Lab: Paramiko and Netmiko

Exercise: Paramiko and Netmiko

In this Exercise you will configure network devices with Paramiko and Netmiko. Both of these Python libraries work exactly the same with a network device as an SSH or Telnet session does. However, parsing is much less powerful than with XML or JSON. Be sure you have finished lab 9 before starting this lab.

Task 1: Configure IOS XE router with Paramiko

In this task you will use Paramiko to configure your CSR1000v. Paramiko demands SSH and sends any CLI commands to a device that the device supports.

How-to Steps

1. Starting from your login page on Remotelabs.com, click on **Connect via Topology**.

2. The lab topology diagram will open. Notice how you can hover your mouse over the individual computers to see which you can connect into directly form the web portal.

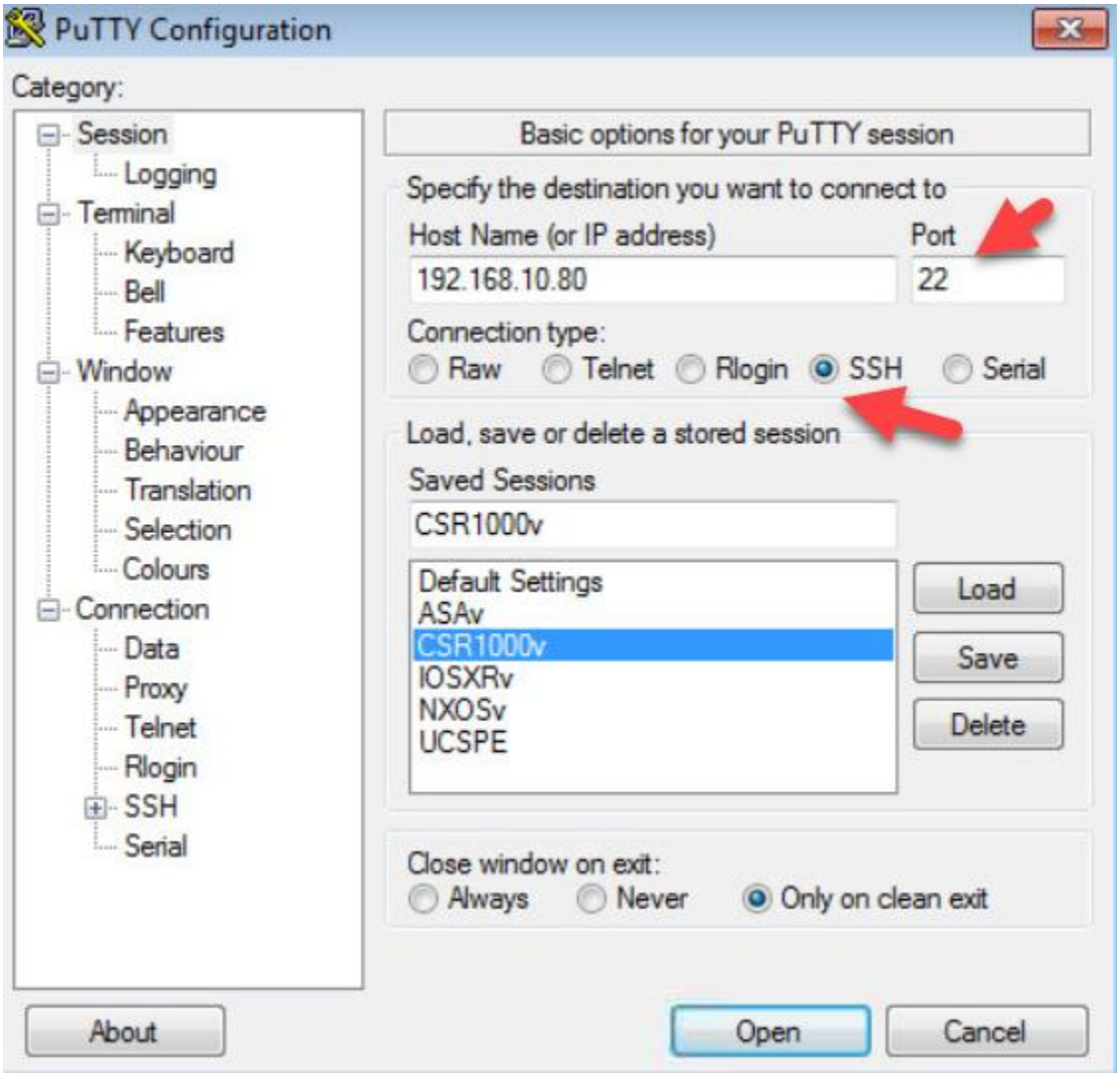
NOTE_CALLOUT
In this Global Knowledge lab environment, you can open any of the virtual machines with a "hot link," but you can have only one VM open at any one time. Opening a second VM will close the first connection you were in, without losing any of your settings or internal network connections.

