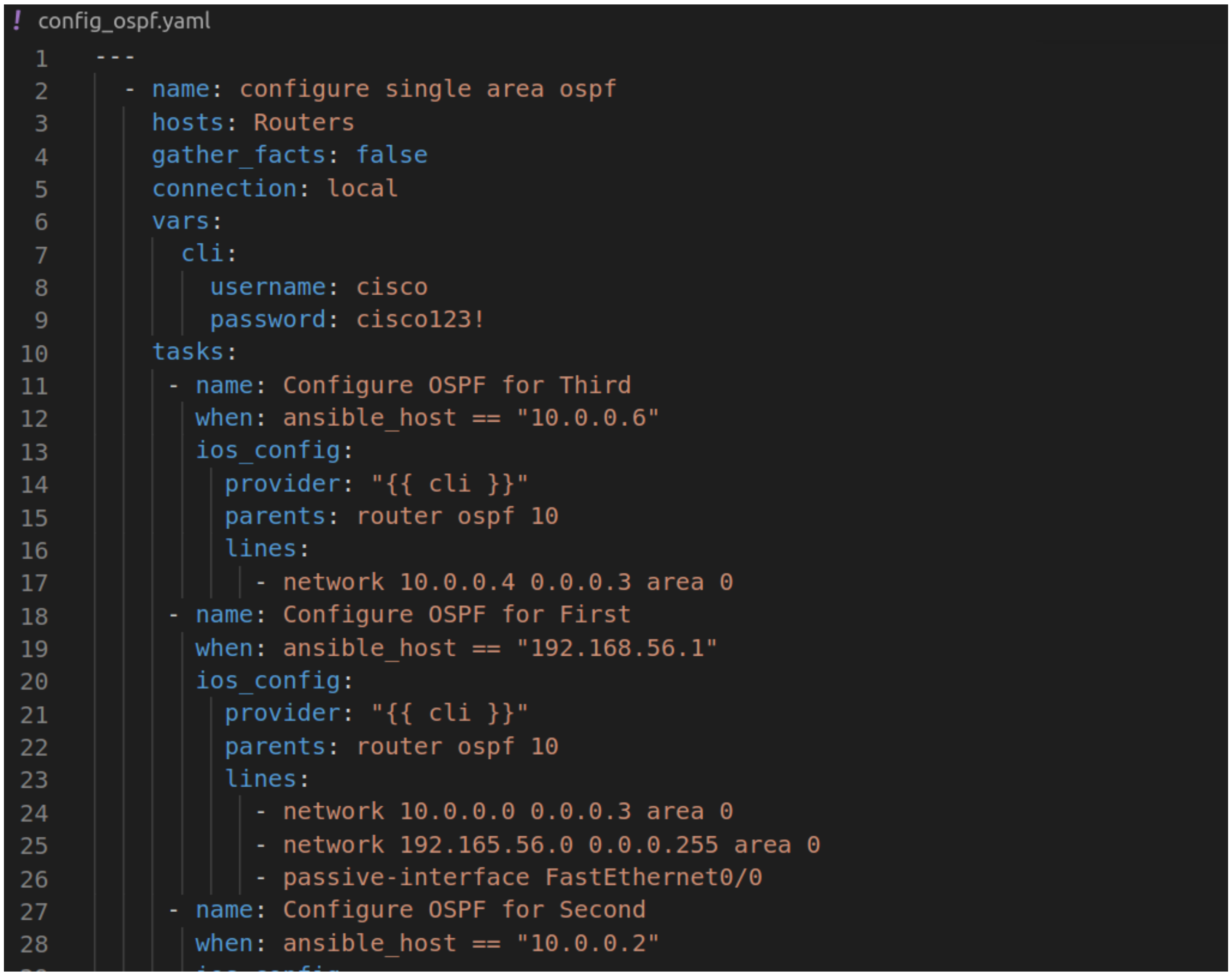
This image shows the configuration of the checking IP. To work the same output, create the directory with a name file.



```
devasc@labvm:~/labs/devnet-src/ansible/caseStudy$ ansible-playbook check_ip.yaml

PLAY [checking ip address] ****************************************************************************************

TASK [print ip interfaces] ****************************************************************************************
ok: [Third]
ok: [Second]
ok: [First]

TASK [save output to File] ****************************************************************************************
changed: [First]
changed: [Second]
changed: [Third]

PLAY RECAP ********************************************************************************************************
First                      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
Second                     : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
Third                      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

devasc@labvm:~/labs/devnet-src/ansible/caseStudy$
```

