TLEN 5410 – Network Management and Automation

Lab 8

DevOps-Ansible, Automation, and Networking

University of Colorado Boulder

Interdisciplinary Telecom Program

Professor Levi Perigo, Ph.D.

# Summary

Automation is an important component to achieve new data center deployments in short time intervals. Data centers require similar type devices to be configured multiple numbers of times and automation eliminates human intervention process of logging into devices using a console for configuration purpose.

# Objectives

* Create Ansible playbooks using a Python script based on your network topology and the router configuration requirements.
* Create Jinja2 templates based on your final router configuration requirements.
* Create a final router configuration file using Ansible and a Jinja2 template.
* Learn to use Ansible, Jinja2, and Python to automate network deployments and device configurations.

Problem Statement:

In your previous labs, you worked on automating the complete network in your data center using Netmiko and you have also worked using Ansible for configuration and package management. In this lab, you will automate new routers deployment in your data center network using Python, Ansible playbooks, and Jinja2 templates.

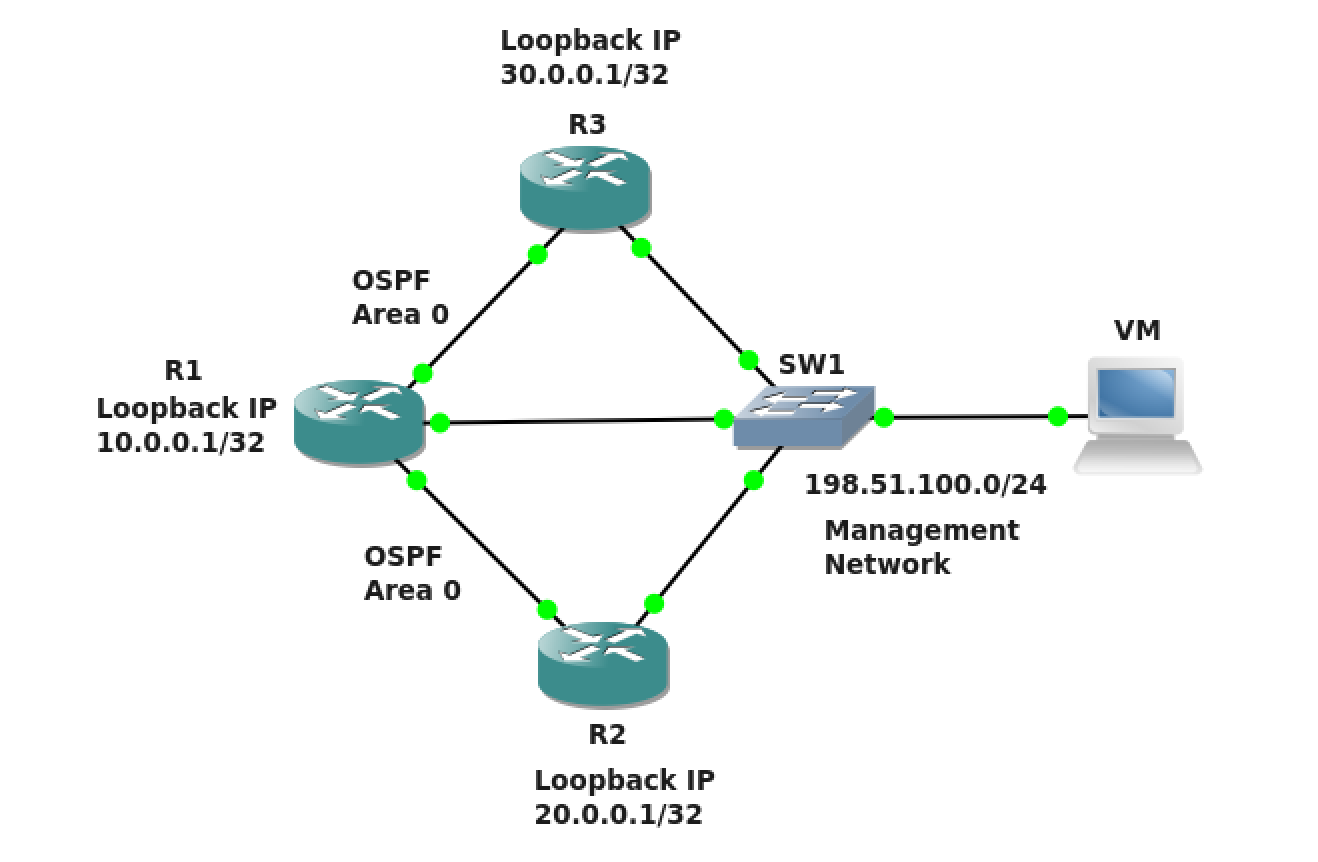


Figure 1

Guidelines:

1. Create the above topology in GNS3 on the NetMan VM. Configure the management network and enable SSH on R1, R2, and R3. Do not configure anything else.

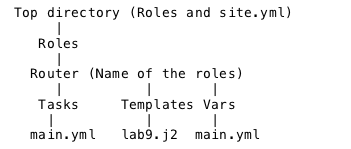
Ansible and Jinja2 Help:

1. https://pynet.twb-tech.com/blog/ansible/ansible-cfg-template.html

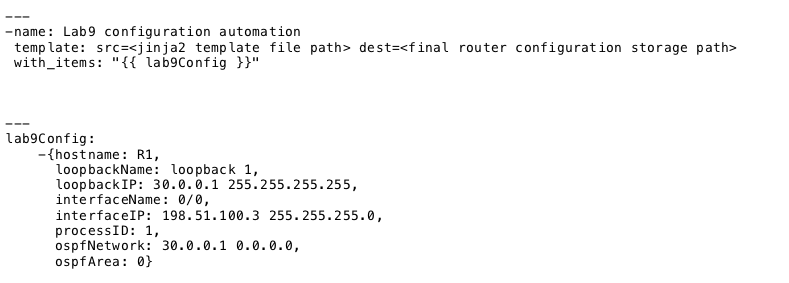
Ansible Example:

Use Ansible Roles directory as the main directory containing Tasks, Templates, and Vars as subdirectories to deploy your automation. Refer to the above Ansible automation reference link for guidance. The below Ansible playbook examples are only for your reference and your Ansible playbooks generated using Python will not be exactly similar to these, they will contain additional elements according to the configuration requirements.

The following is the directory structure required. Follow the above reference link for more information.

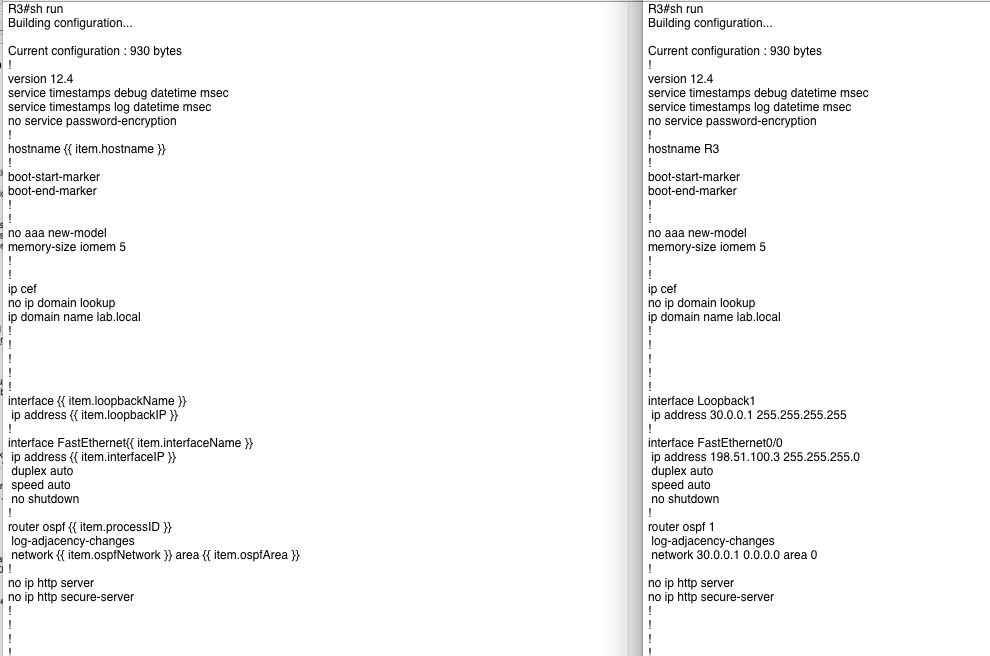


The following are two separate example YAML files. When the first file is executed, it will call the second variables containing file (created using Python). Follow the above reference link for more information.



Jinja2 Example:

The below Jinja2 template example (lab9.j2) is only for your reference and your Jinja2 template will not be exactly similar to this, it will consist of additional elements according to the configuration requirements.

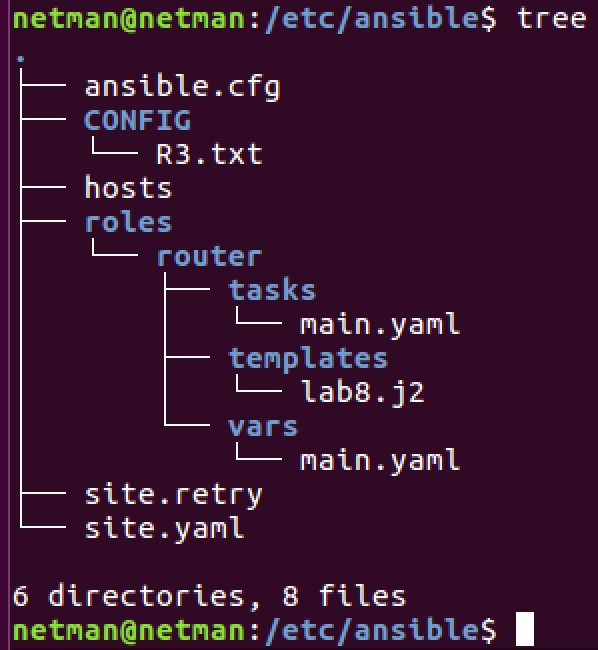


Lab configuration requirements:

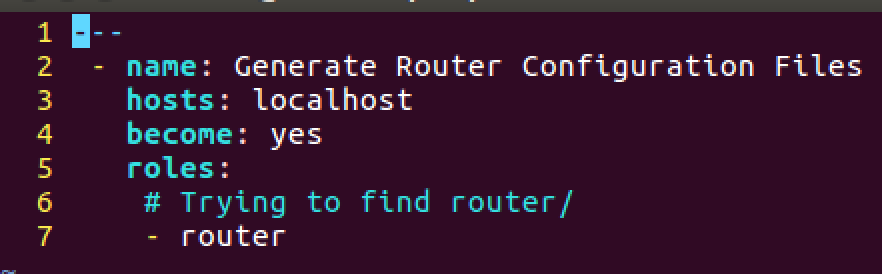
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Hostname | Interface Type | Interface Name | IP/Subnet | OSPF Enabled | OSPF Process ID | OSPF Area |
| R1 | Loopback | 1 | 10.0.0.1/32 | Yes | 1 | 0 |
| R1 | FastEthernet | 0/0 | 198.51.100.3/24 | Yes | 1 | 0 |
| R1 | FastEthernet | 1/0 | 198.51.101.3/24 | Yes | 1 | 0 |
| R1 | FastEthernet | 2/0 | 198.51.102.3/24 | Yes | 1 | 0 |
| R2 | Loopback | 1 | 20.0.0.1/32 | Yes | 2 | 0 |
| R2 | FastEthernet | 0/0 | 198.51.100.4/24 | Yes | 2 | 0 |
| R2 | FastEthernet | 1/0 | 198.51.101.4/24 | Yes | 2 | 0 |
| R3 | Loopback | 1 | 30.0.0.1/32 | Yes | 3 | 0 |
| R3 | FastEthernet | 0/0 | 198.51.100.5/24 | Yes | 3 | 0 |
| R3 | FastEthernet | 1/0 | 198.51.102.5/24 | Yes | 3 | 0 |

Steps:

1. Make a directories like the following pictures:

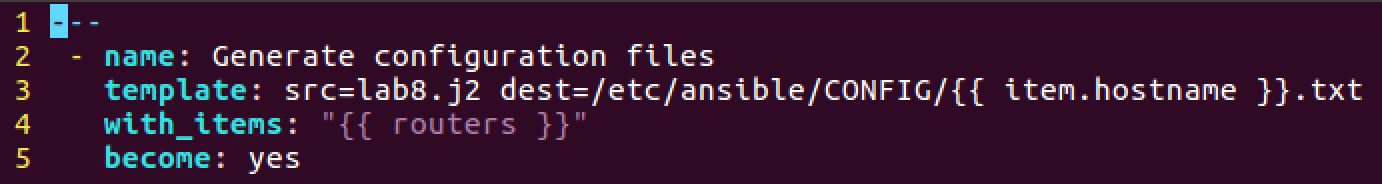
 note: site.retry will generate once you run site.yaml