3. Open **Win7**.
4. Open a Putty telnet session to your **CSR1000v** router.
5. Configure your router under **config t** to enable SSH:

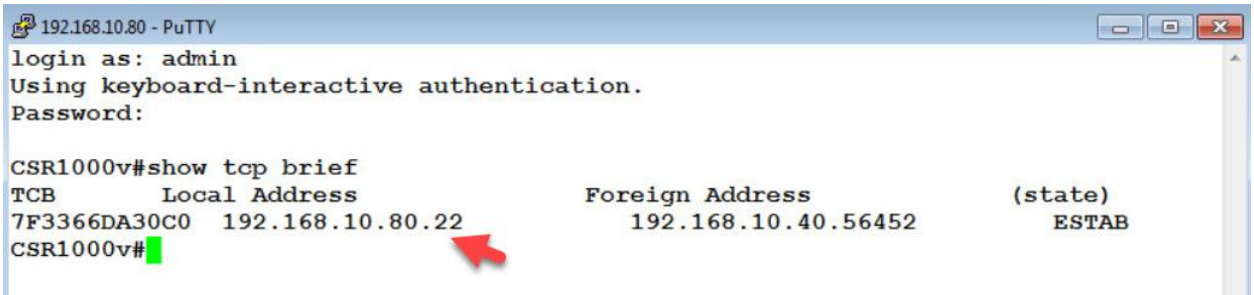
```
username admin privilege 15 password 0 cisco
```

```
ip ssh timeout 10
line vty 0 4
  login local
  transport input ssh
end
wr
```

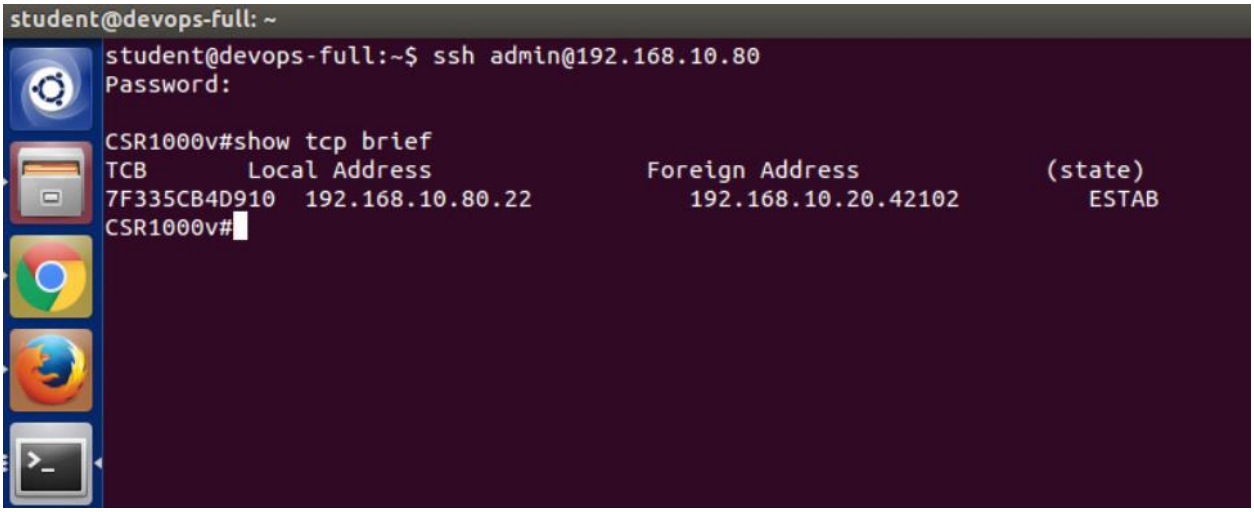
6. Verify that SSH is working from **Win7** by opening a new Putty session to the **CSR1000v**, but to **SSH** Port **22**.



7. Login with user: **admin** and password: **cisco** and verify that you can see your TCP port 22 SSH session. You may see other ports open depending on which prior labs you have done.