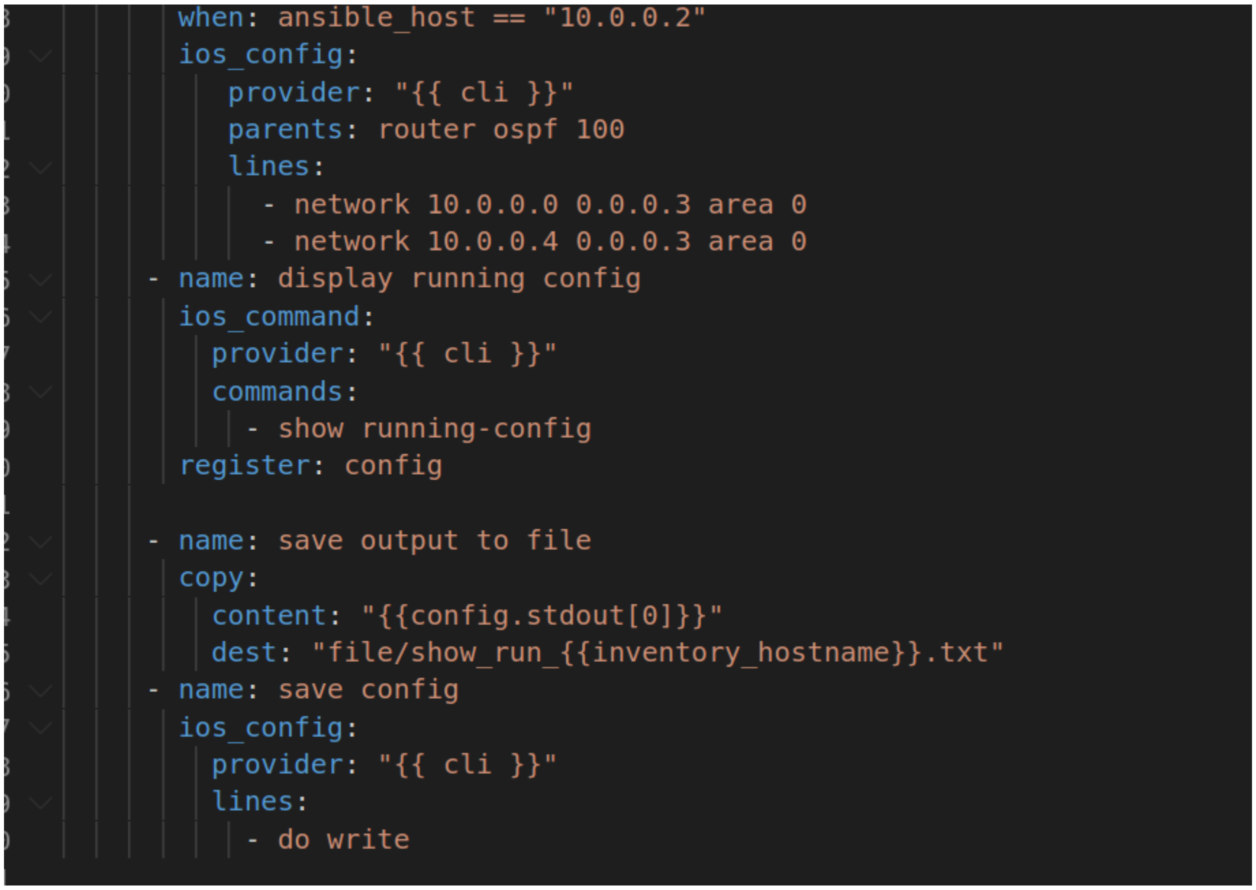
Figure 9

This image shows the result of the configuration of the check_ip.yaml. The command is ansible-playbook the name of the file.



```
config_ospf.yaml
---
  - name: configure single area ospf
    hosts: Routers
    gather_facts: false
    connection: local
    vars:
      cli:
        username: cisco
        password: cisco123!
    tasks:
      - name: Configure OSPF for Third
        when: ansible_host == "10.0.0.6"
        ios_config:
          provider: "{{ cli }}"
          parents: router ospf 10
          lines:
            - network 10.0.0.4 0.0.0.3 area 0
      - name: Configure OSPF for First
        when: ansible_host == "192.168.56.1"
        ios_config:
          provider: "{{ cli }}"
          parents: router ospf 10
          lines:
            - network 10.0.0.0 0.0.0.3 area 0
            - network 192.165.56.0 0.0.0.255 area 0