2. Playbook on site.yaml



Note: the ‘router’ under roles: will try to locate the router/ above

3. main.yaml under /tasks/

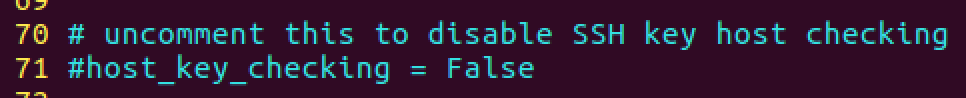


Note: src=lab8.j2 will go to src to find the file, and store in dest once it’s done jobs. You need to create a folder if you don’t have any and if you do the way like above /etc/ansible/<folder name>

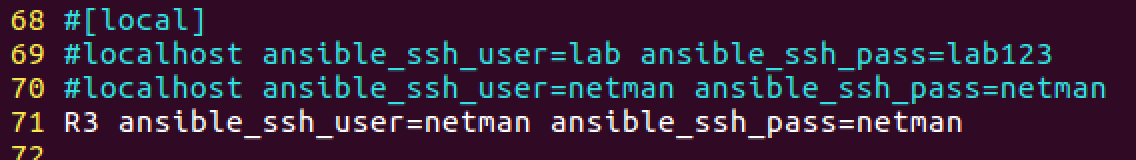
4. lab8.j2 under /templates/

First,

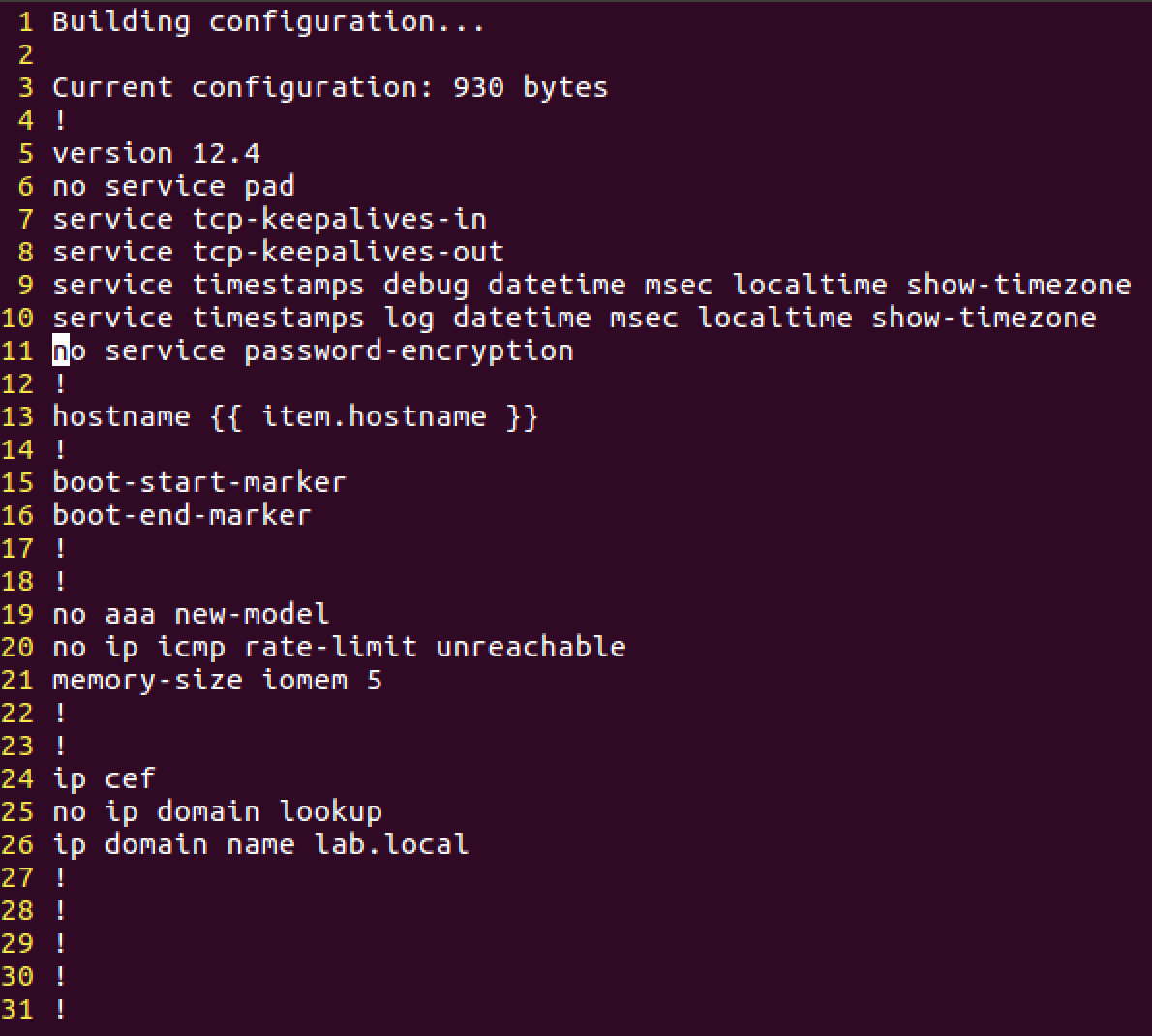
Note: I need to comment out the following in /etc/ansible/ansible.cfg



And configure the following in /etc/ansible/hosts:

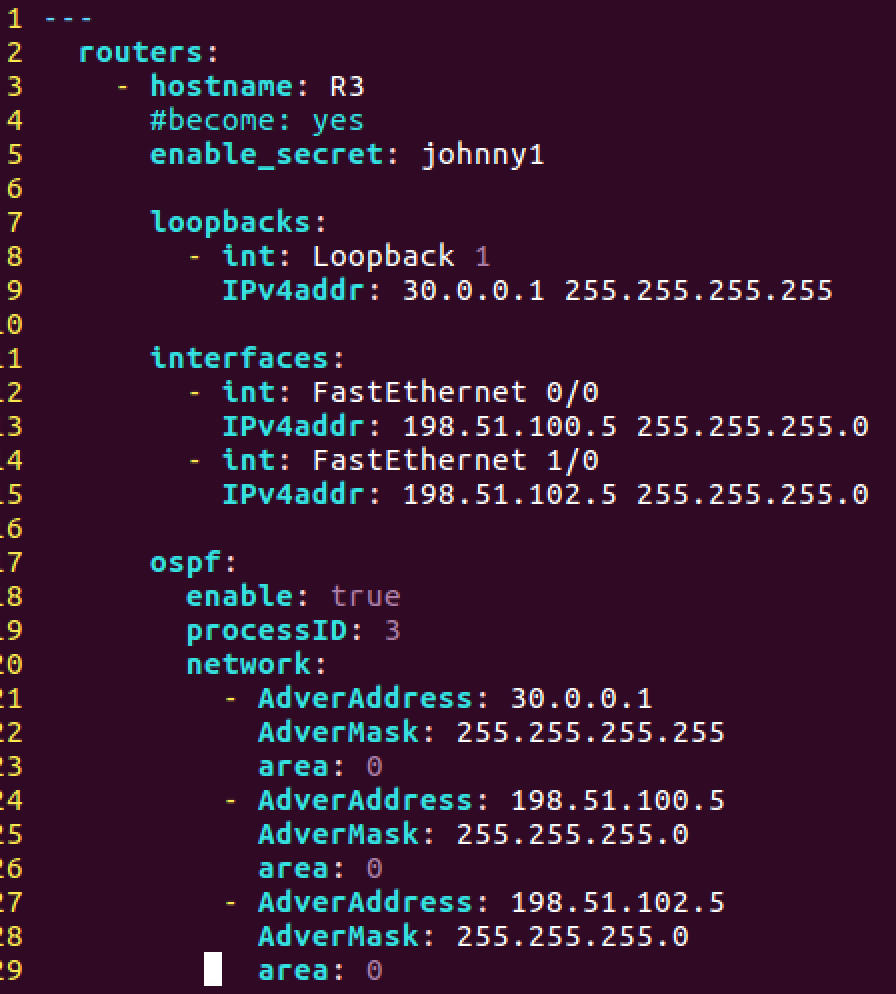


Then, create a j2 file:



5. main.yaml under vars/

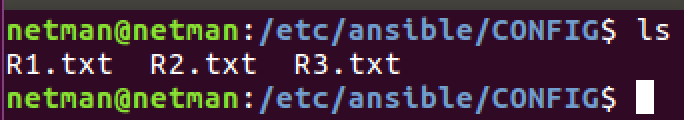
Same follows for R1 and R2. Just build the same logic with requirement underneath



And run the ‘sudo ansible-playbook site.yaml’

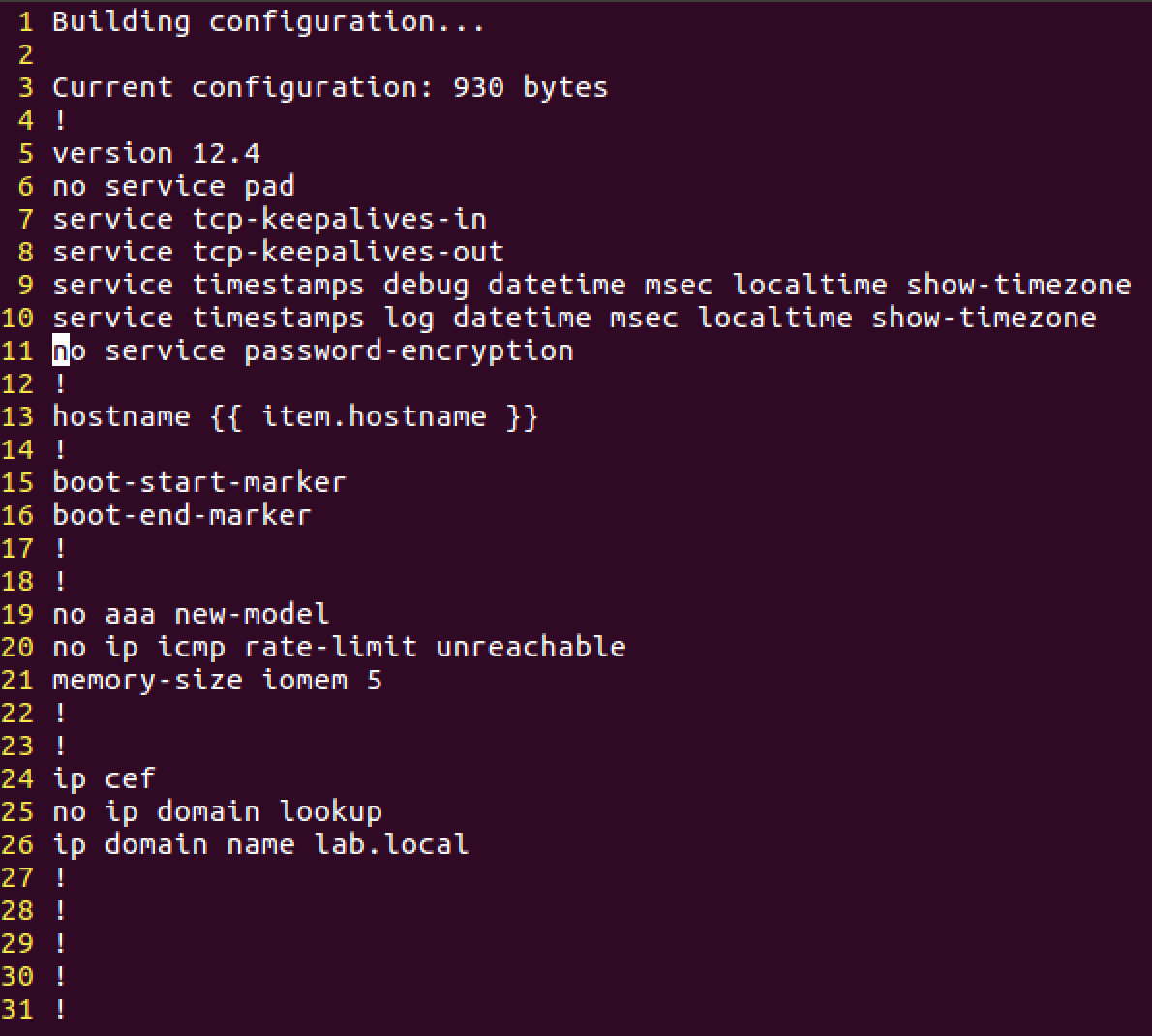


You should see 3 files under CONFIG/



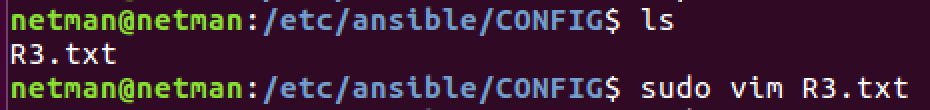
Objective 1: Using the given topology, create a Jinja2 template for the Cisco routers (R1, R2, and R3). Do not configure the Cisco routers in GNS3 in the NetMan VM manually. **[50 Points]**