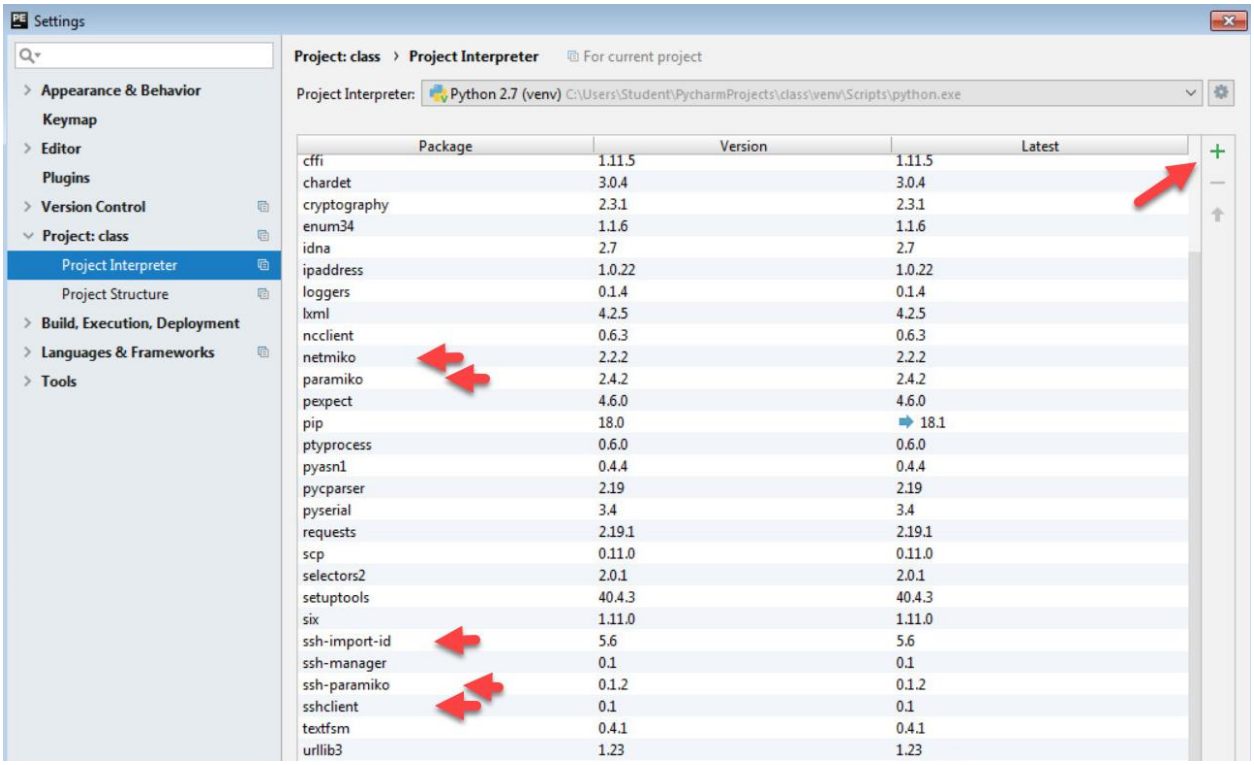


8. To further test, return to your **Ubuntu Configured** VM and from a **New Terminal** open an SSH session to your **CSR1000v** router with the same password **cisco**. Respond **yes** if you get an authenticity warning.

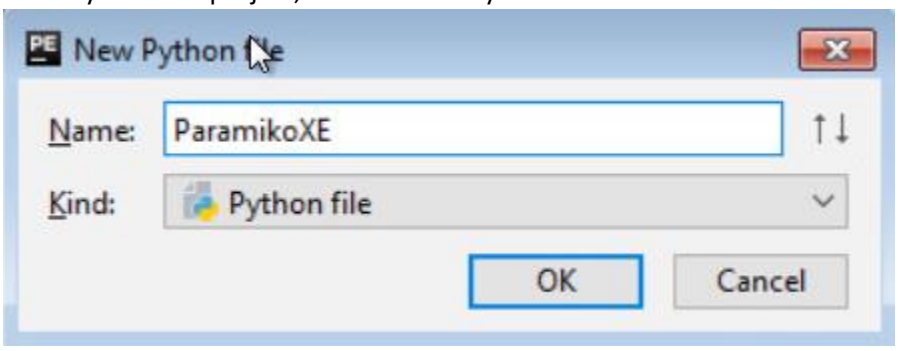


9. Navigate back to **Win7** and Open **PyCharm**.
10. Highlight your project **class** and select **File > Settings…**
11. Select the **Project Interpreter**. Ensure you have these packages. Install them as needed. If any installs, fail, just keep installing them until they load.

```
paramiko
netmiko
sshclient
ssh-paramiko
ssh-import-id
ssh-manager
```



12. Under your **class** project, create a new Python file called **ParamikoXE.**



13. Select **File > Open**. Select the file **Z:\Python\paramiko_example.py**
14. Copy the code into the **ParamikoXE.py** window that you previously created.
15. Study the code.

16. Ensure that this Python script has the correct SSH credentials to your IOS XE router:
    Host: **192.168.10.80**
    Port: **22**
    Username: **admin**
    Password: **cisco**

```
import time

def main():

    # Use Paramiko to create the connection
    pre_connection = paramiko.SSHClient()
    pre_connection.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    pre_connection.connect("192.168.10.80", port=22, username='admin', password='cisco', look_for_keys=False, allow_age
    connection = pre_connection.invoke_shell()
    output = connection.recv(65535)
    print output

    connection.send("show ip int brief\n")
    time.sleep(.5)
    output = connection.recv(65535)
```