            - passive-interface FastEthernet0/0
      - name: Configure OSPF for Second
        when: ansible_host == "10.0.0.2"
```

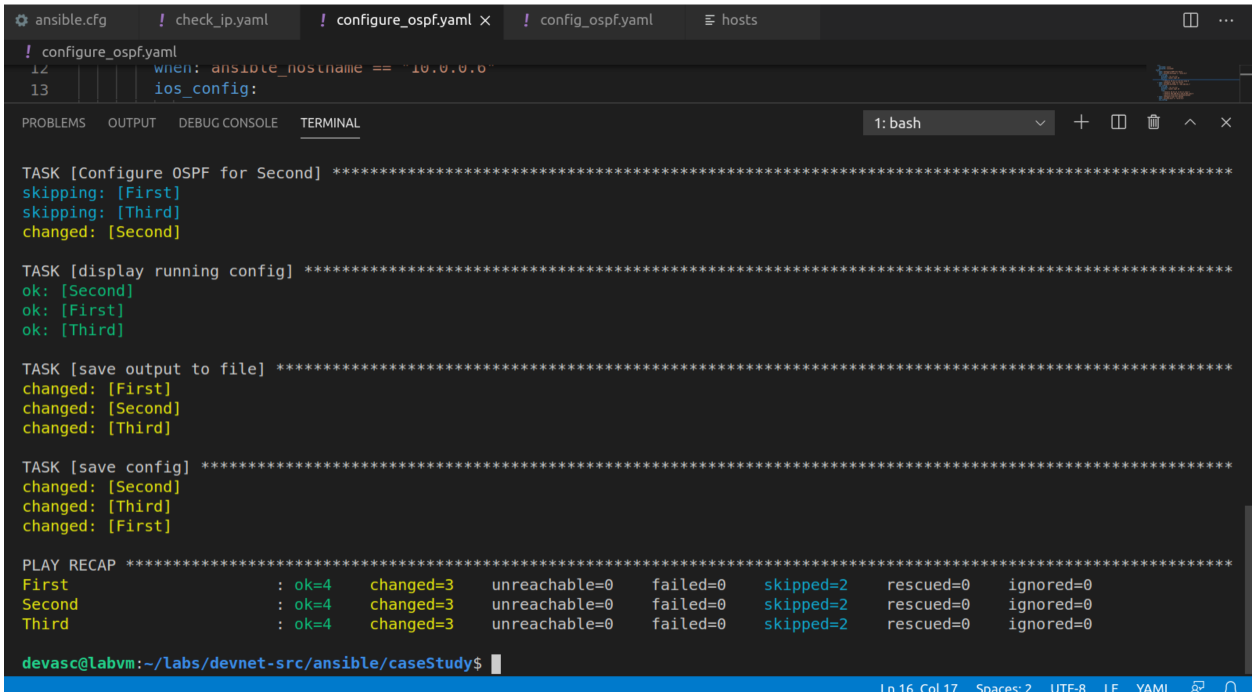
Figure 10.1



```
        when: ansible_host == "10.0.0.2"
        ios_config:
          provider: "{{ cli }}"
          parents: router ospf 100
          lines:
            - network 10.0.0.0 0.0.0.3 area 0
            - network 10.0.0.4 0.0.0.3 area 0
      - name: display running config
        ios_command:
          provider: "{{ cli }}"
          commands:
            - show running-config
        register: config

      - name: save output to file
        copy:
          content: "{{config.stdout[0]}}"
          dest: "file/show_run_{{inventory_hostname}}.txt"
      - name: save config
        ios_config:
          provider: "{{ cli }}"
          lines:
            - do write
```