R3 template:

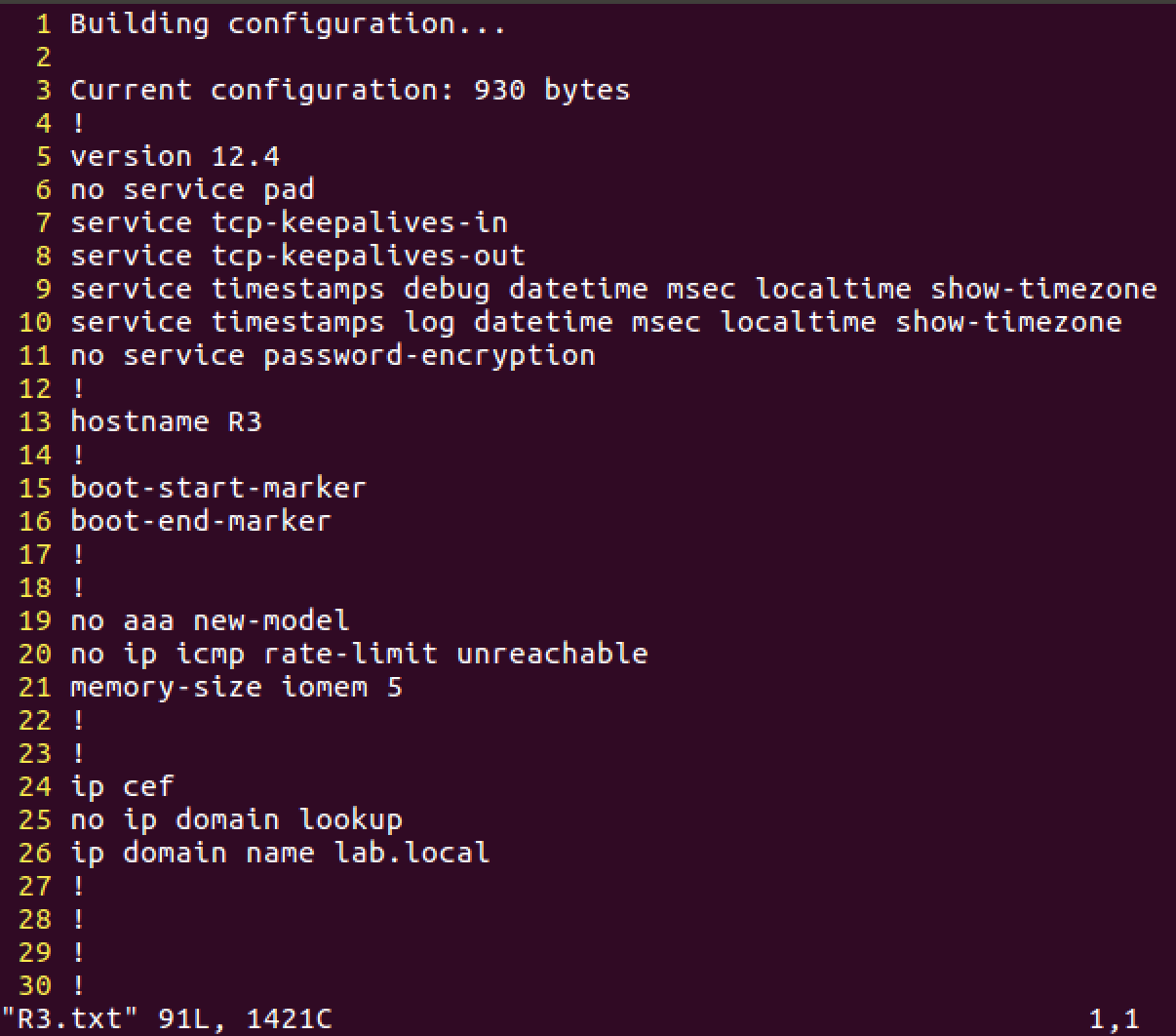




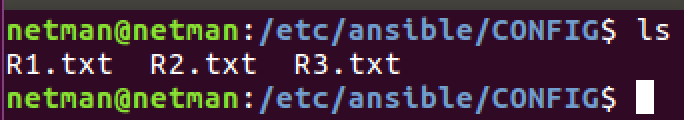
After run ansible-playbook site.yaml, it will generate R3.txt under /etc/ansible/CONFIG/



R3.txt:

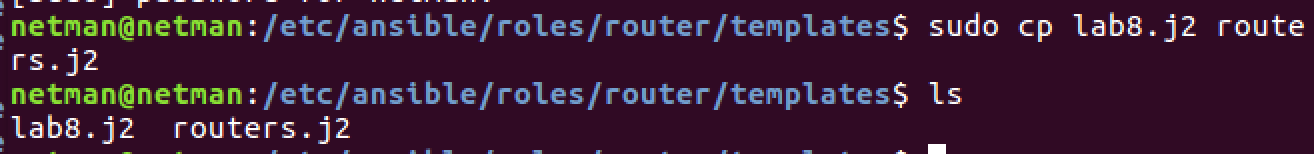


Final result:

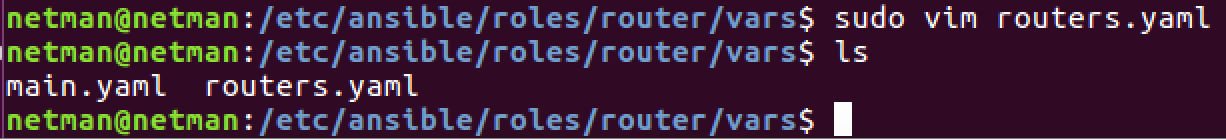


Objective 2: Using the configuration requirements (CSV), create an Ansible playbook in YAML format with the help of a Python script. Follow the Ansible “Roles” directory framework for this objective. This Ansible playbook and Jinja2 templates for R1, R2, and R3 will be used to create the actual router configuration files to configure the above topology in GNS3 in your NetMan VM. **[50 Points]**

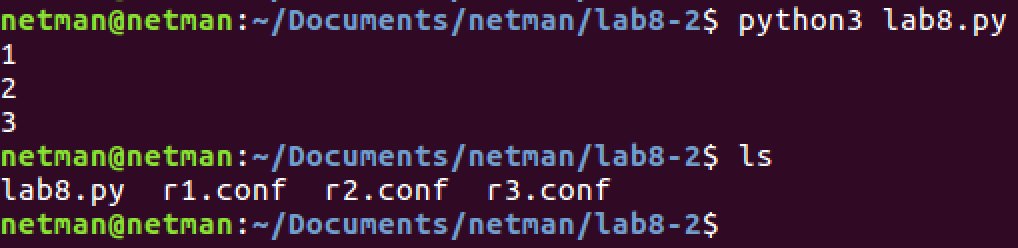
Make a new jinja2 file called routers.j2



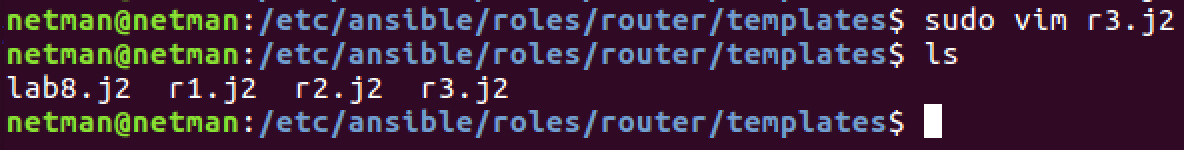
One yaml file (routers.yaml) for 3 routers:



After run lab8.py file:

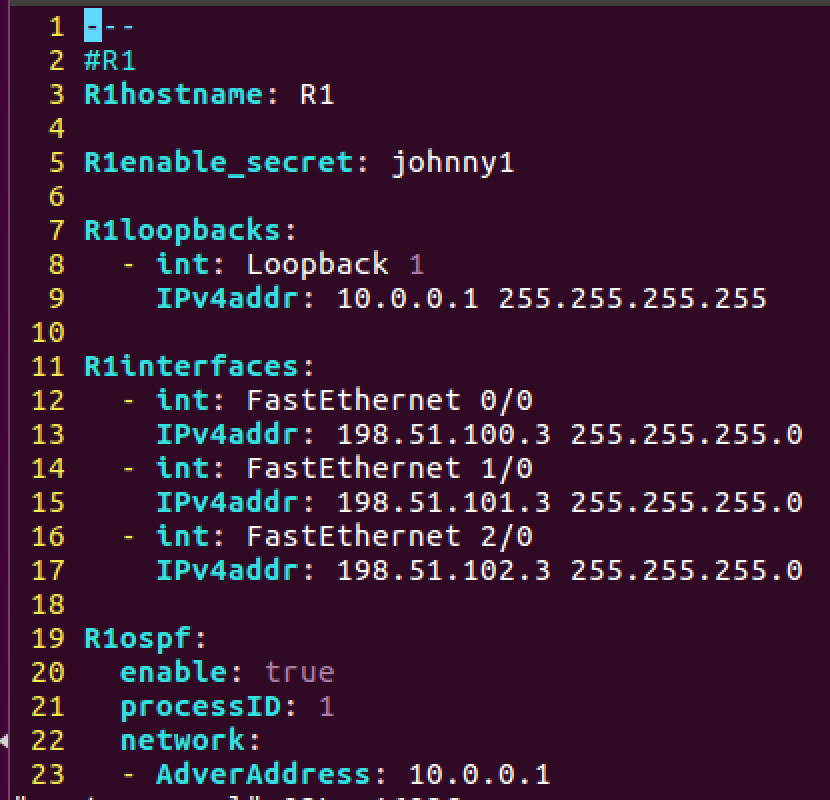


Create 3 jinja2 files for R1, R2, R3:

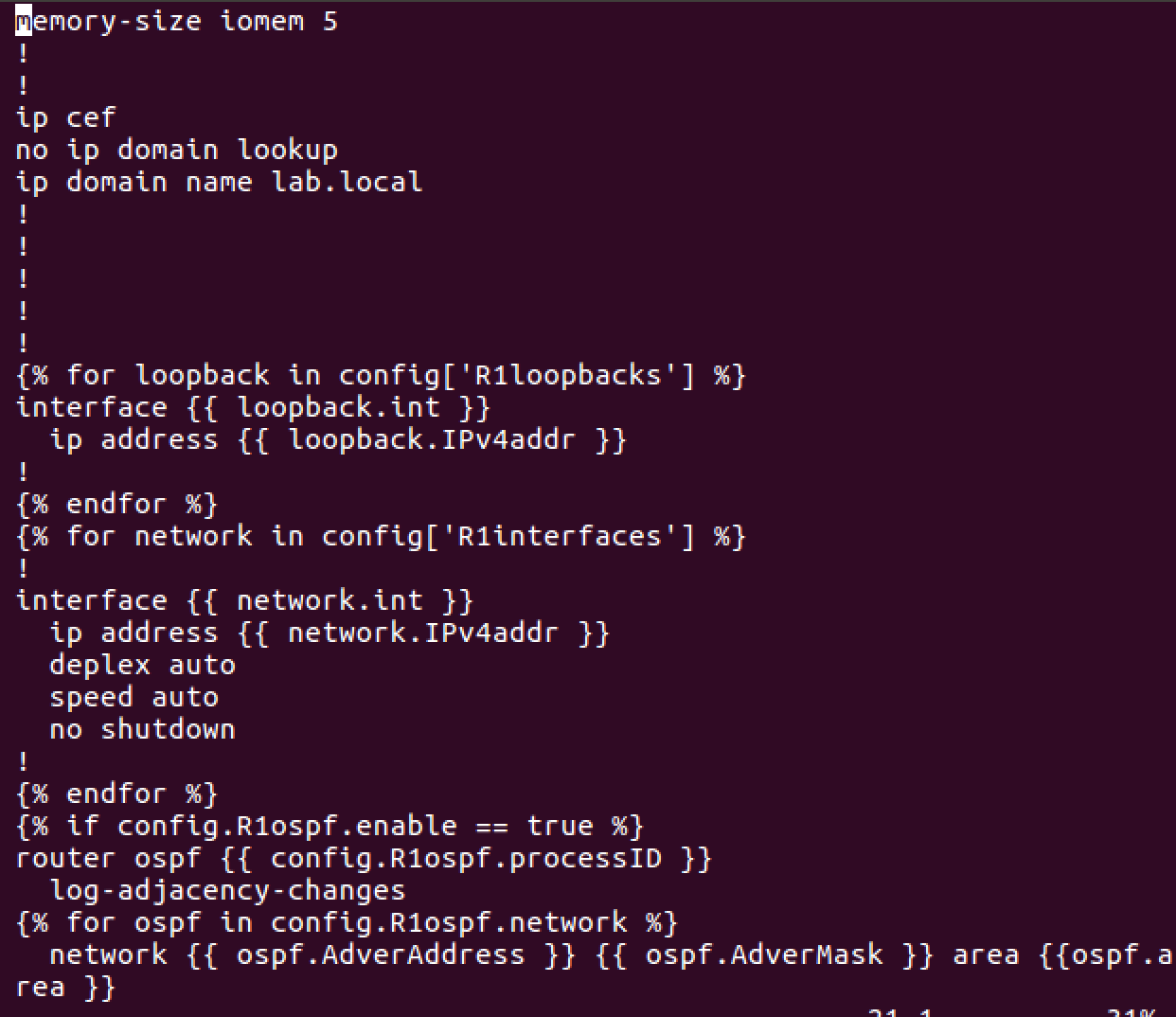


In order to make python to notice which variable I was referring to, I make a separate variable name representing each router such as R1loopbacks or R1 interfaces:

For example: (routers.yaml)



And in R1: (R1.j2)



Objective 3: Configure R1, R2, and R3 in GNS3 in your NetMan VM using Netmiko and achieve complete reachability to the real and loopback interfaces of R1, R2, and R3 from your NetMan VM terminal. **[50 Points]**

First, I made sure the ssh has configured on R1, R2, and R3:

Ssh configuration steps:

Username lab privilege 15 password lab123

Ip domain-name lab

Crypto key generate rsa (select 1024 as the option)

Ip ssh version 2

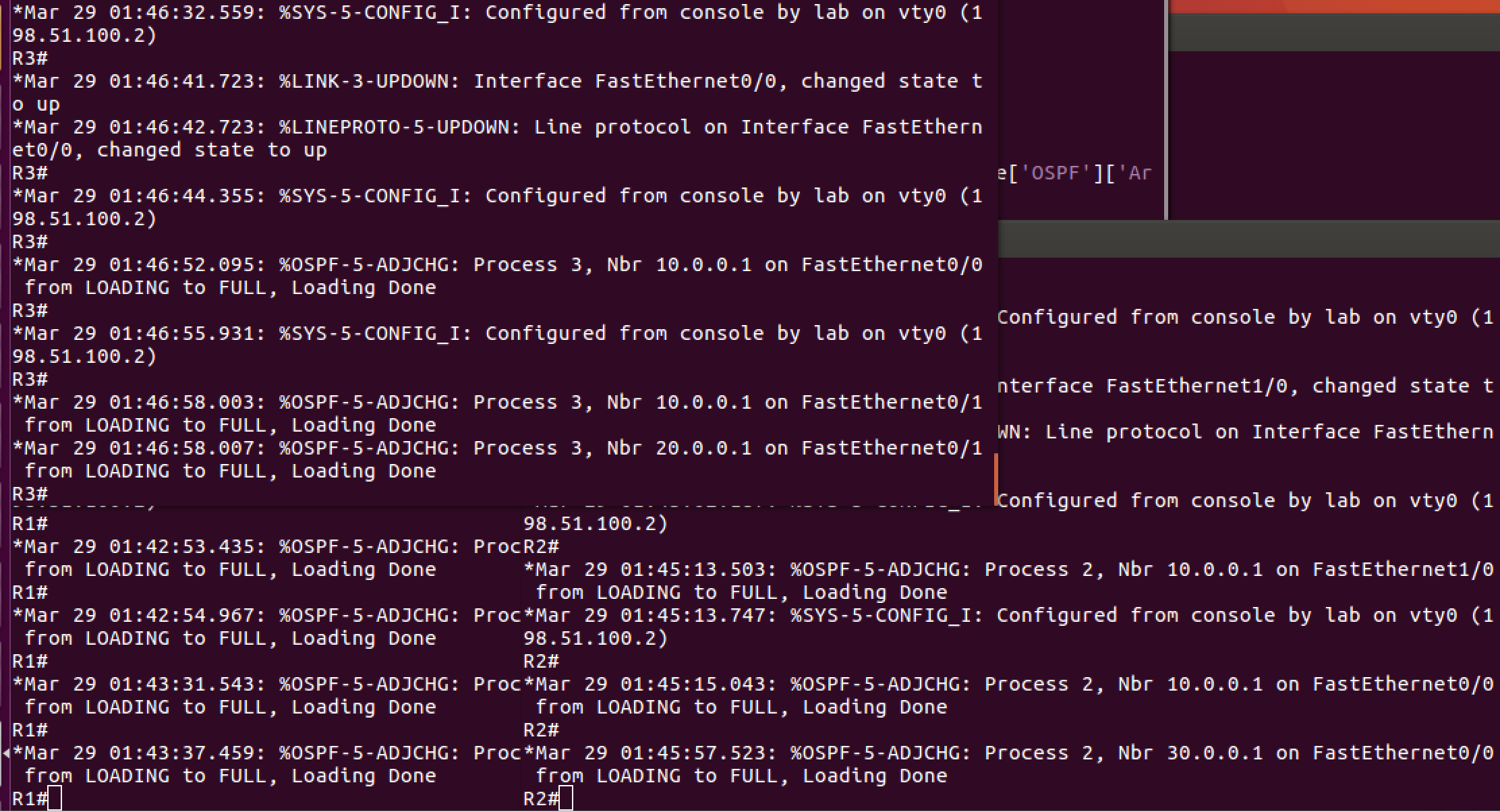
Line vty 0 15

Login local

Transport input all

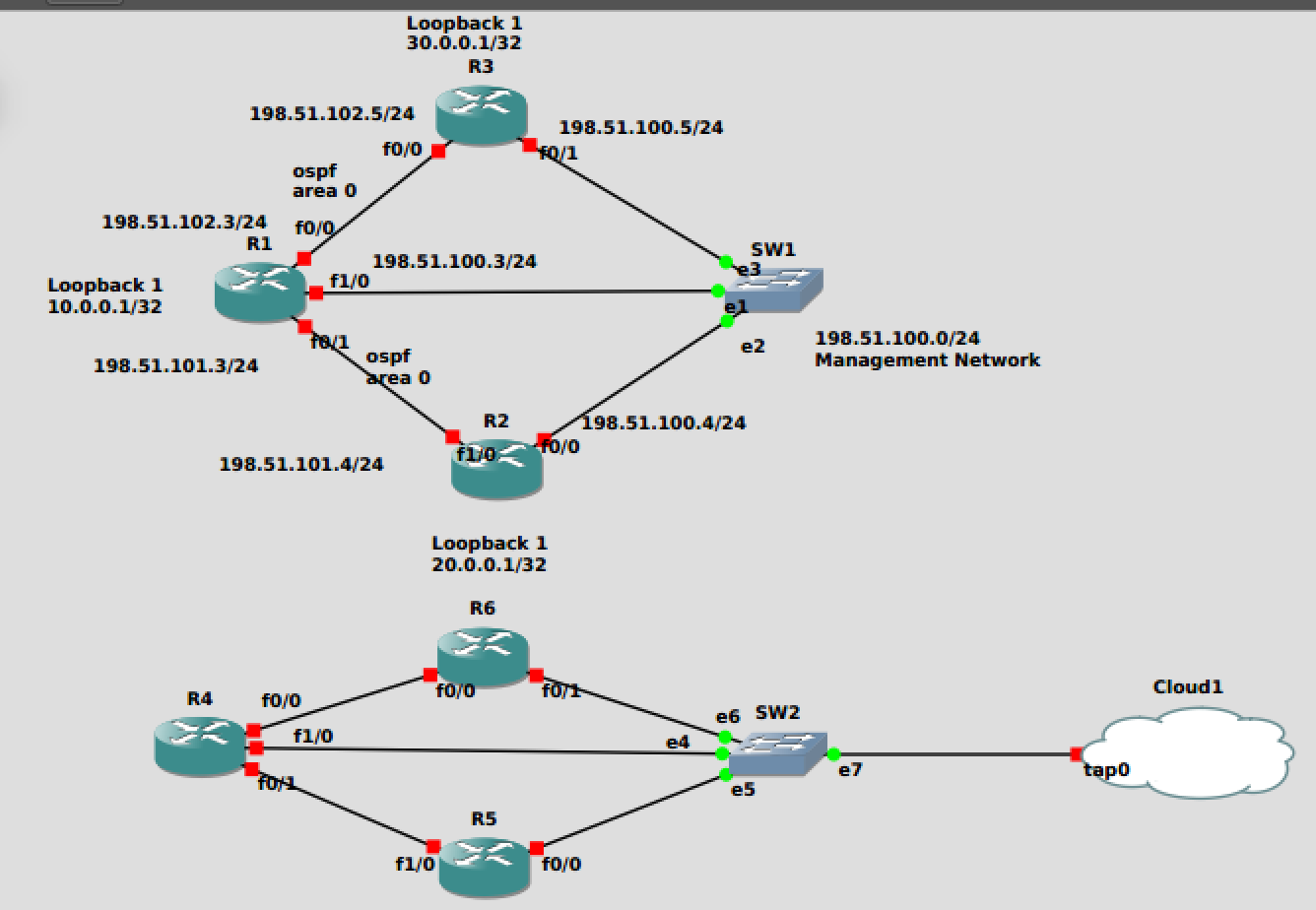
Second, I configured interfaces on interfaces of each router facing the VM:

Result after running script:



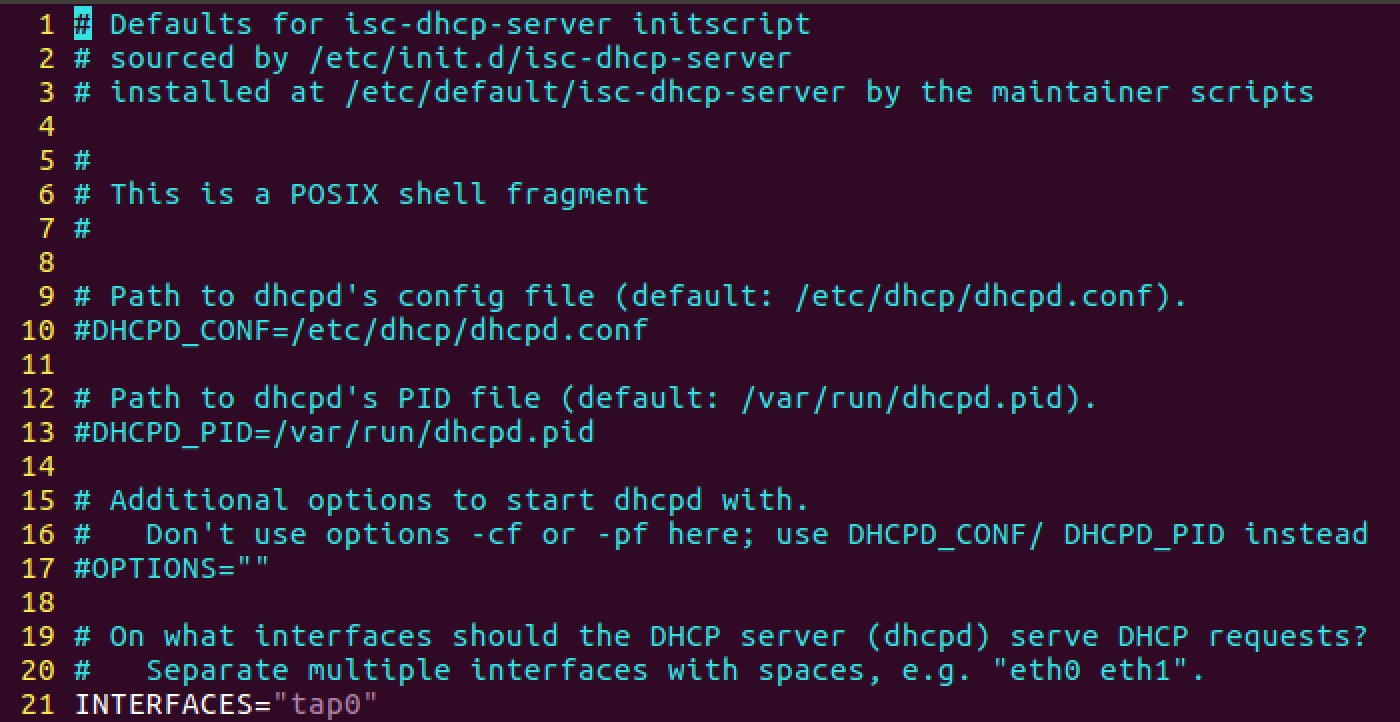
Extra Credit I: Configure R1, R2, and R3 by pulling the new configuration file generated in Objective 2 using ZTP. Configure a DHCP server to also act as a TFTP server on your NetMan Ubuntu machine and configure R1, R2, and R3 as DHCP clients to request IP address and the new configuration file from Objective 2. Use below links for reference and also search the Internet for more help and reference. **[25 Points]**

First of all, I created another topology just in case if the new topology is not working.