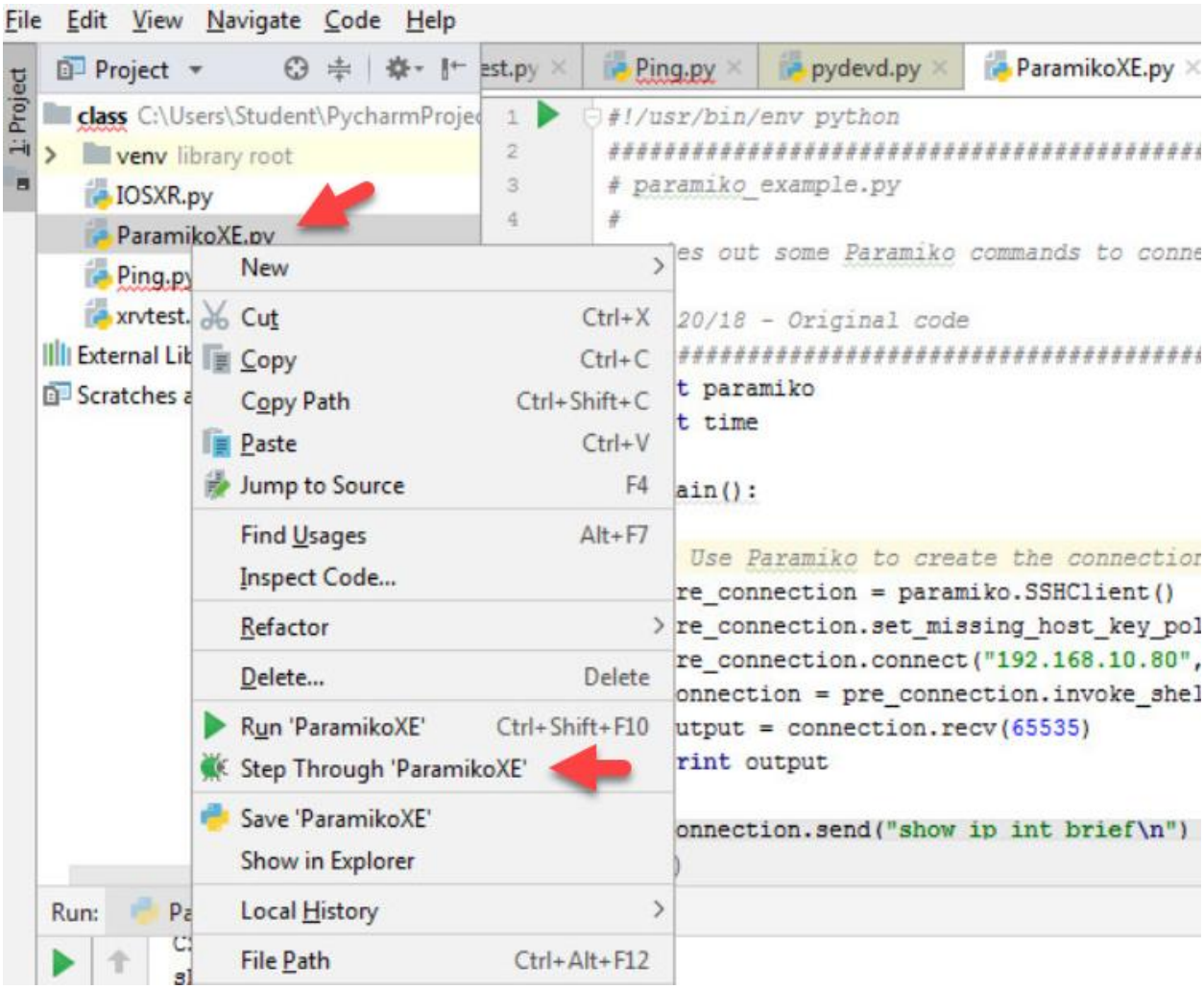
Note: Paramiko only uses SSH TCP port 22.

