CSCI 5020:

Fundamentals of Network Programming

Fundamentals of Network Automation

University of Colorado Boulder

Department of Computer Science

Network Engineering

Professor Levi Perigo, Ph.D.

# Summary

Organizations today look for network automation capabilities in their hardware and software. In its simplest form, network automation is aimed to reduce manual effort in routine tasks and automate what used to be typed by network administrators into command line interfaces (CLIs).  Network automation can start with scripting and progress to intelligent network control, and ultimately allow efficient translation and deployment of network plans and policies.

# Objectives

* Learn how to take user input and remotely access and configure network devices with the help of Python automation libraries.

# 

# 

# Problem Statement:

You are a Sales Engineer assigned with the task of configuring a hundred new devices in your organization’s data center network. The operating expense required for manually configuring a network of this scale will be upwards of 300 hours (if you are being optimistic). Your manager has asked you to come up with a more efficient, less time-consuming approach. You have a week to demonstrate your method on a smaller topology, as a proof of concept (POC).

For the given topology (**Figure 1**), write a Python script that takes input from the user and logs into R1 and configures an IP address on the Fa1/0 interface and creates and configures an IP address on a loopback interface.

The input from the user should be the following:

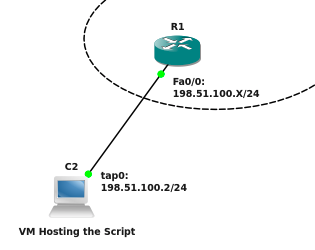
* IP address of the router you want to configure
* SSH login Username
* SSH login Password
* Interface description and IP address (that you want to configure on the router—fa1/0 in our example)
* Loopback interface description and IP address
* Subsequently, the script should also display the output of the “show ip interface brief” command and save the configuration to a file when all tasks are completed. 

Figure 1

Guidelines:

Here are some recommended configuration and setup steps, but you are free to deviate as long as you meet the objectives in the lab:

1. Download the NetMan VM from the link provided on Canvas and follow the instructions for setting up the VM in VMware Workstation (posted on Canvas).
   1. Again, this step is not necessary, you can use any environment you want, but this may save some “setup” time for GNS3, Python, etc.
2. Create the above topology in GNS3 on the NetMan VM.
3. **Steps to create a new file in Ubuntu VM (if needed):** To create a new file, open up the terminal app from the dock (bottom) in NetMan VM and type “**vi filename.py**”. Once Editor opens up, Hit “i” key to start writing. To save the file, Hit “ESC” and “:wq!”. Change the created file permissions to make it executable by using the following command:

**#chmod 777 filename.py**

To execute your script type:

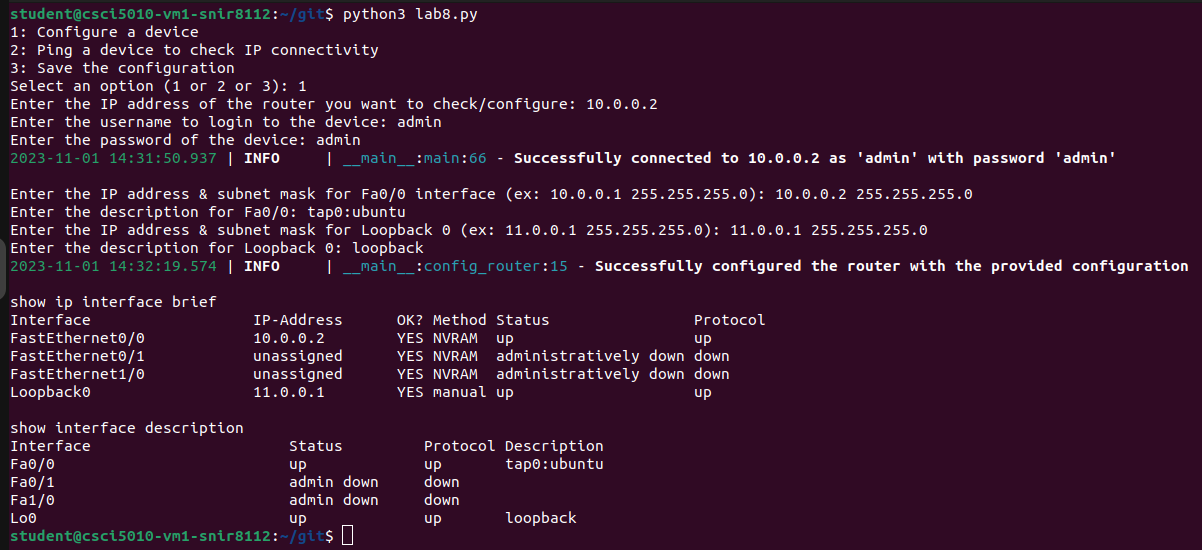
**#python filename.py**

1. Manually configure the Fa0/0 interface IP on the router R1 (make sure the IP you configure is in the same subnet as tap0 interface on Host). Follow the IP addressing scheme as mentioned in the above topology. Try to ping Fa0/0 IP from Host terminal. The pings should be successful.
2. Before writing the code, ensure that SSHv2 is enabled on the router R1. Refer: <https://www.pluralsight.com/blog/tutorials/configure-secure-shell-ssh-on-cisco-router>
   1. Note: you may have to re-configure SSH setting each time you close your GNS3 application
3. It is recommended that you use the netmiko library in Python for SSH. However, feel free to think out of the box and use any library that meets the objectives.

Code Requirements and Deliverables:

**NOTE**: Your **script** should fulfill the below requirements. Paste relevant supporting screenshots to this document and upload this document and your Python program to the course site.

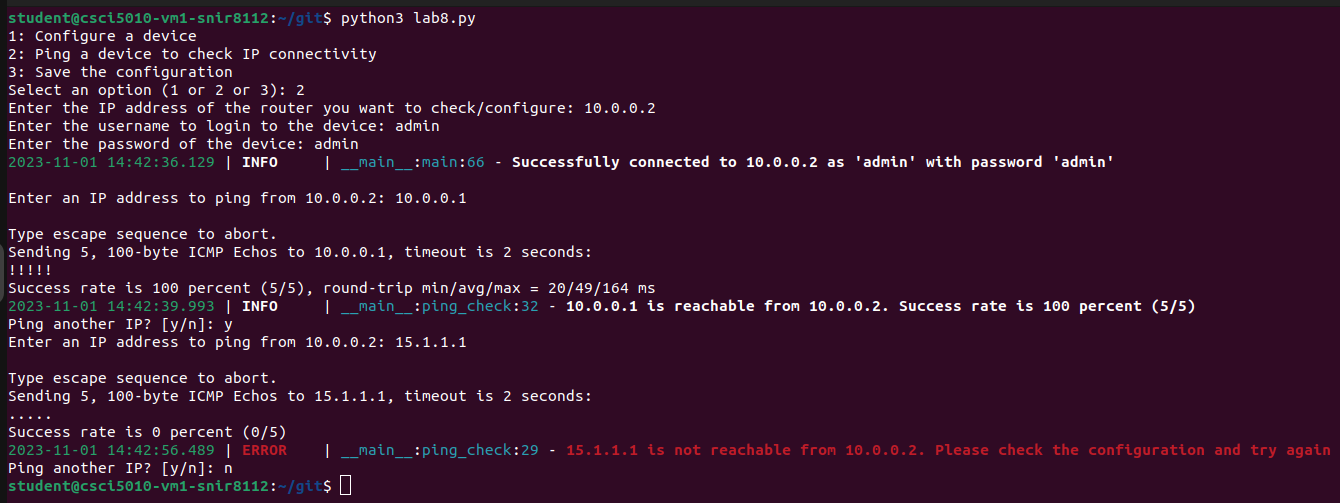
* The program should provide the user with a choice of options: 1) Configure a device; 2) ping a device to check for IP connectivity; 3) save the configuration.
* The script should successfully take user input (attach screenshots of the following objectives) [**30 Points**]:
* Option 1
  + IP address of the router you want to configure
  + SSH login Username
  + SSH login Password
  + Interface description and IP address (that you want to configure on the router—fa1/0 in our example)
  + Interface description and IP address that you want to configure for Loopback 1