Second of all, I downloaded the dhcp package for Ubuntu by ‘sudo apt install isc-dhcp-server’

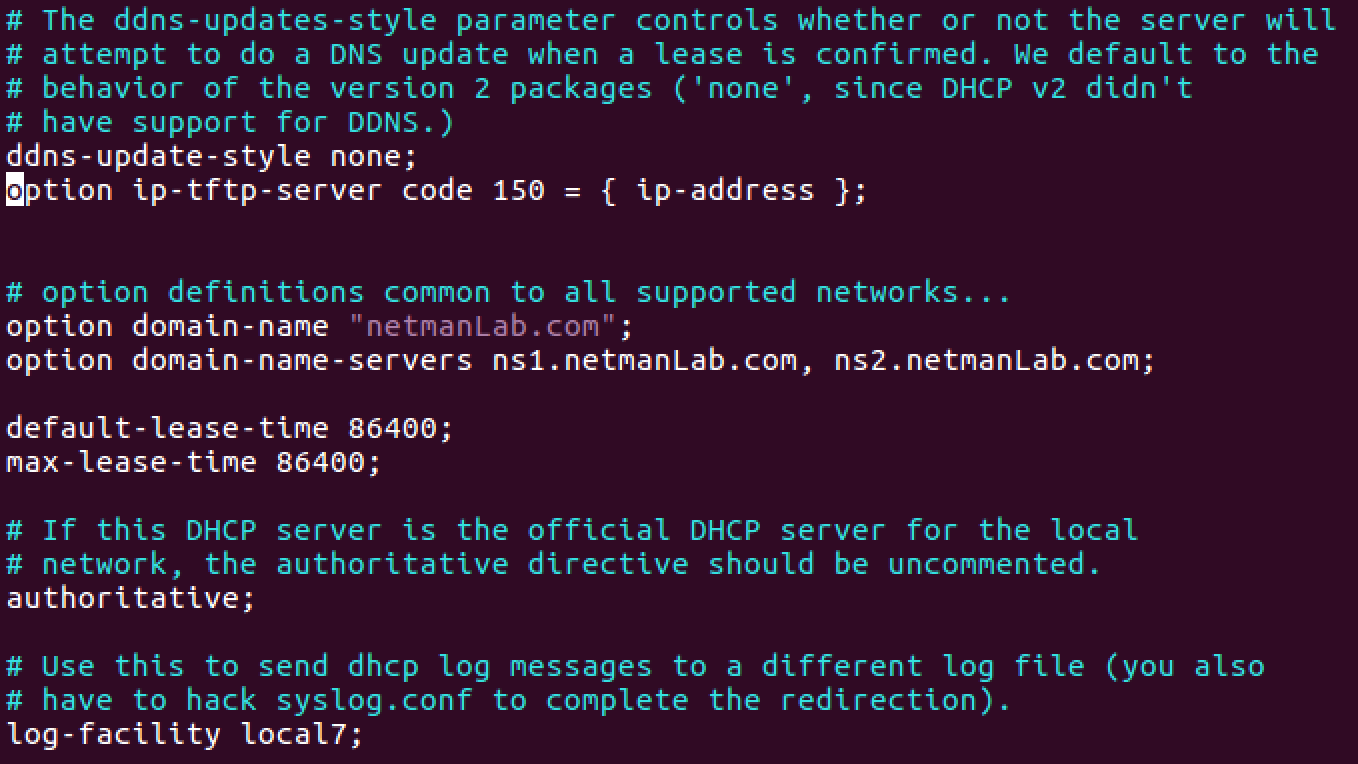
,then modify isc-dhcp-server file under /etc/default/ , change to the interface to be tap0 like the following:



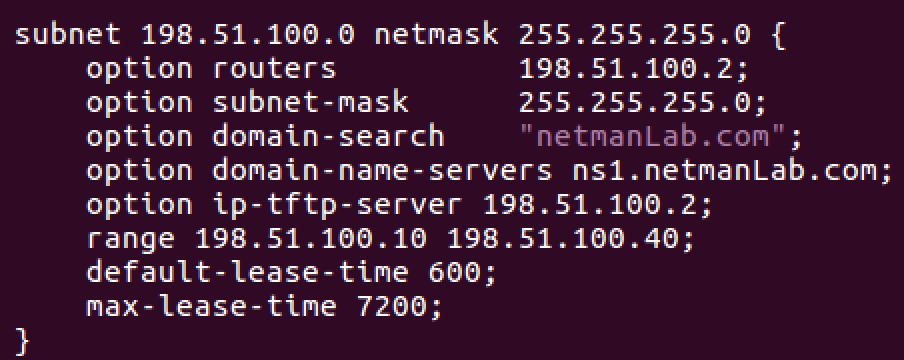
Then, modify a file in:



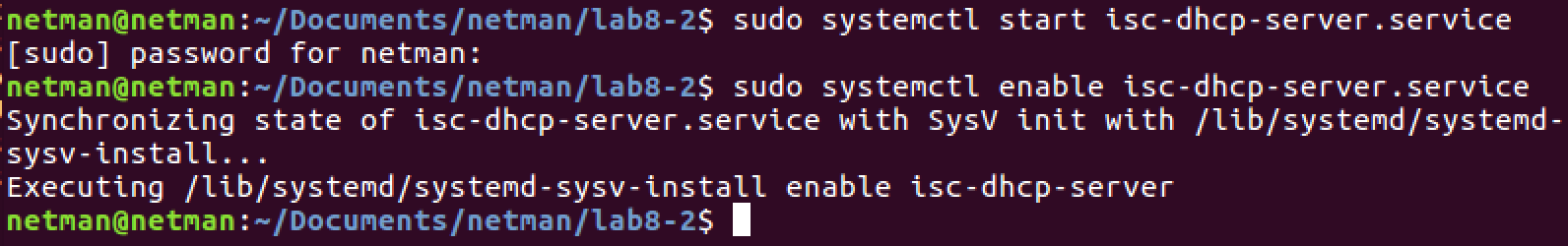
To be like the following:



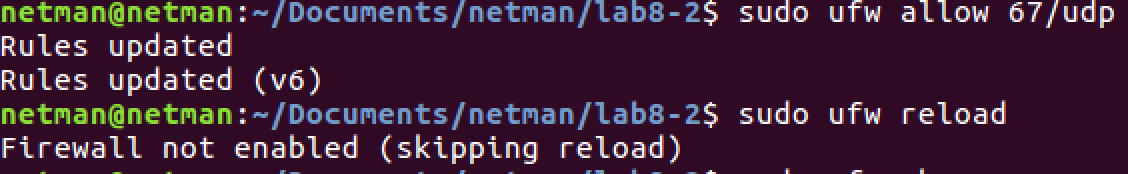
Then add the following config inside of dhcpd.conf



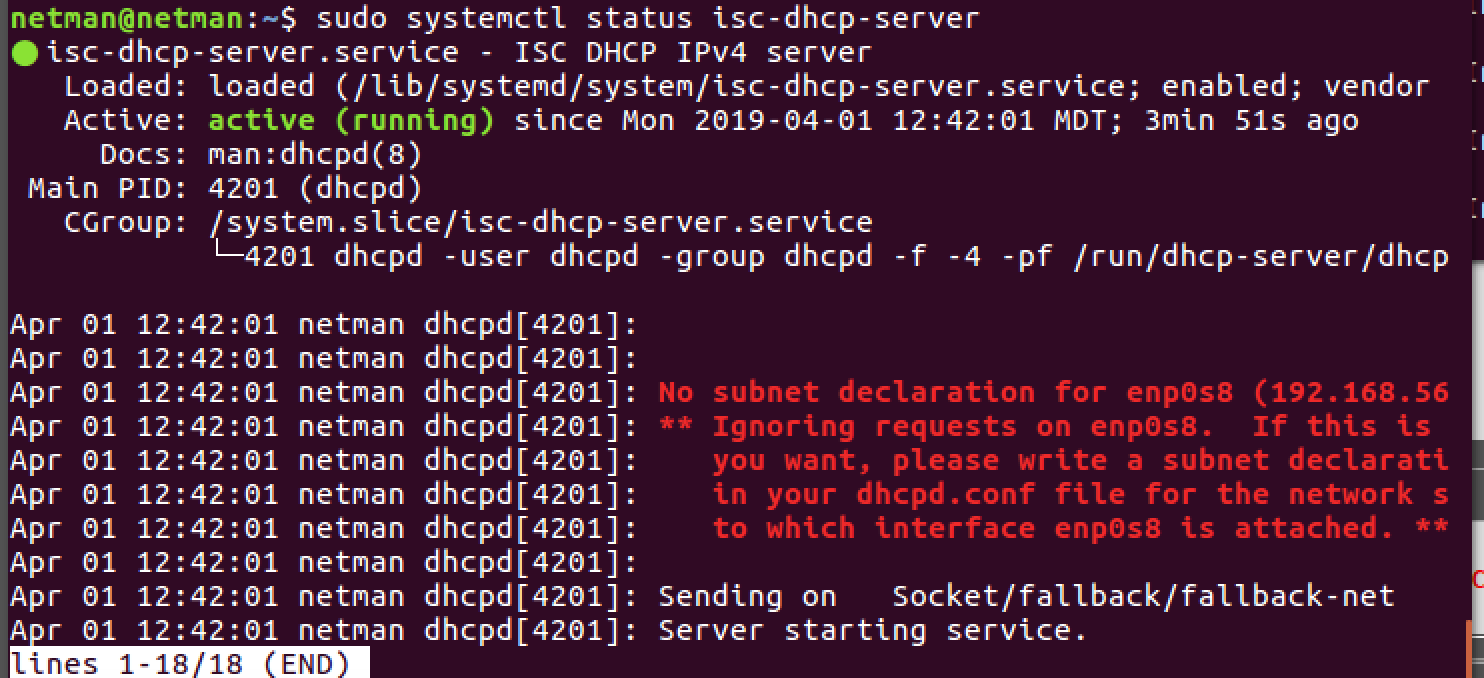
After that, start the server:



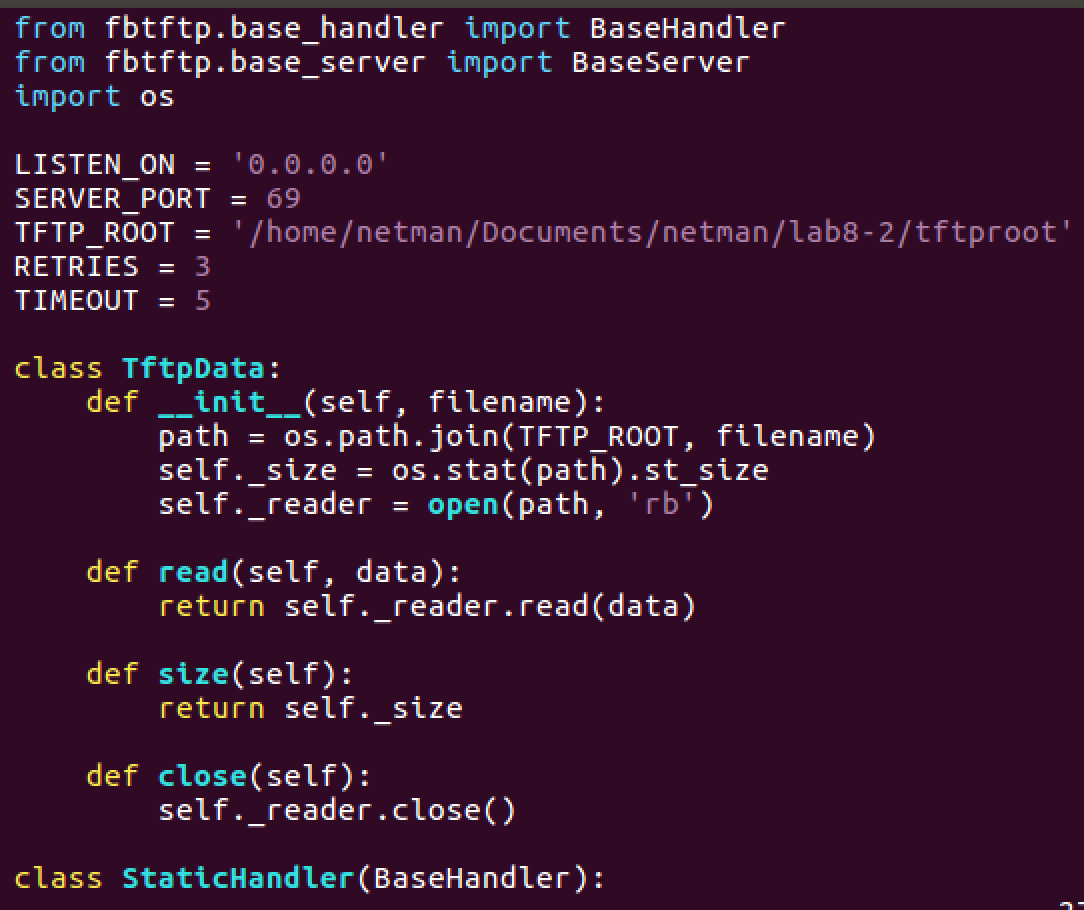
Next, permit DHCP service to listen port 67:

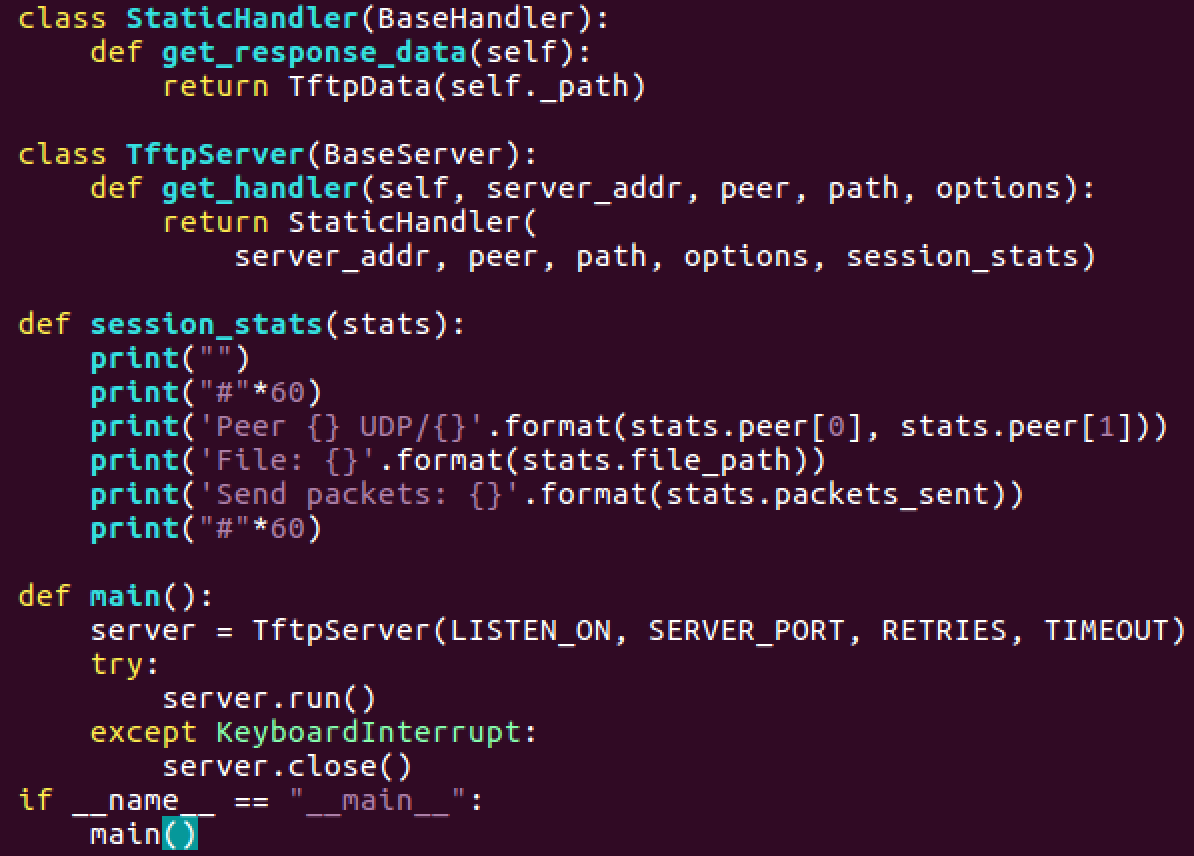


Double check if dhcp server is up:

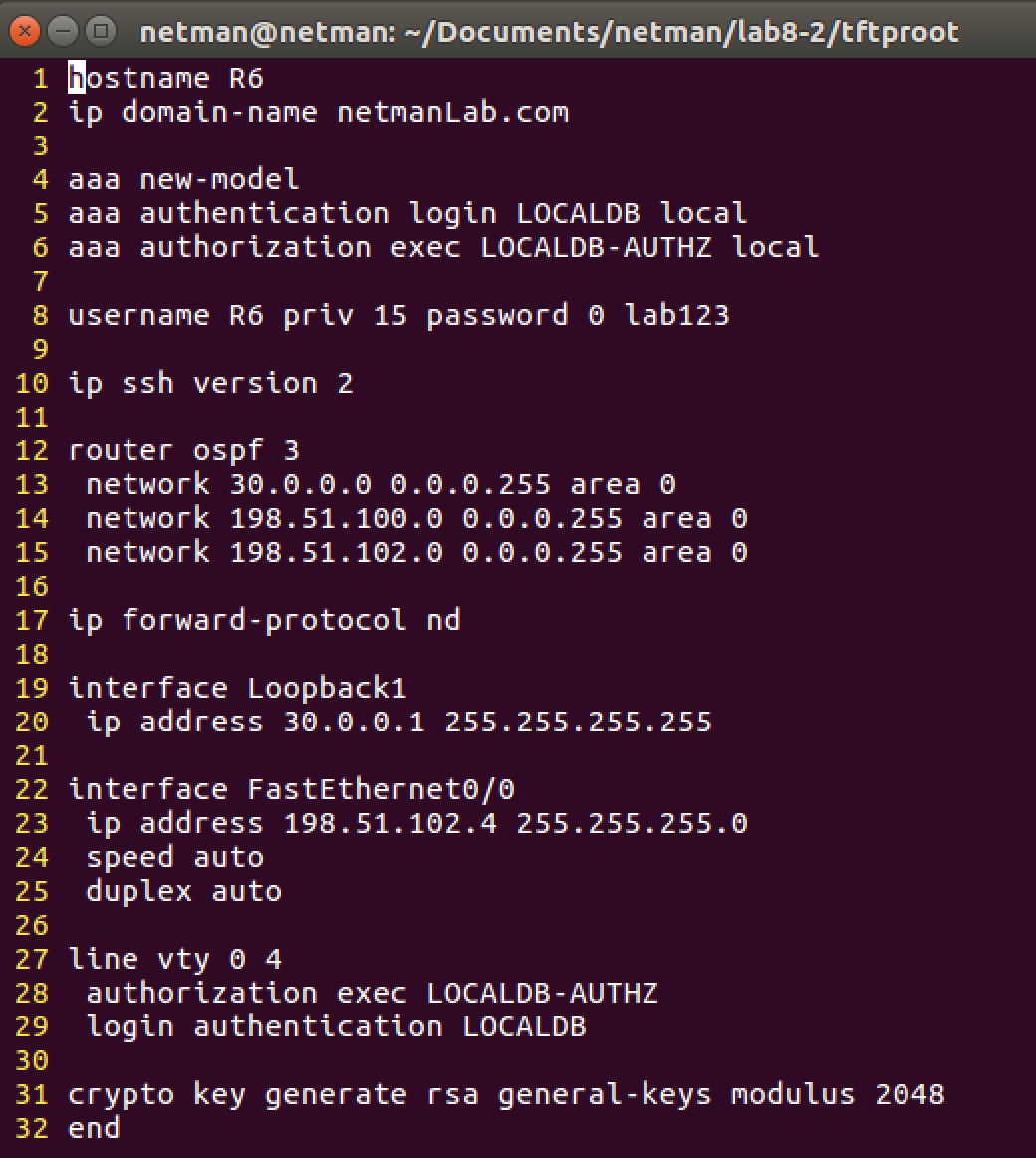


I wrote a python file to run the tftp server:





Notice that because my TFTP\_ROOT variable is going to a directory, and cisco router will look for a config file called network-config, put the following in the file: (Modify it if needed)



To run tftp server, do the followings:

apt-get update

apt-get install python3-venv

python3 -m venv venv

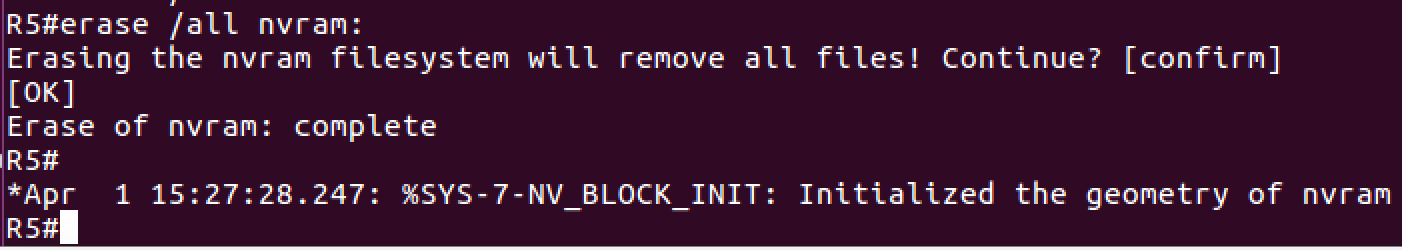
source venv/bin/activate

pip install fbtftp

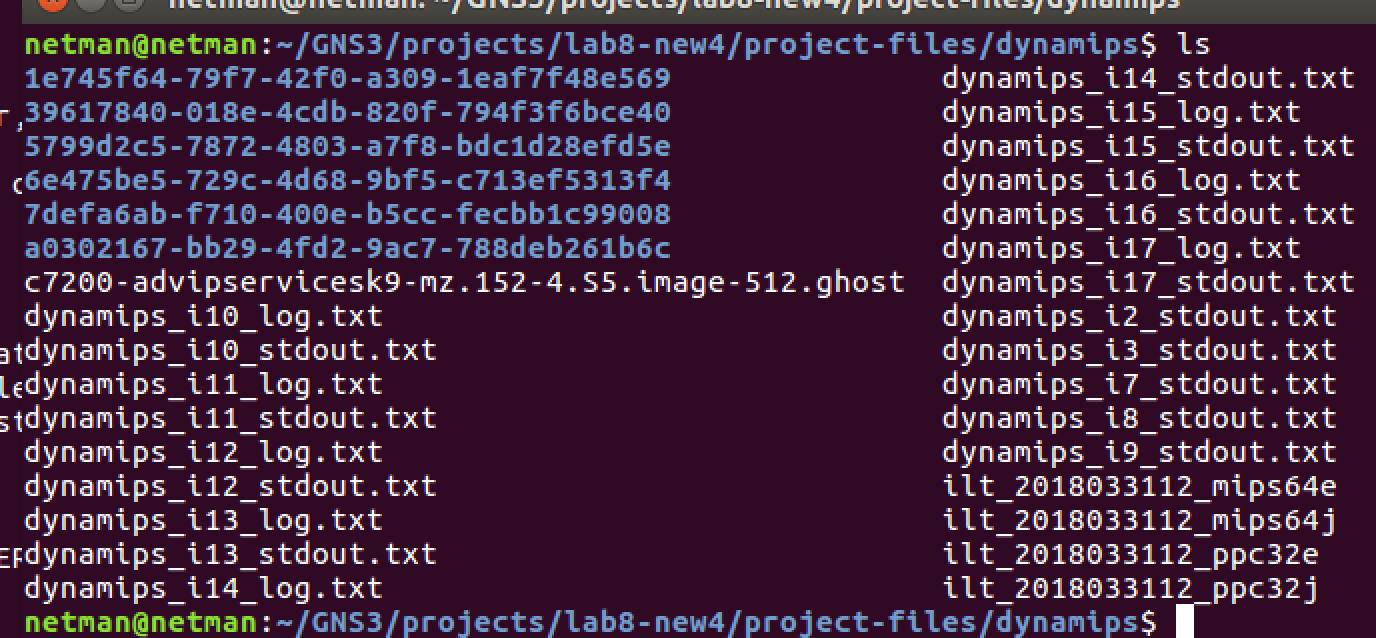
sudo python3 ztp.py

Note: If you get a warning while running python script, that’s fine.