17. Note that this simple initial Python script only sends the commands in the **connection.send** python line:

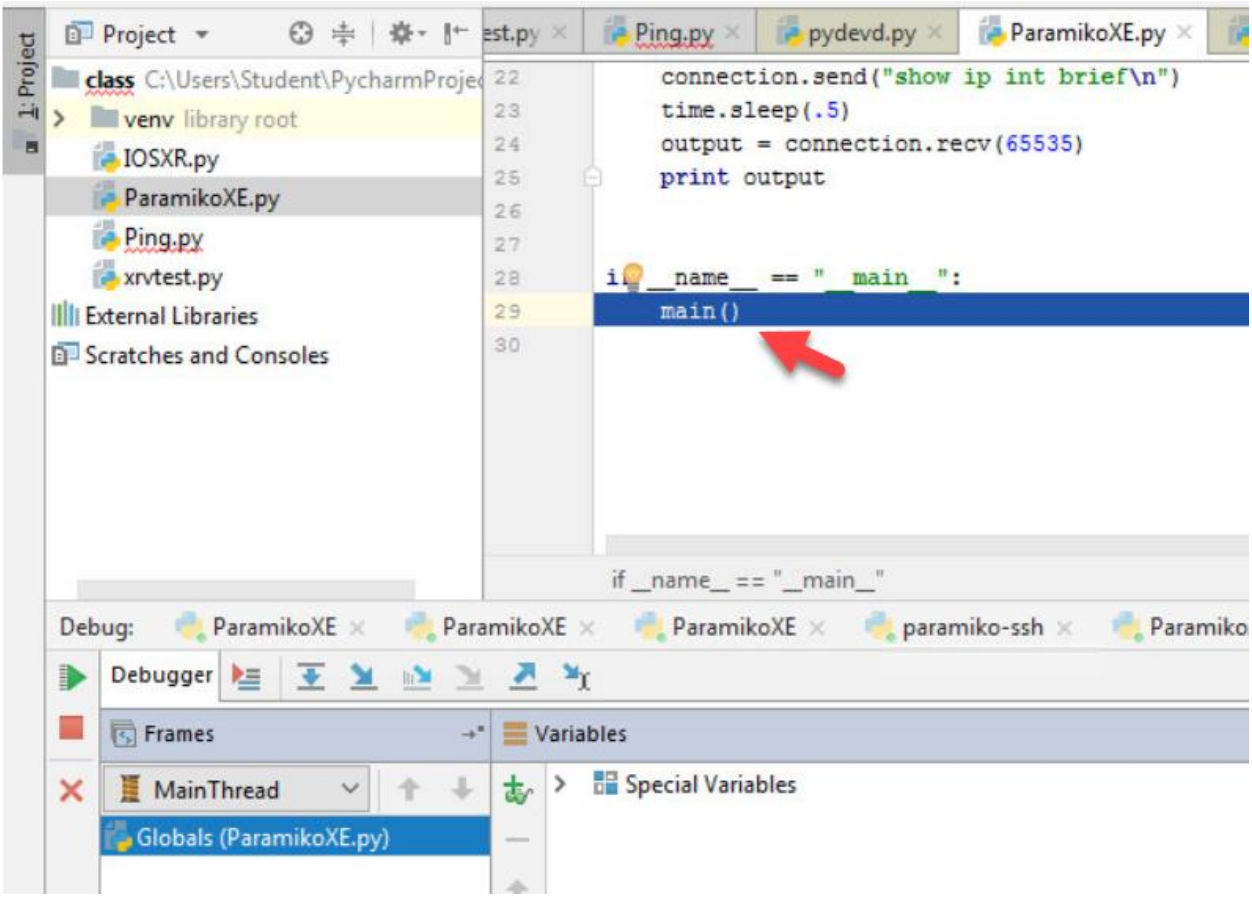
```
    # Use Paramiko to create the connection
    pre_connection = paramiko.SSHClient()
    pre_connection.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    pre_connection.connect("192.168.10.80", port=22, username='admin', password='cisco', look_for_keys=False, allow_ager
    connection = pre_connection.invoke_shell()
    output = connection.recv(65535)
    print output

    connection.send("show ip int brief\n")
    time.sleep(.5)
    output = connection.recv(65535)
    print output
```