Figure 10.2

This figure 10.1 and 10.2 show the configuration of the config_ospf.YAML. First configure OSPF to first, second, third. The next step is to display running-config. After this step, the next is to save the output to a file. The last step is to save the configuration.

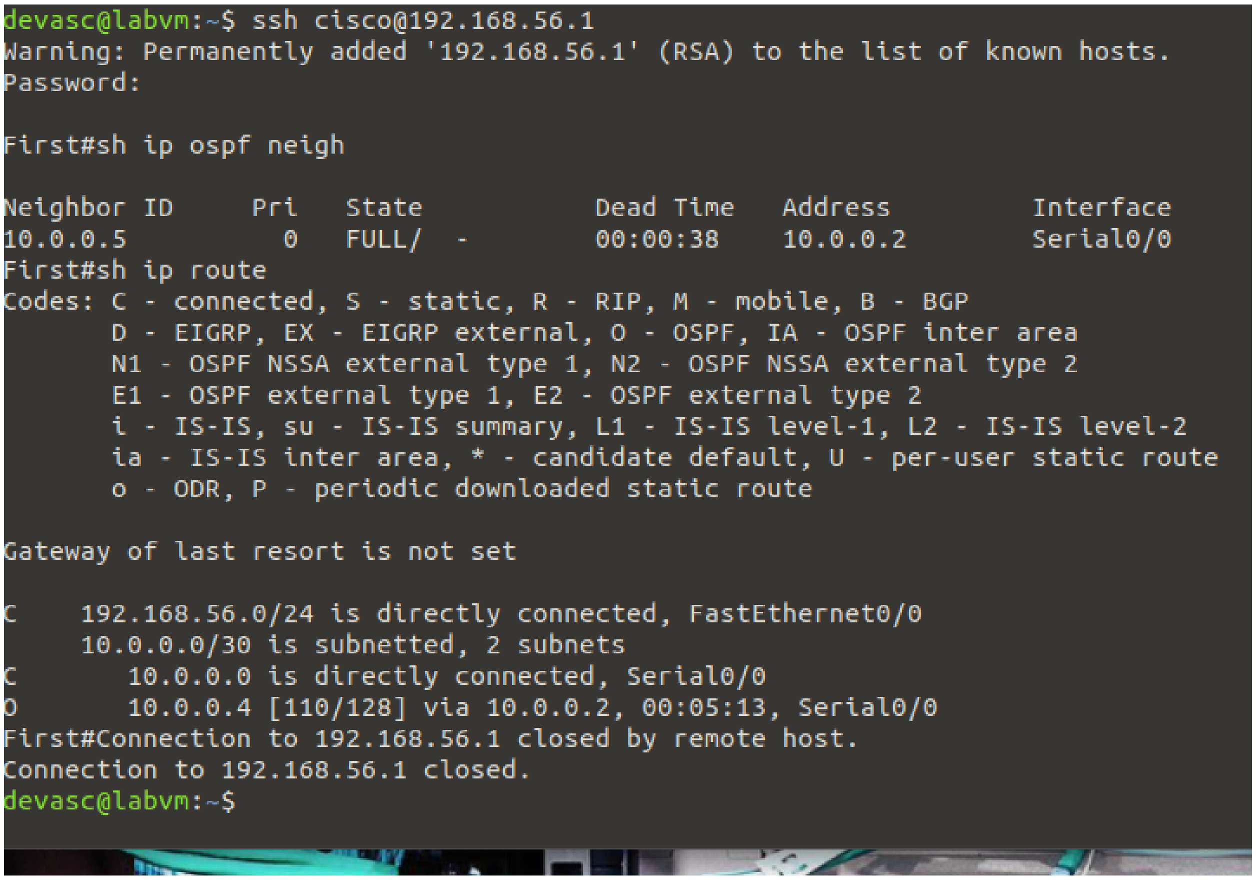


```
      when: ansible_hostname == "10.0.0.6"
      ios_config:
```

```
TASK [Configure OSPF for Second] ***********************************************************************************************
skipping: [First]
skipping: [Third]
changed: [Second]

TASK [display running config] **************************************************************************************************
ok: [Second]
ok: [First]
ok: [Third]

TASK [save output to file] *****************************************************************************************************
changed: [First]
changed: [Second]
changed: [Third]

TASK [save config] *************************************************************************************************************
changed: [Second]
changed: [Third]
changed: [First]

PLAY RECAP *********************************************************************************************************************
First                      : ok=4    changed=3    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
Second                     : ok=4    changed=3    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
Third                      : ok=4    changed=3    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0

devasc@labvm:~/labs/devnet-src/ansible/caseStudy$ 
```

