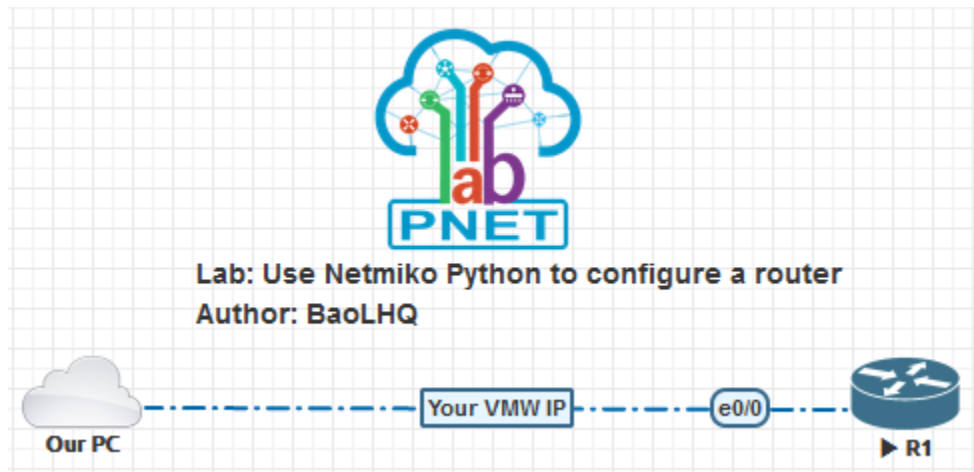


Use Netmiko Python to configure a router

Lab Topology:

The lab network topology is illustrated below:



<https://user.pnetlab.com/store/labs/detail?id=16043760603977>

Lab Objective:

The objective of this lab exercise is for you to learn and understand Netmiko Python.

Task:

We have a router that had ssh configuration. And we will use Netmiko Python to collect some information and configure this router.

Solution

Basic Configuration:

Router	R1
Configuration	<pre>hostname R1 ! ip domain name example.netacad.com crypto key generate rsa modulus 2048 ! username cisco privilege 15 password 0 cisco123! ! interface Ethernet0/0 no shutdown ip address 192.168.149.100 255.255.255.0 !</pre>

	line vty 0 4 exec-timeout 0 0 logging synchronous login local transport input ssh
--	---

You can see that 192.168.149.0/24 is my VMW IP. So you can change it.

My router have ip that is 192.168.149.100/24. Our ssh account: cisco/ cisco123!

First, you need to download netmiko module. You can follow this link:

<https://pypi.org/project/netmiko/>

Next, I am using Sublimetext Software for Python. You can download from this link:

<https://www.sublimetext.com/3>

After you have done, let's start:

Step1: Import Netmiko's ConnectHandler() Function.

```
from netmiko import ConnectHandler
```

Step2: The netmiko module includes the ConnectHandler() function. This function requires the following parameters for establishing an SSH connection with the IOS device:

- device_type - identifies the remote device type
- host - the address (host or IP) of the remote device
- username - remote ssh username
- password - remote ssh password

```
a={  
    'device_type':'cisco_ios',  
    'ip':'192.168.149.100',  
    'username':'cisco',  
    'password':'cisco123!',
```