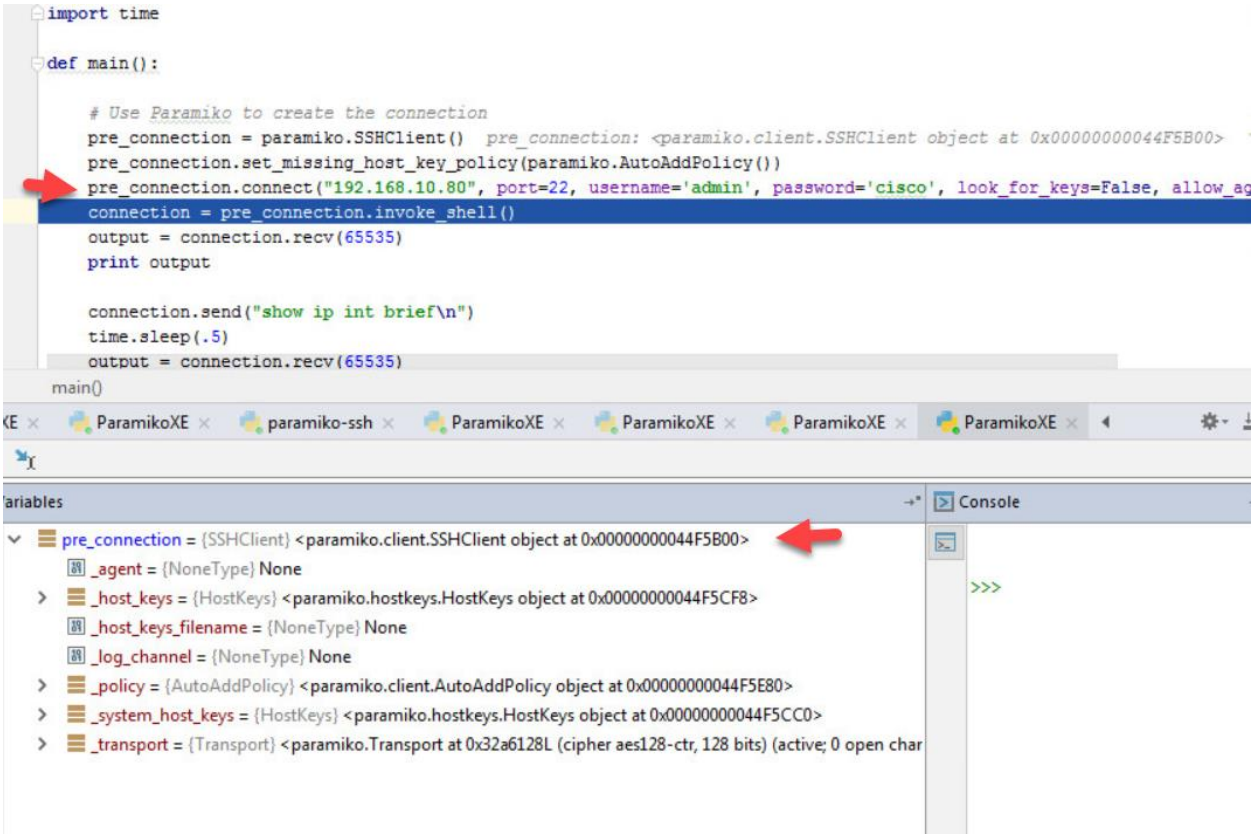
18. Highlight your script on the right and click Step Through **ParamikoXE**.



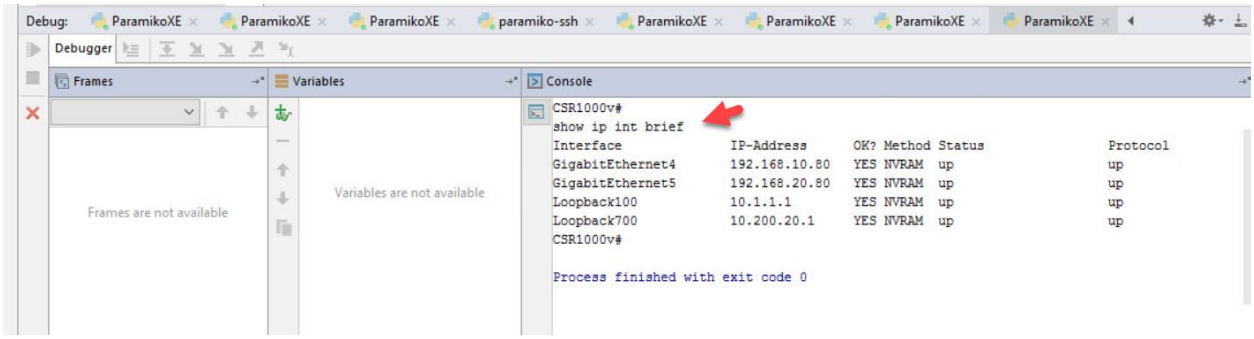
19. Step through your code and note that the function **main()** is initially "stepped over", but after you call the function **main()** after the entry point that the code steps through each line of the function.