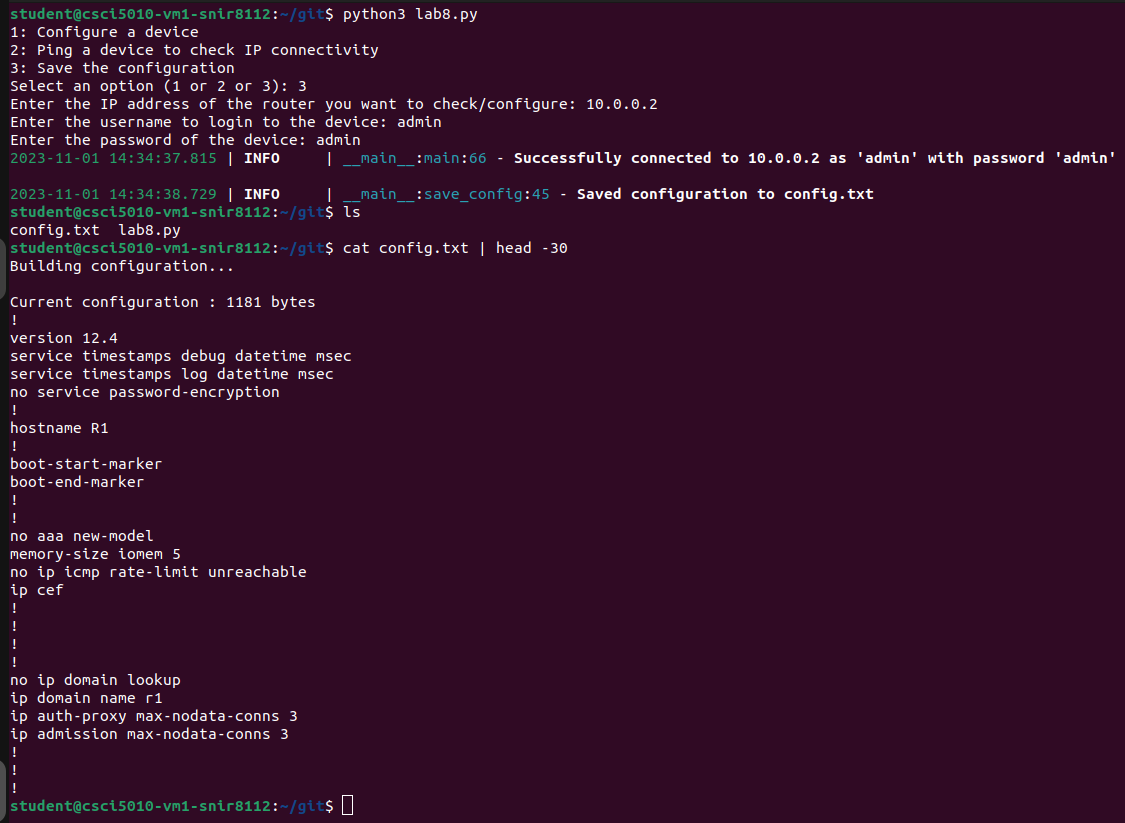
* Option 2
  + IP address to ping
  + Indication of successful/failure

I have 2 test cases here. The first one is where I entered 10.0.0.1 as the IP address to ping from the router (*10.0.0.1 is the tap0 interface on my machine*) and the ping worked.

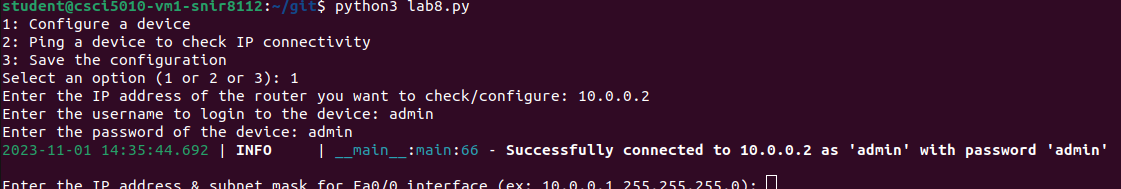
Next, I tried pinging 15.1.1.1 (which does not exist/not configured) and the ping errored (expected).