Next, try to boot up all R4, R5, R6 routers and when it’s done booting, do ‘write’ command first, and then:

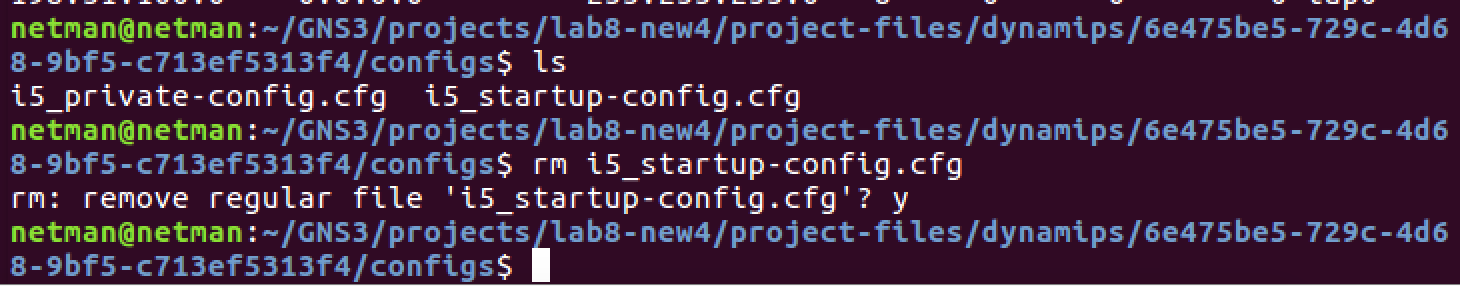


Next go to /GNS3/projects/lab8-new4/project-files/dynamips/



All blues directories are router files, go find for R4, R5, and R6 and go into configs/ and delete startup-config file like the picture:

Note: a0302167-xxx… is my R4, 6e475be5-xxx… is my R5, 5799d2c5-xxx… is my R6.

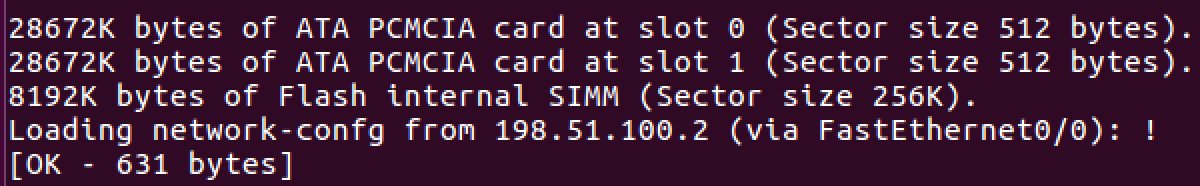


Now, R5 should be clean, and reboot R5.

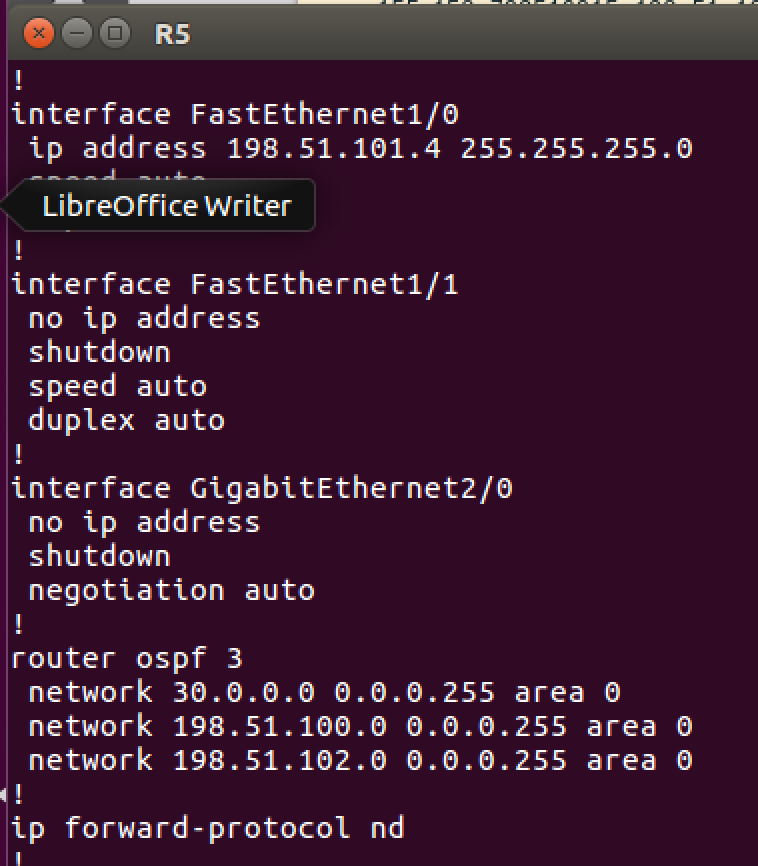
You can see in wireshark, it will request file for R5:



In R5 console:

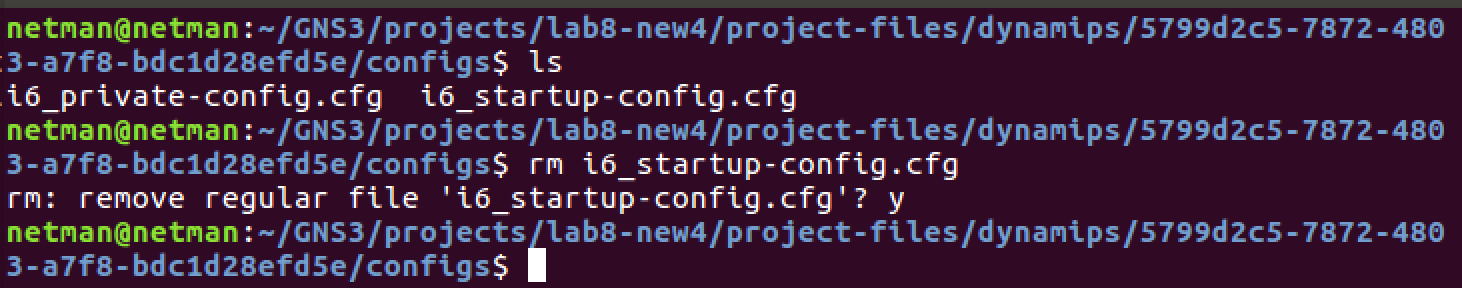


Show run on R5:



R6:

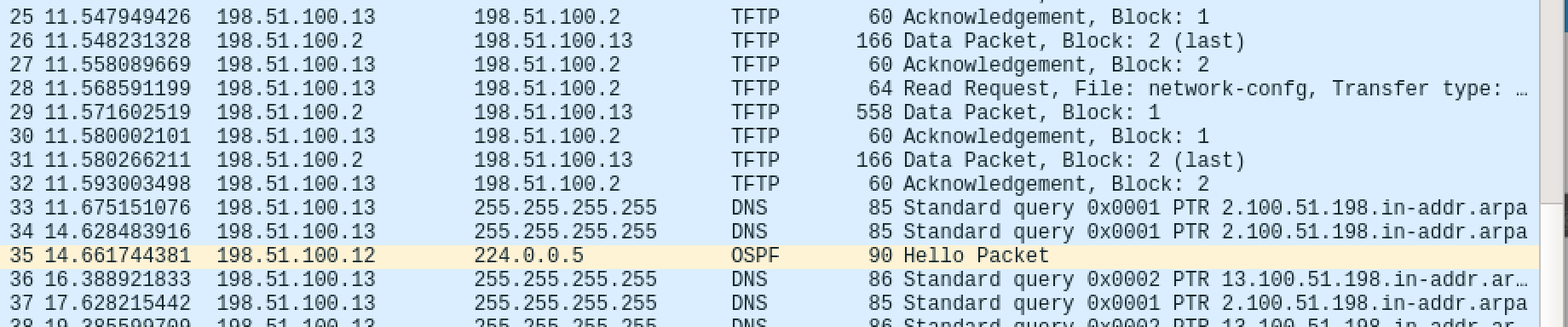




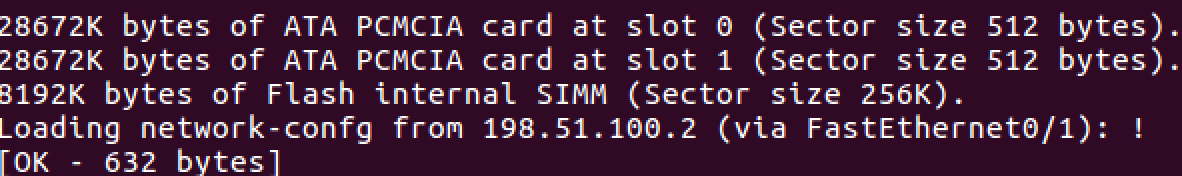
Next, do R6:

Make a file called

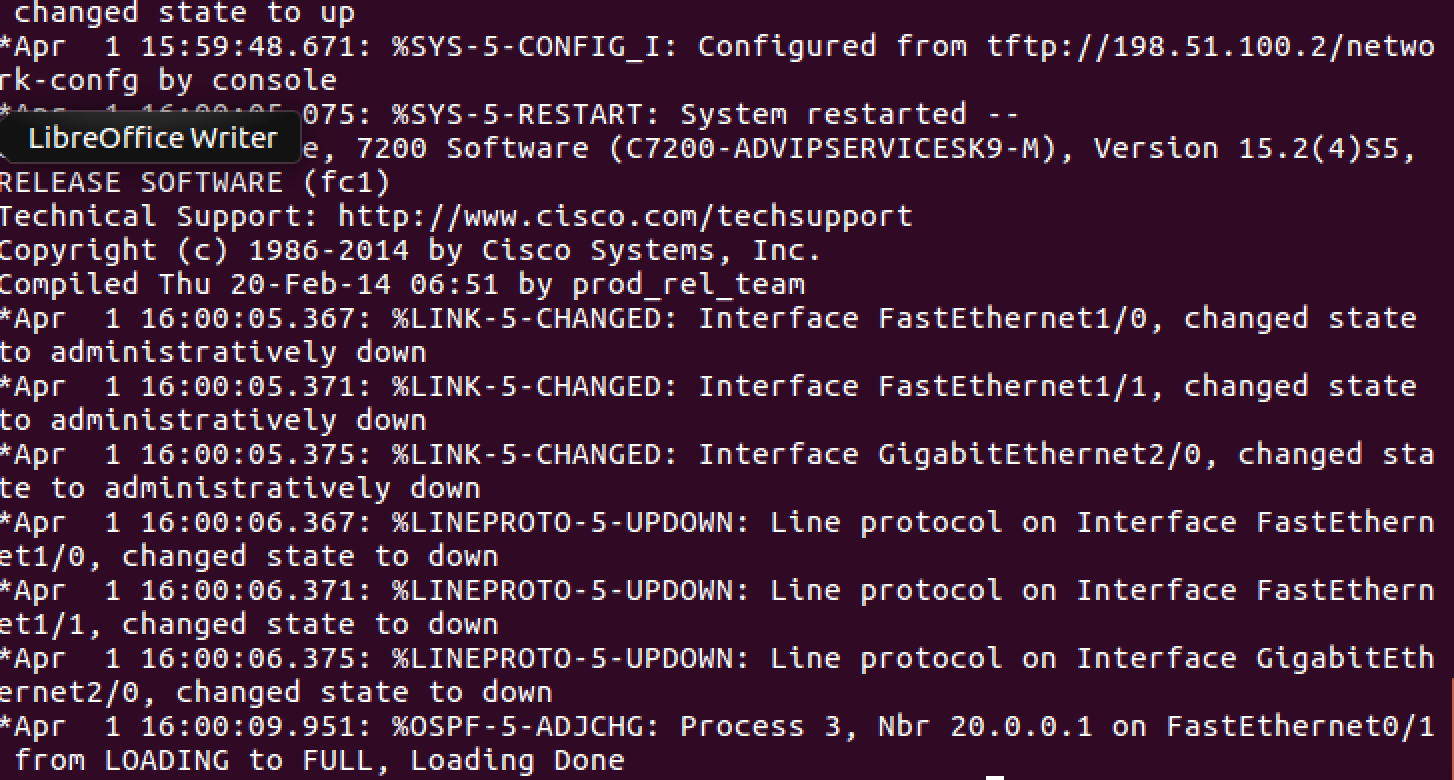
Wireshark on R6:



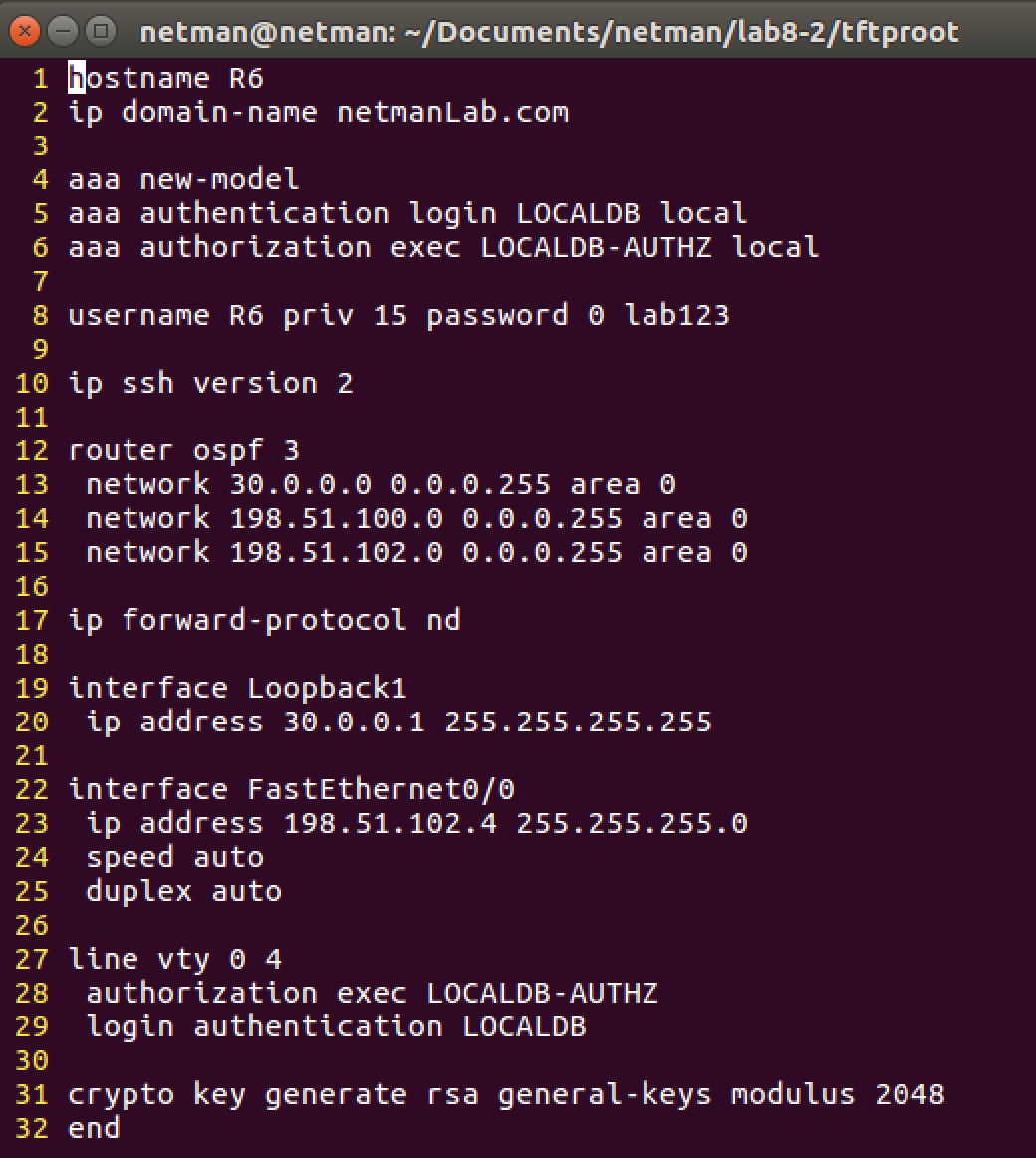
Console on R6:



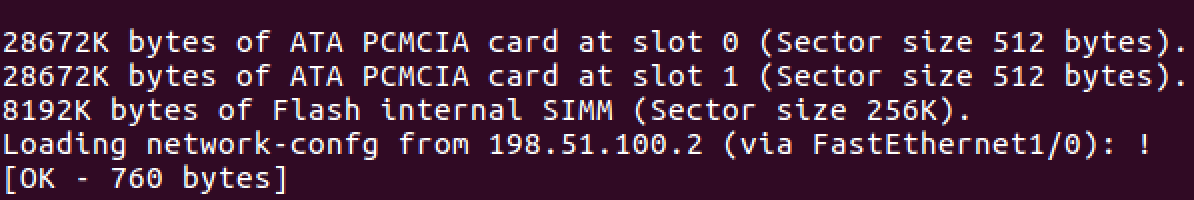
Form Neightship for OSPF:



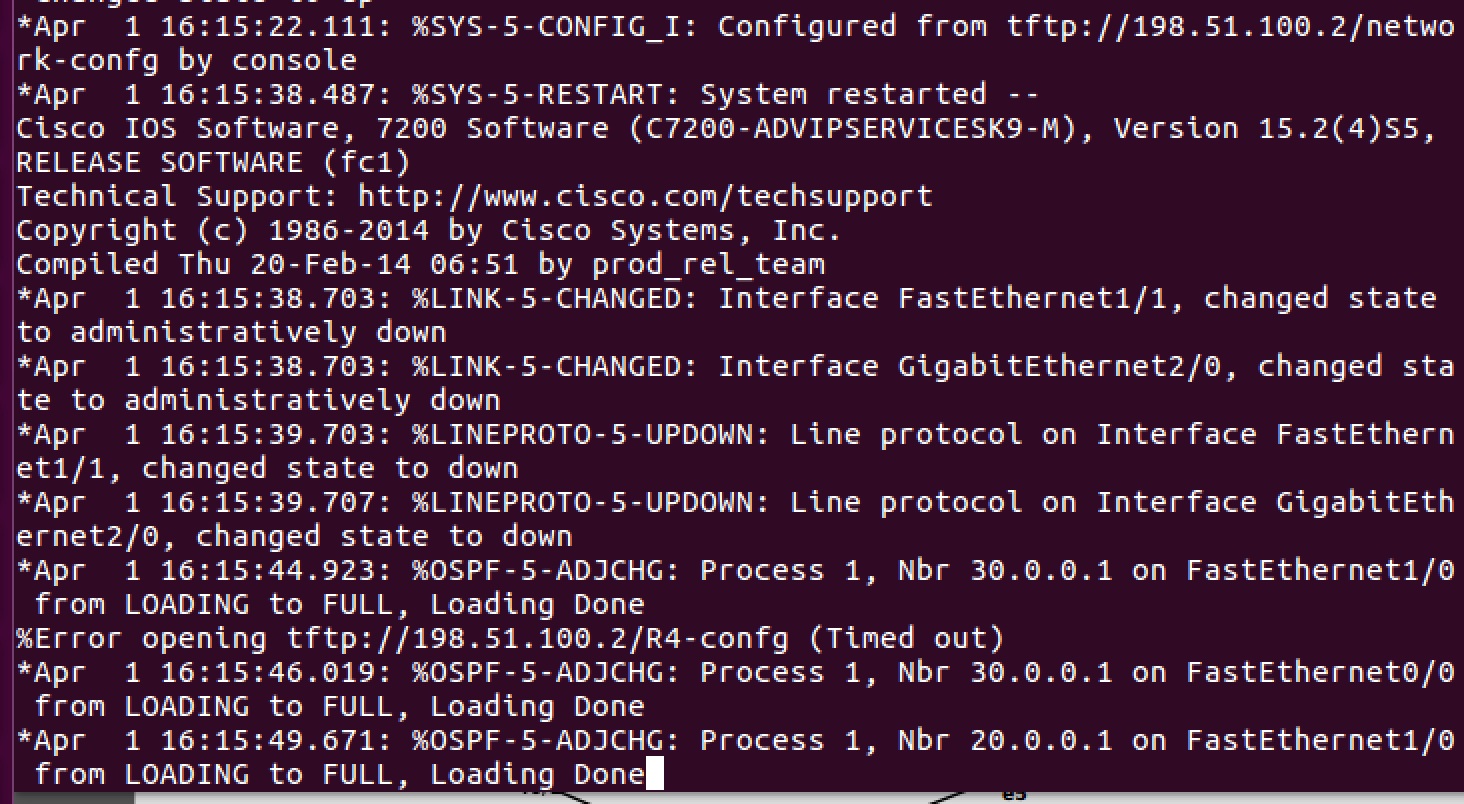
R6 of network-confg file:

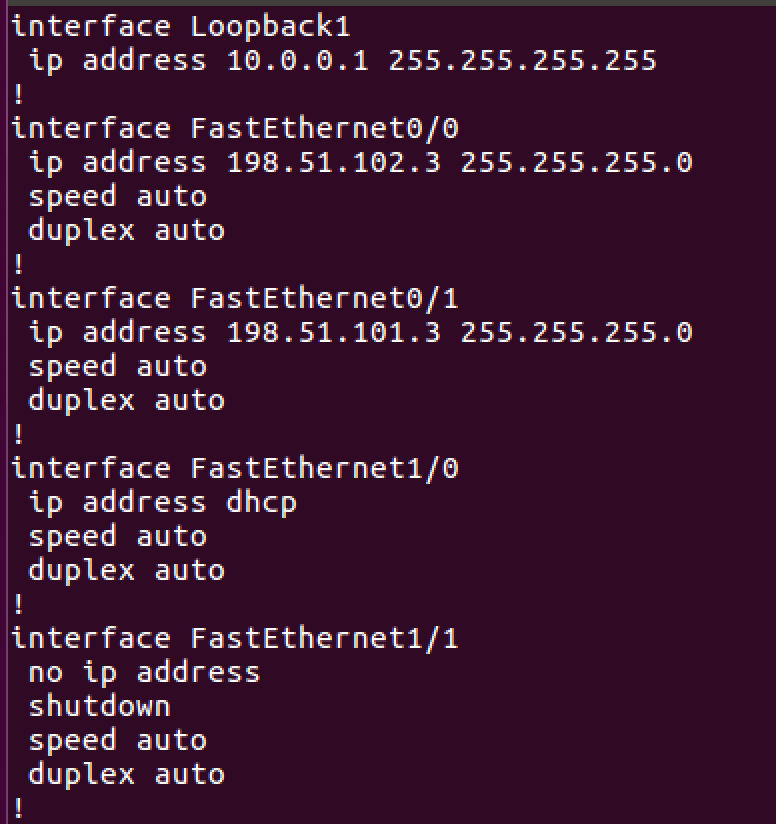


On R4 console:



For neighborship of OSPF:



****

**Use below links for ZTP reference,**

1. https://blog.digitalocean.com/zero-touch-provisioning-how-to-build-a-network-without-touching-anything/
2. https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst9500/software/release/16-5/configuration\_guide/prog/b\_165\_prog\_9500\_cg/zero-touch\_provisioning.pdf

Report Questions:

1. After completing this lab, explain ZTP and when can ZTP be implemented in the real world with an example? **[10 points]**

Essentially, ZTP is a switch feature that allows the devices to be provisioned and configure automatically, avoiding most of the manual labor involved with adding them to a network. [1]

When the new switch arrives, it has an OS to help bootstrap the device. It is removed from the box and goes to staging area, then an initial configuration is made to establish basic network connectivity. Once the initial OS and configuration has been verified, the device can be installed into the environment (racked and cabled), where further customized configuration can be made that is specific to the application and location within the network. [1]

Total Points \_\_\_\_\_\_\_\_\_\_\_\_ / 160 (+25 extra credit)

References

1. <https://www.networkworld.com/article/2876096/zero-touch-provisioning-can-help-the-network-world-catch-up-to-server-advances.html>