Figure 11

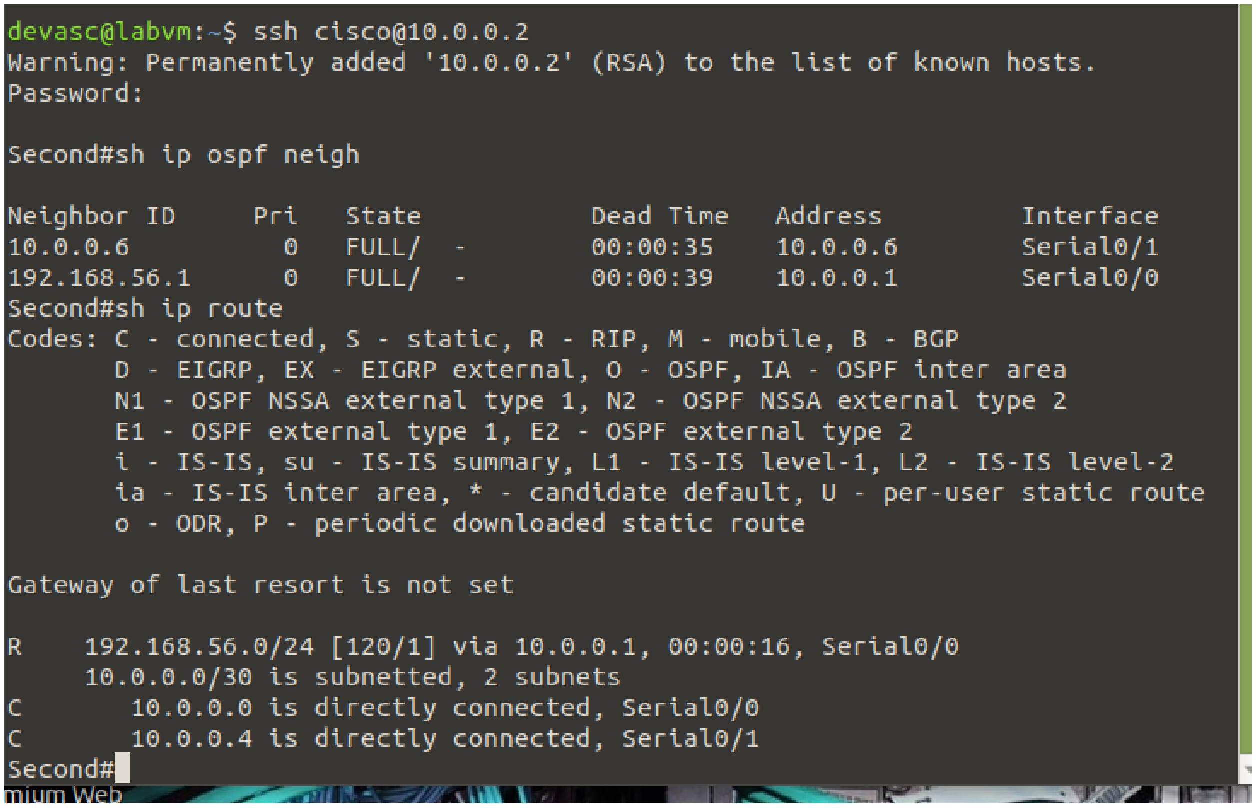
This figure shows the output of the config_ospf.yaml. The user saw the config of each router and displayed running-config, save the output to file, and save the configuration. Lastly, the user saw the changing of the configuration on the play recap



```
devasc@labvm:~$ ssh cisco@192.168.56.1
Warning: Permanently added '192.168.56.1' (RSA) to the list of known hosts.
Password:

First#sh ip ospf neigh

Neighbor ID     Pri   State           Dead Time   Address         Interface
10.0.0.5          0   FULL/  -        00:00:38    10.0.0.2        Serial0/0
First#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

C    192.168.56.0/24 is directly connected, FastEthernet0/0
     10.0.0.0/30 is subnetted, 2 subnets
C       10.0.0.0 is directly connected, Serial0/0
O       10.0.0.4 [110/128] via 10.0.0.2, 00:05:13, Serial0/0
First#Connection to 192.168.56.1 closed by remote host.
Connection to 192.168.56.1 closed.
devasc@labvm:~$
```