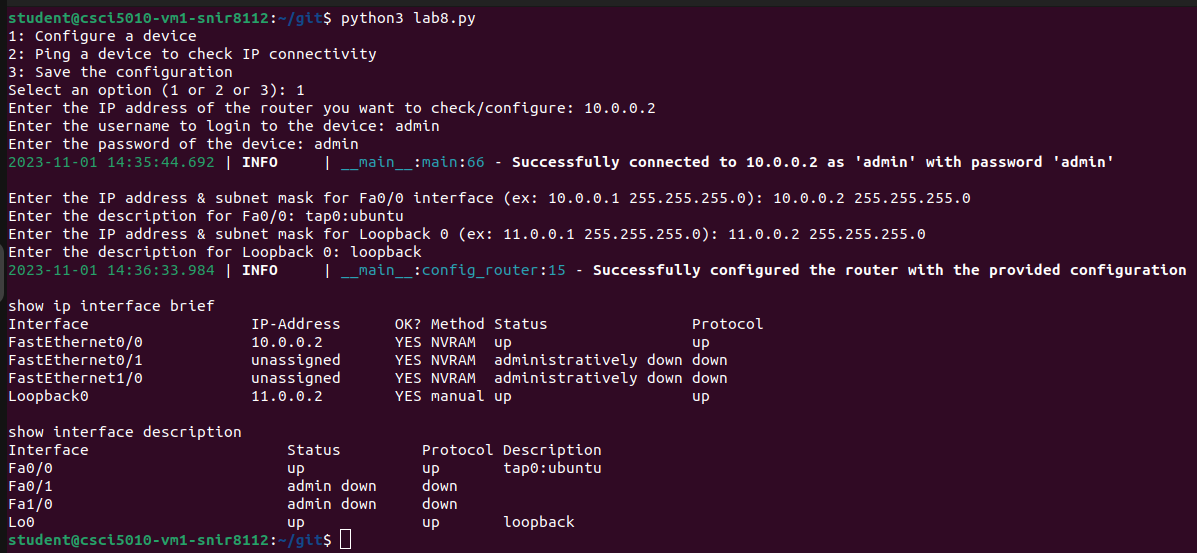
* Option 3
  + Indicate the startup configuration has been saved to a file



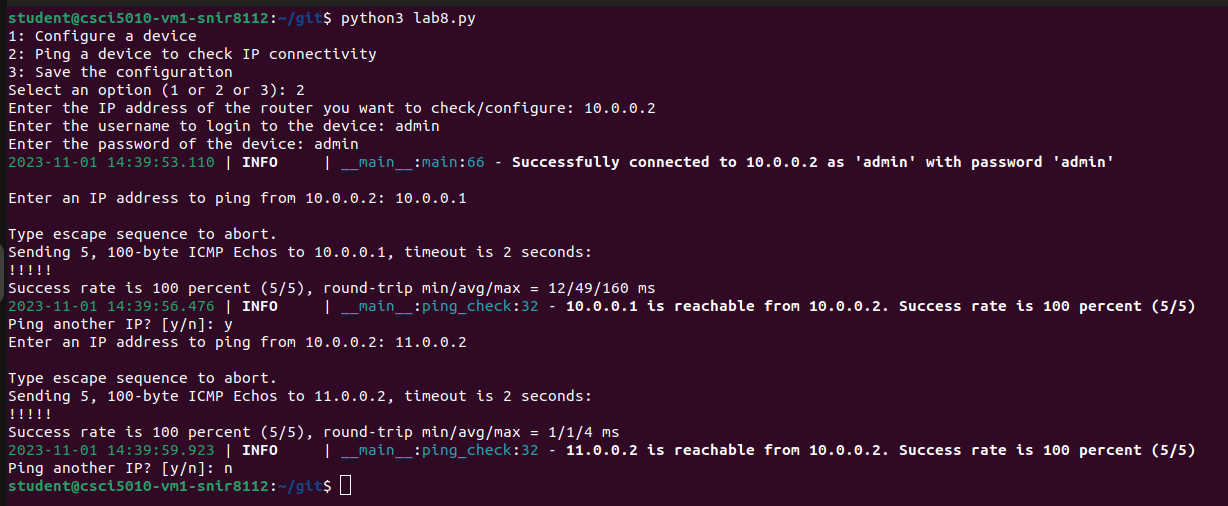
* Create print statements that indicate successful SSH into the router using Netmiko. The print statements should show the UN/PW, and IP address that was successfully logged into via your script as well as the IP address information configured (paste screenshot). [**15 Points**]



* Screenshot of the **script output** that displays the output of the “show ip interface brief” command demonstrating successful configuration of IP address on Fa1/0 and Loopback 1. **[20 Points]**