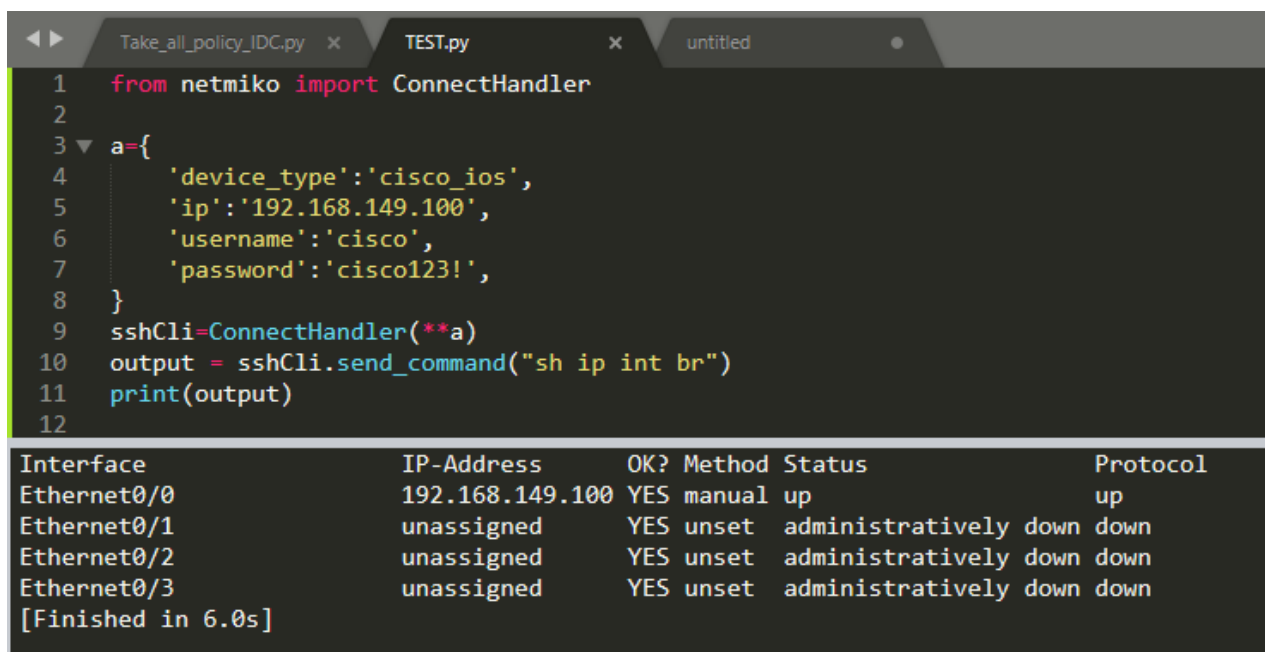
```
}  
sshCli=ConnectHandler(**a)
```

Step3: Use Netmiko to collect some information

We will try to collect information from command “ show ip int br”

```
output = sshCli.send_command("sh ip int br")  
print(output)
```

Press Ctrl + B in order to run the program, Here is our result:



```
1 from netmiko import ConnectHandler  
2  
3 a={  
4     'device_type':'cisco_ios',  
5     'ip':'192.168.149.100',  
6     'username':'cisco',  
7     'password':'cisco123!',  
8 }  
9 sshCli=ConnectHandler(**a)  
10 output = sshCli.send_command("sh ip int br")  
11 print(output)  
12
```

Interface	IP-Address	OK?	Method	Status	Protocol
Ethernet0/0	192.168.149.100	YES	manual	up	up
Ethernet0/1	unassigned	YES	unset	administratively down	down
Ethernet0/2	unassigned	YES	unset	administratively down	down
Ethernet0/3	unassigned	YES	unset	administratively down	down

[Finished in 6.0s]

Set a variable to hold the output of the show command you are sending. Use the send_command function of the sshCli object, which is the SSH session previously established, to send the desired command. In this case, we are sending sh ip int br.