Figure 12

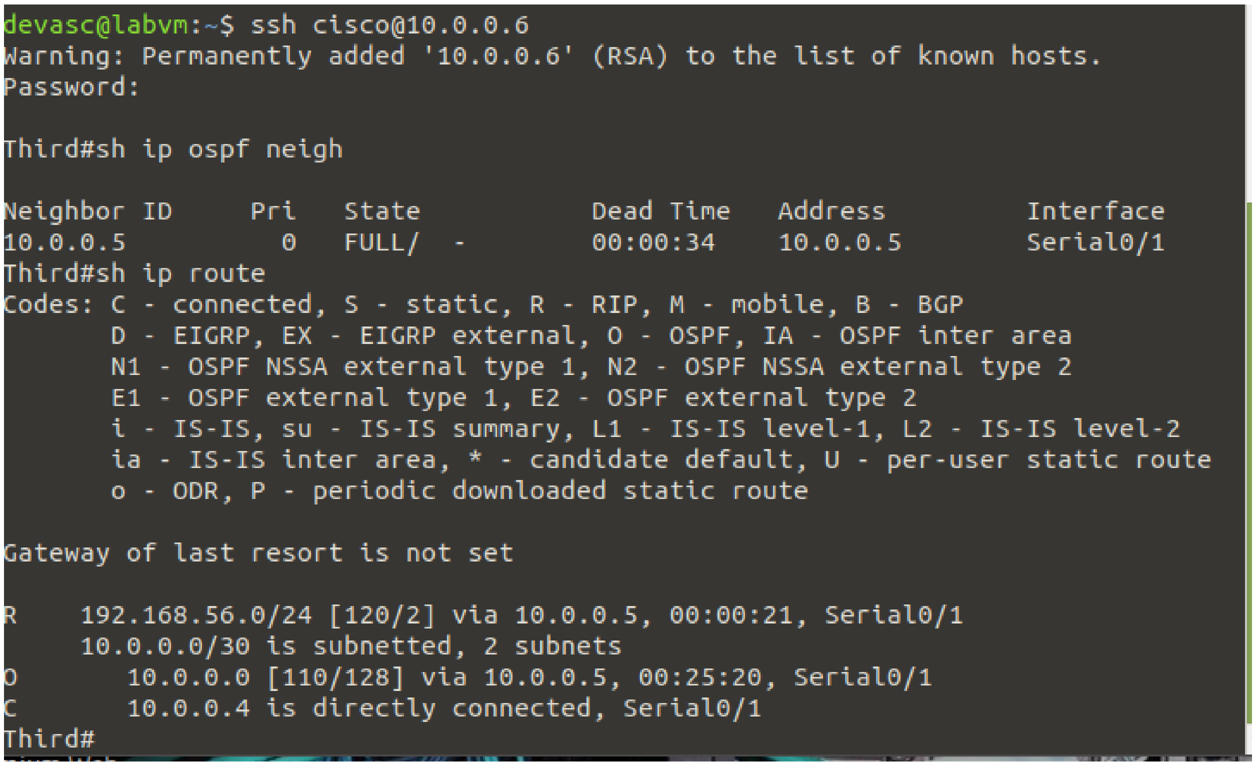
This image shows the IP OSPF neigh and the IP router of the First router.



```
devasc@labvm:~$ ssh cisco@10.0.0.2
Warning: Permanently added '10.0.0.2' (RSA) to the list of known hosts.
Password:

Second#sh ip ospf neigh

Neighbor ID     Pri   State           Dead Time   Address         Interface
10.0.0.6          0   FULL/  -        00:00:35    10.0.0.6        Serial0/1
192.168.56.1      0   FULL/  -        00:00:39    10.0.0.1        Serial0/0
Second#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

R    192.168.56.0/24 [120/1] via 10.0.0.1, 00:00:16, Serial0/0
     10.0.0.0/30 is subnetted, 2 subnets
C       10.0.0.0 is directly connected, Serial0/0
C       10.0.0.4 is directly connected, Serial0/1
Second#
```