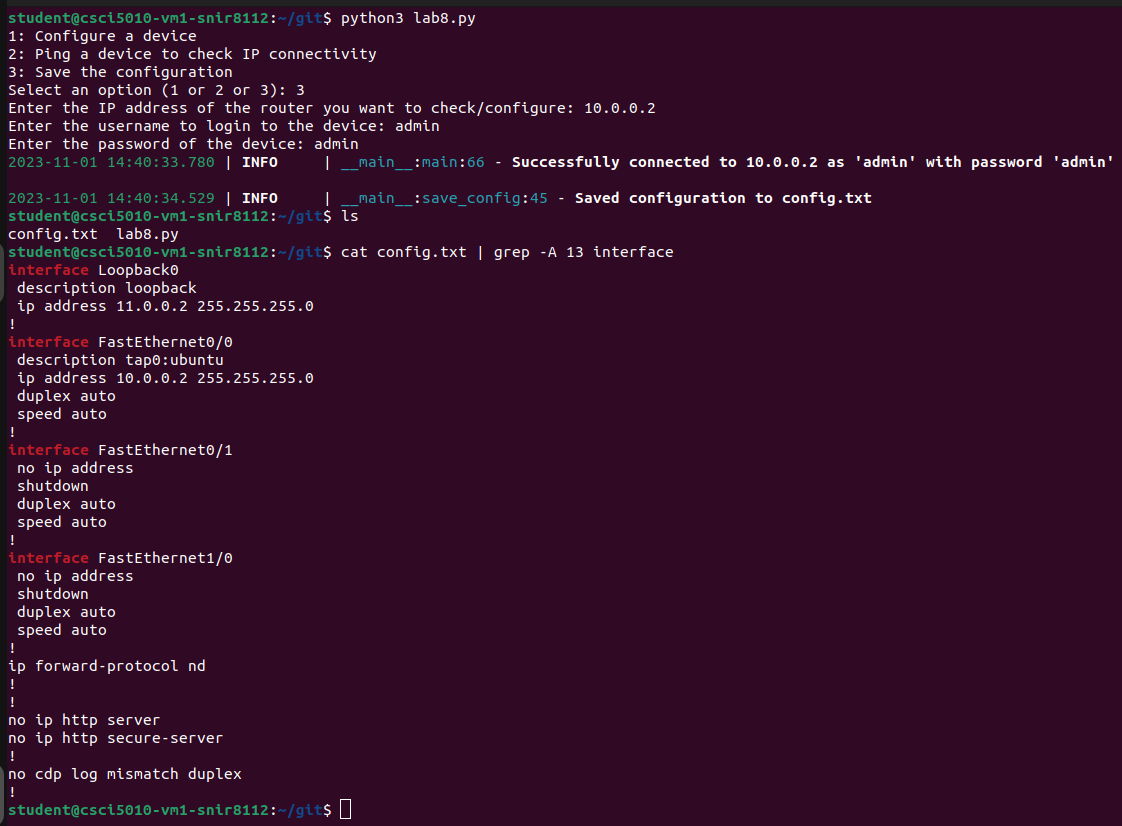
****

* Create print statements that indicate successful ping on the router. Provide a screenshot showing your code can successfully ping all the addresses on the router. [**10 points]**

****

* Screenshot of the startup configuration file that was saved (including the interfaces configured) **[10 points]**

**Note**: just grep-ed the interface config part of the file, but in reality, the file saves the entire running config of the device in config.txt file.

****

Questions:

1. Why would it be beneficial to take the time to create this application, when it would be faster to configure these settings manual? [**5 Points**]

The real advantage of automation becomes evident when you need to configure multiple devices with the same settings. Manual configurations can be error-prone and time-consuming when dealing with numerous devices. With automation, you can ensure consistency and save a significant amount of time when applying the same configurations across multiple devices.

1. Based on the description of this lab, what would you recommend as next steps to your manager for the proof of concept to make this more scalable in the future? [**10 Points**]

I would suggest the following:

Configuration management tool like Ansible to manage configurations across a larger number of devices. This will provide more robust scaling capabilities, version control, and auditing.

Create thorough documentation for the script, including usage instructions, dependencies, and any necessary setup steps.

Implement monitoring and alerting mechanisms to detect and respond to issues in real-time.

Implement mechanisms for backing up configurations before making changes and for restoring configurations in case of failures. This could be a version control system like github where all the configuration is stored.

Total Points \_\_\_\_\_\_\_\_\_\_\_\_ / 100