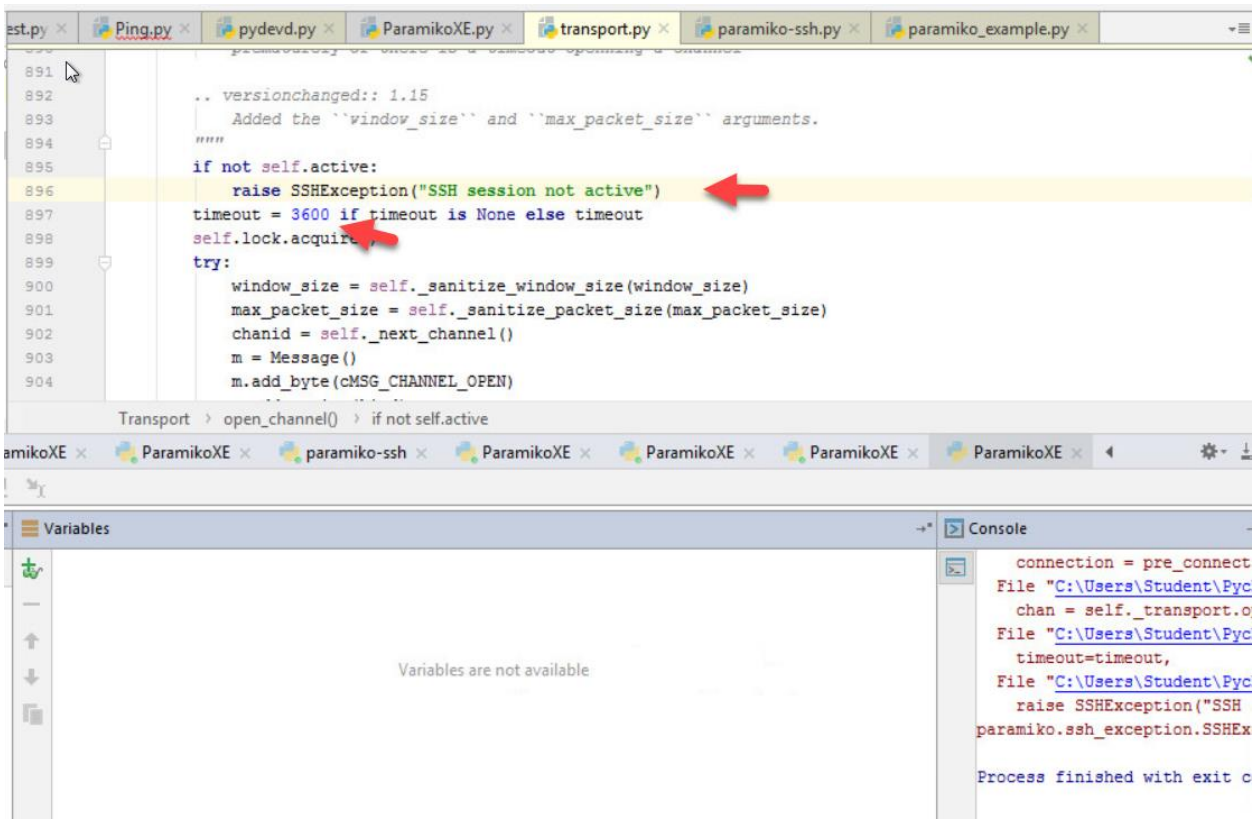
20. Verify that the connection of SSH succeeded after you step over it.



21. Continue to step all the way through the code and verify that your Python script displays the router interfaces in the PyCharm console.



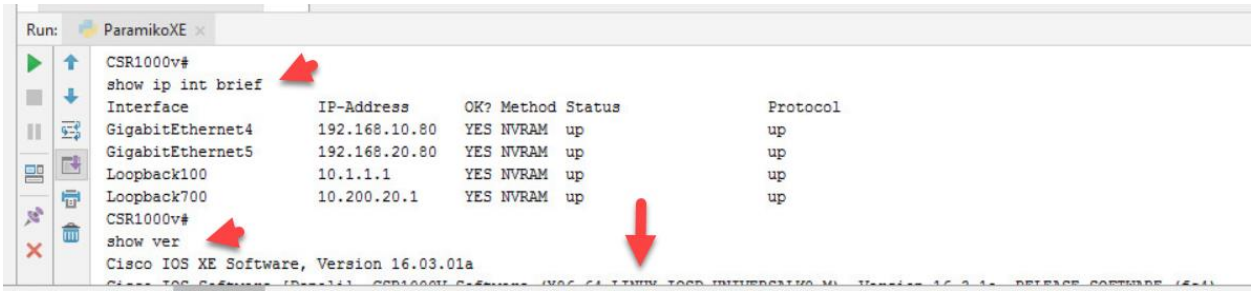
Note: If you are stepping through too slow, you may get this timeout error. If you do, simply run the script again and don't spend as much time stepping through the code.



22. Add another command **show ver** to your script with a new output variable.

Note: You can abbreviate the **show version** command as **show ver** exactly as you can in the IOS device. Paramiko is only sending the commands exactly as you give them.

23. Step through your updated script in full and verify you can see the output of both show commands in the PyCharm console. You may have to scroll down to see the entire **show version** output.



24. To demonstrate an example of an IOS configuration change with Paramiko, note that you currently do not have any EIGRP configuration.

```
CSR1000v#
CSR1000v#sh run | begin eigrp
CSR1000v#
```

25. Update your script to configure basic EIGRP one command at a time.

```
    # Use Paramiko to create the connection
    pre_connection = paramiko.SSHClient()
    pre_connection.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    pre_connection.connect("192.168.10.80", port=22, username='admin', password='cisco', look_for_keys=False, allow_agent=False)
    connection = pre_connection.invoke_shell()
    output = connection.recv(65535)
    print output

    connection.send("config t\n")
    time.sleep(.5)
    output = connection.recv(65535)
    print output
    connection.send("router eigrp 10 \n")
    time.sleep(.5)
    output1 = connection.recv(65535)
    print output1
    connection.send("network 10.0.0.0 \n")
    time.sleep(.5)
    output2 = connection.recv(65535)
    print output2
    connection.send("end\n")
    time.sleep(.5)
    output3 = connection.recv(65535)
    print output3
    connection.send("wr\n")
    time.sleep(.5)
    output4 = connection.recv(65535)
    print output4

if __name__ == "__main__":
    main()
```

main()

26. Step through your script in full. While all the **print** statements are not required for this router configuration change, note how they show each IOS command executed in the PyCharm console.