Figure 13

This image shows the IP OSPF neigh and the IP router of the second router.

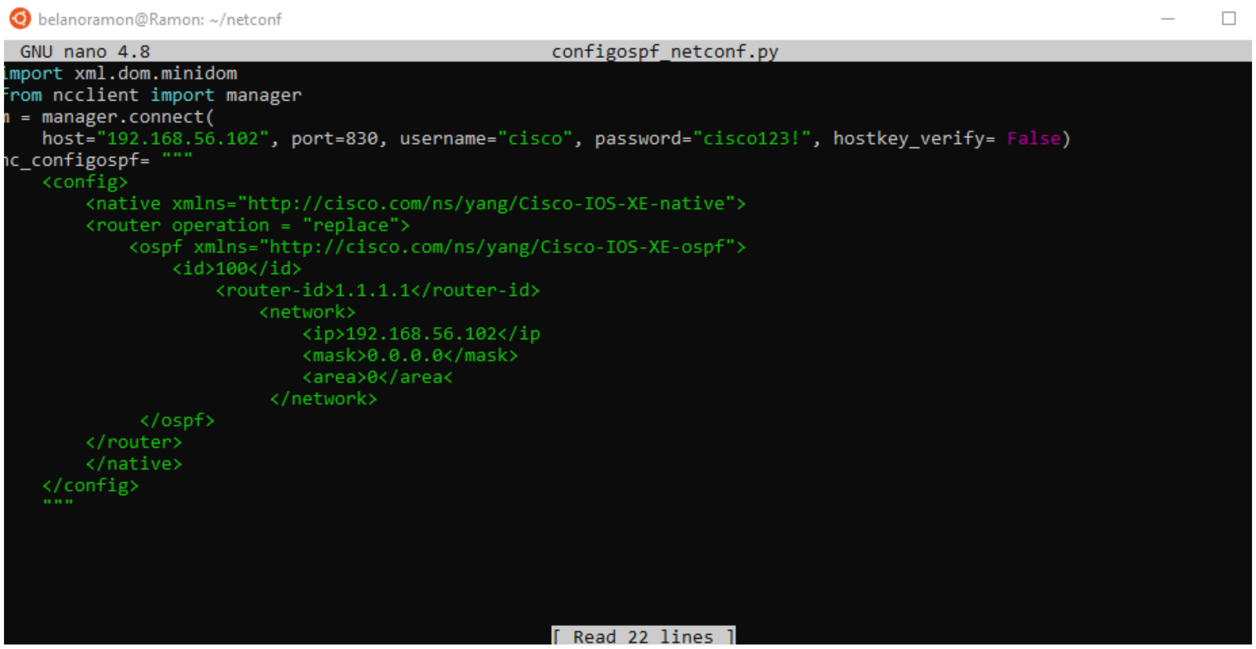


```
devasc@labvm:~$ ssh cisco@10.0.0.6
Warning: Permanently added '10.0.0.6' (RSA) to the list of known hosts.
Password:

Third#sh ip ospf neigh

Neighbor ID     Pri   State           Dead Time   Address         Interface
10.0.0.5          0   FULL/  -        00:00:34    10.0.0.5        Serial0/1
Third#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

R    192.168.56.0/24 [120/2] via 10.0.0.5, 00:00:21, Serial0/1
     10.0.0.0/30 is subnetted, 2 subnets
O       10.0.0.0 [110/128] via 10.0.0.5, 00:25:20, Serial0/1
C       10.0.0.4 is directly connected, Serial0/1
Third#
```