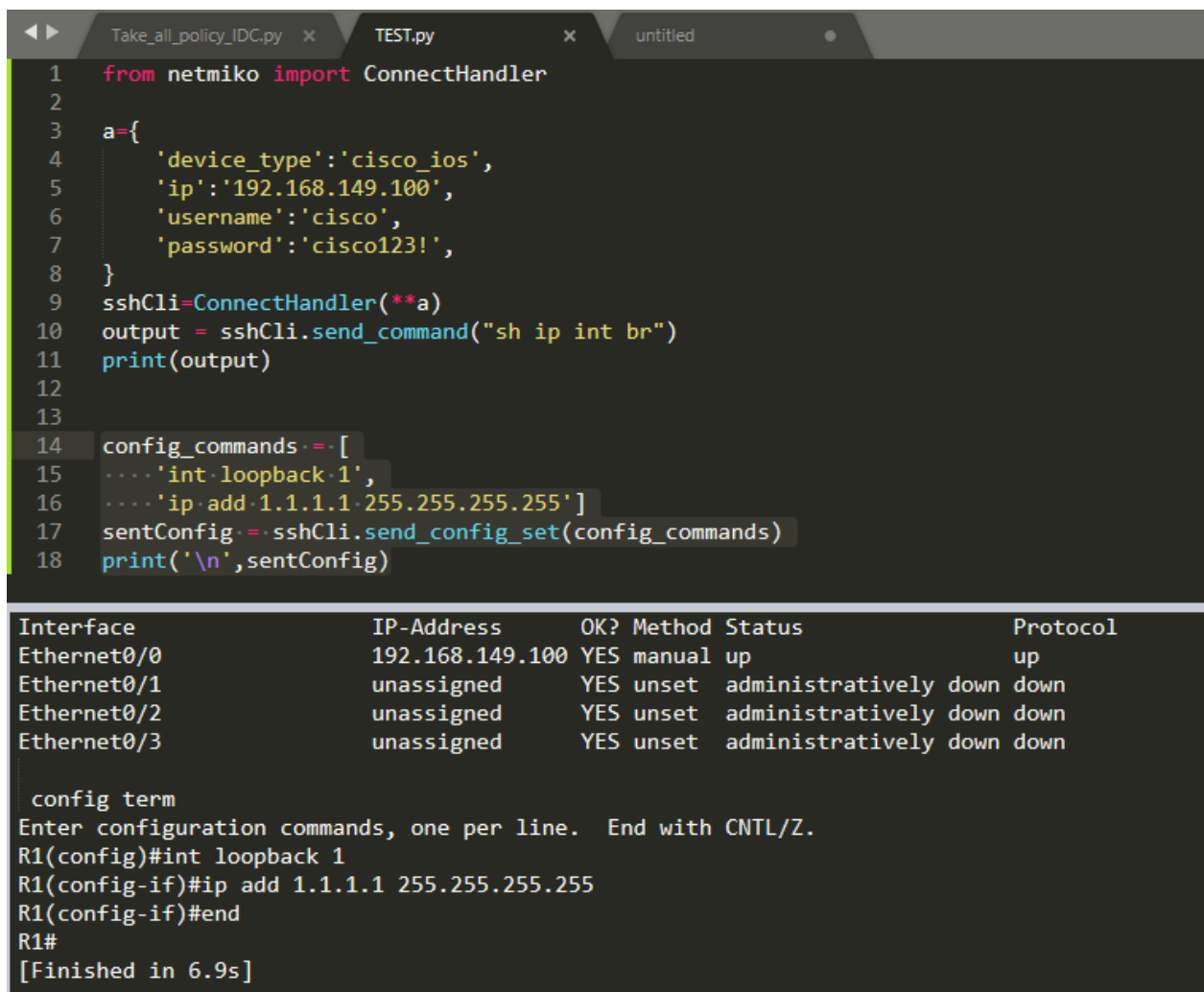
After sent the command, we would print the result like above.

Step4: Use Netmiko configure a router

Now, we are configuring the Loopback 1 interface on R1:

```
config_commands = [
    'int loopback 1',
    'ip add 1.1.1.1 255.255.255.255']
sentConfig = sshCli.send_config_set(config_commands)
print('\n',sentConfig)
```

And this is the result:



```
1 from netmiko import ConnectHandler
2
3 a={
4     'device_type':'cisco_ios',
5     'ip':'192.168.149.100',
6     'username':'cisco',
7     'password':'cisco123!',
8 }
9 sshCli=ConnectHandler(**a)
10 output = sshCli.send_command("sh ip int br")
11 print(output)
12
13
14 config_commands = [
15     'int loopback 1',
16     'ip add 1.1.1.1 255.255.255.255']
17 sentConfig = sshCli.send_config_set(config_commands)
18 print('\n',sentConfig)
```

Interface	IP-Address	OK?	Method	Status	Protocol
Ethernet0/0	192.168.149.100	YES	manual	up	up
Ethernet0/1	unassigned	YES	unset	administratively down	down
Ethernet0/2	unassigned	YES	unset	administratively down	down
Ethernet0/3	unassigned	YES	unset	administratively down	down

```

config term
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#int loopback 1
R1(config-if)#ip add 1.1.1.1 255.255.255.255
R1(config-if)#end
R1#
[Finished in 6.9s]
```

Config_commands is a list command that we want to configure into R1. We use send_config_set function to enter the configure terminal mode and push all our list command into the router. Let's go into R1 check interface.

R1#show ip int br

Interface	IP-Address	OK?	Method	Status	Protocol
Ethernet0/0	192.168.149.100	YES	manual	up	up
Ethernet0/1	unassigned	YES	unset	administratively down	down



<i>Ethernet0/2</i>	<i>unassigned</i>	<i>YES unset administratively down down</i>
<i>Ethernet0/3</i>	<i>unassigned</i>	<i>YES unset administratively down down</i>
<i>Loopback1</i>	<i>1.1.1.1</i>	<i>YES manual up up</i>

It is working! Nice!