27. Verify these EIGRP commands appear in your router. This is just a demonstration. EIGRP is not used for routing at this point in this configuration.

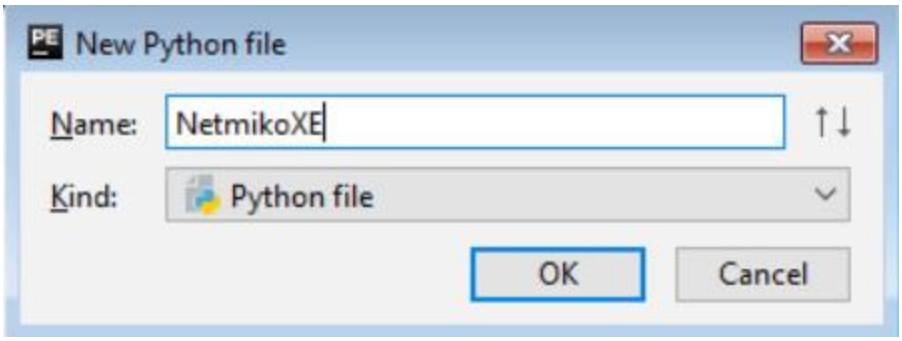
```
CSR1000v#sh run | begin eigrp
router eigrp 10
 network 10.0.0.0
!
```

Task 2: View the CSR1000v configuration with Netmiko

In this task you will use Netmiko. While Paramiko demands an SSH session, Netmiko can be used for older devices as it supports TCP ports 22 (SSH) and 23 (Telnet).

How-to Steps

1. In **Win7**, under your PyCharm **class** project, create a **new Python file** called **NetmikoXE.**



2. Select **File > Open**. Select the file **Z:\Python\netmiko_example.py**
3. Copy the code into the **NetmikoXE.py** window that you previously created.
4. Study the code.
5. Ensure that this Python script has the correct SSH credentials to your IOS XE router:
   Host: **192.168.10.80**
   Device_type: **cisco_ios**
   Username: **admin**
   Password: **cisco**
6. Modify the import command to correctly import **ConnectHandler**.

```
#
# 06/20/18 - Original code
###############################################################################
from netmiko import ConnectHandler

def main():
    # Use Netmiko to create the connection
    connection = ConnectHandler(device_type="cisco_ios", ip="192.168.10.80", username="admin", password="cisco")
    # Grab the running config
    output = connection.send_command("show run")
    # Print the current cooooonfig
    print ("Current running config: \n")


if __name__ == "__main__":
    main()
```

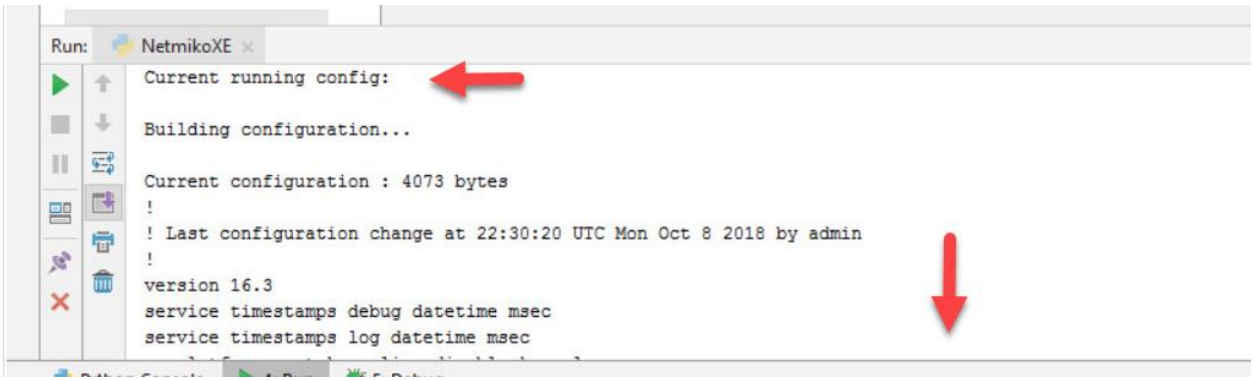
7. Add a command to print the output.

```
from netmiko import ConnectHandler

def main():
    # Use Netmiko to create the connection
    connection = ConnectHandler(device_type="cisco_ios", ip="192.168.10.80", username="admin", password="cisco")
    # Grab the running config
    output = connection.send_command("show run")
    # Print the current cooooonfig
    print ("Current running config: \n")
    print output


if __name__ == "__main__":
    main()
```

8. Step through your code and verify you can see the running configuration in the PyCharm **Console**.



Challenge results

This lab demonstrated the use of Netmiko and Paramiko to view show commands and perform configuration changes to a Cisco router.