Figure 14

This image shows the IP OSPF neigh and the IP router of the Third router.

Part 2 The Network Programmability using Netconf


belanoramon@Ramon: ~/netconf

```
  GNU nano 4.8                          configospf_netconf.py
import xml.dom.minidom
from ncclient import manager
m = manager.connect(
   host="192.168.56.102", port=830, username="cisco", password="cisco123!", hostkey_verify= False)
nc_configospf= """
   <config>
       <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
       <router operation = "replace">
           <ospf xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-ospf">
               <id>100</id>
                   <router-id>1.1.1.1</router-id>
                       <network>
                           <ip>192.168.56.102</ip
                           <mask>0.0.0.0</mask>
                           <area>0</area<
                        </network>
            </ospf>
       </router>
       </native>
   </config>
   """

                                    [ Read 22 lines ]
```

Figure 15

First, install ncclient, and after the user installs the ncclient, the second step is a configuration set the router OSPF to 100 the add a router-id the add the network, mask and are.

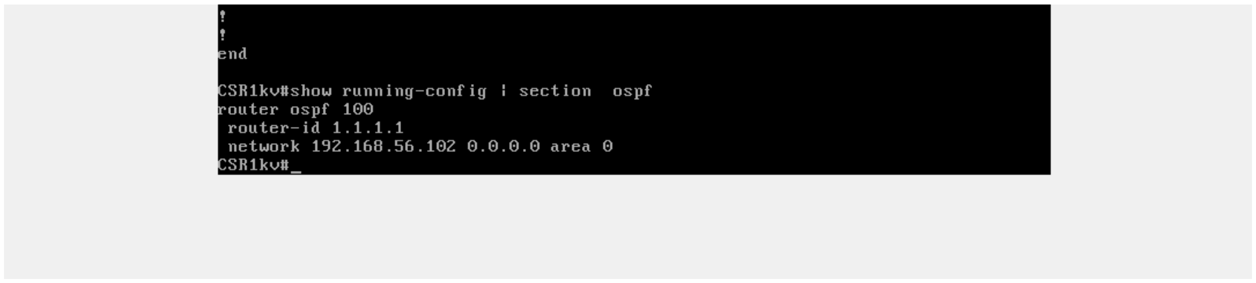

```
!
!
end

CSR1kv#show running-config | section  ospf
router ospf 100
 router-id 1.1.1.1
 network 192.168.56.102 0.0.0.0 area 0
CSR1kv#_
```

Figure 16

This image shows that the configure was successful.