Lab: Paramiko and Netmiko

Exercise: Paramiko and Netmiko

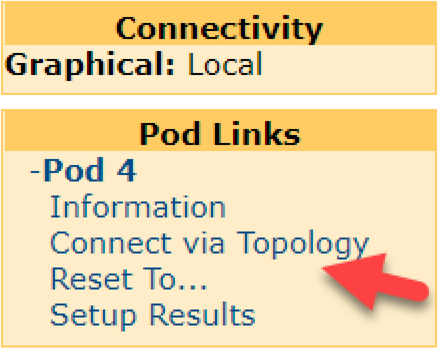
In this Exercise you will configure network devices with Paramiko and Netmiko. Both of these Python libraries work exactly the same with a network device as an SSH or Telnet session does. However, parsing is much less powerful than with XML or JSON.

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**.



Connectivity
Graphical: Local

Pod Links
-**Pod 4**
Information
Connect via Topology
Reset To...
Setup Results

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.

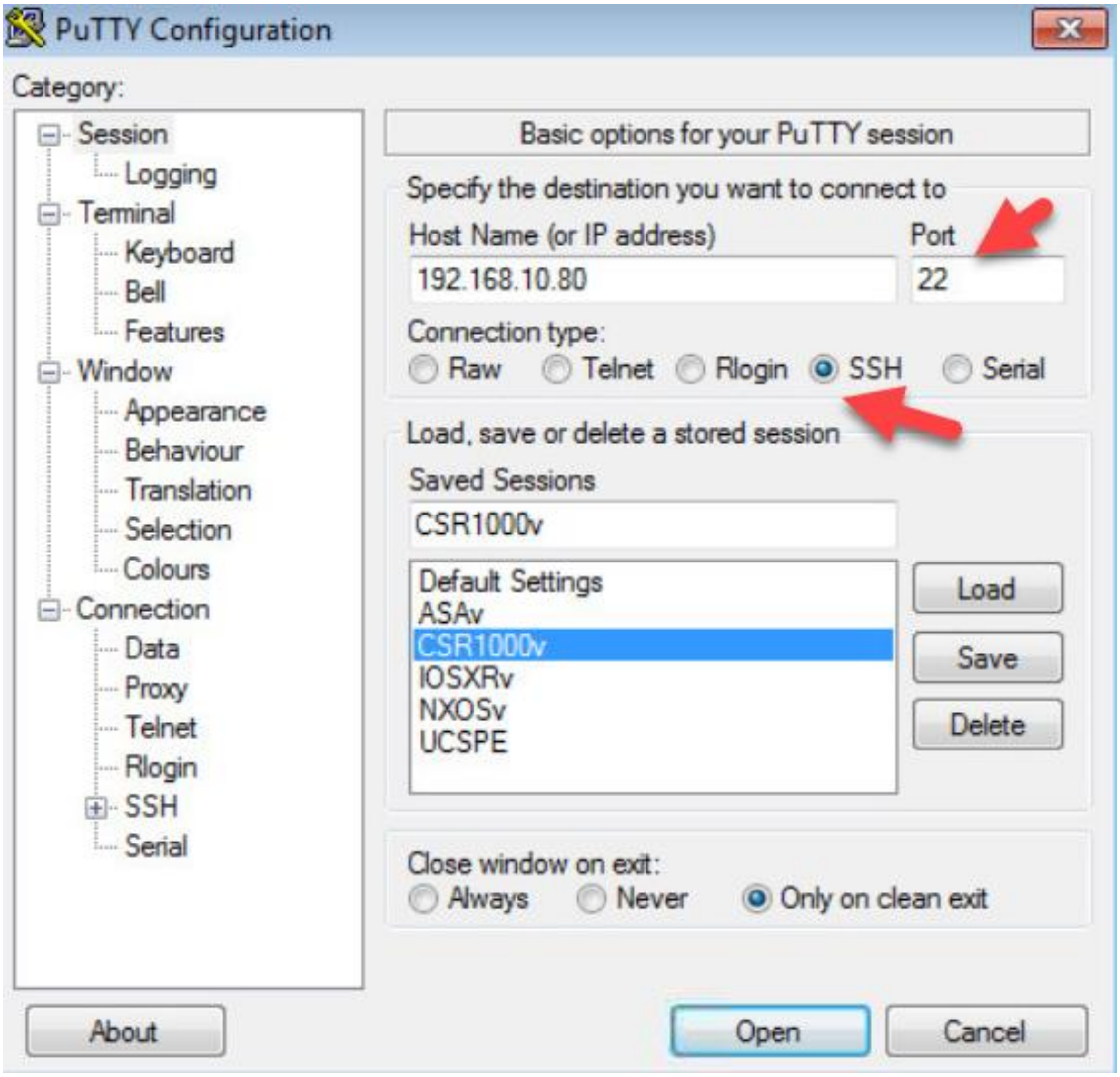
NOTE_CALLOUT
In this Global Knowledge lab environment, you can open any of the virtual machines with a “hot link,” but you can have only one VM open at any one time. Opening a second VM will close the first connection you were in, without losing any of your settings or internal network connections.

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**.



Specify the destination you want to connect to

Host Name (or IP address): 192.168.10.80 Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

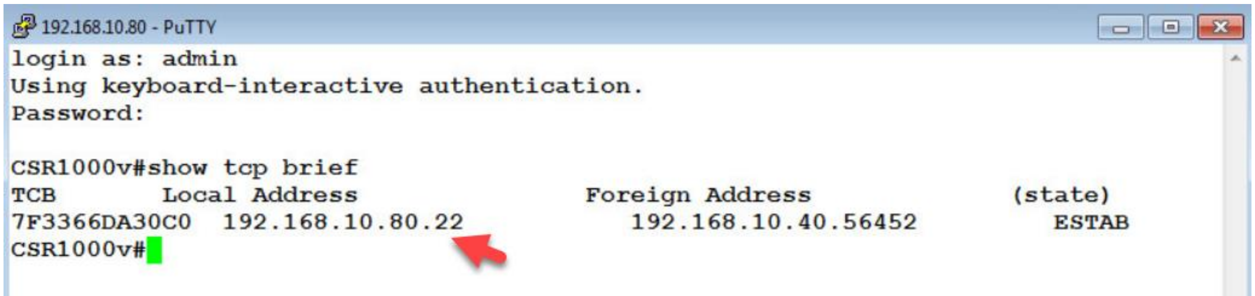
Load, save or delete a stored session

Saved Sessions: CSR1000v

Default Settings
ASAv
CSR1000v
IOSXRv
NXOSv
UCSPE

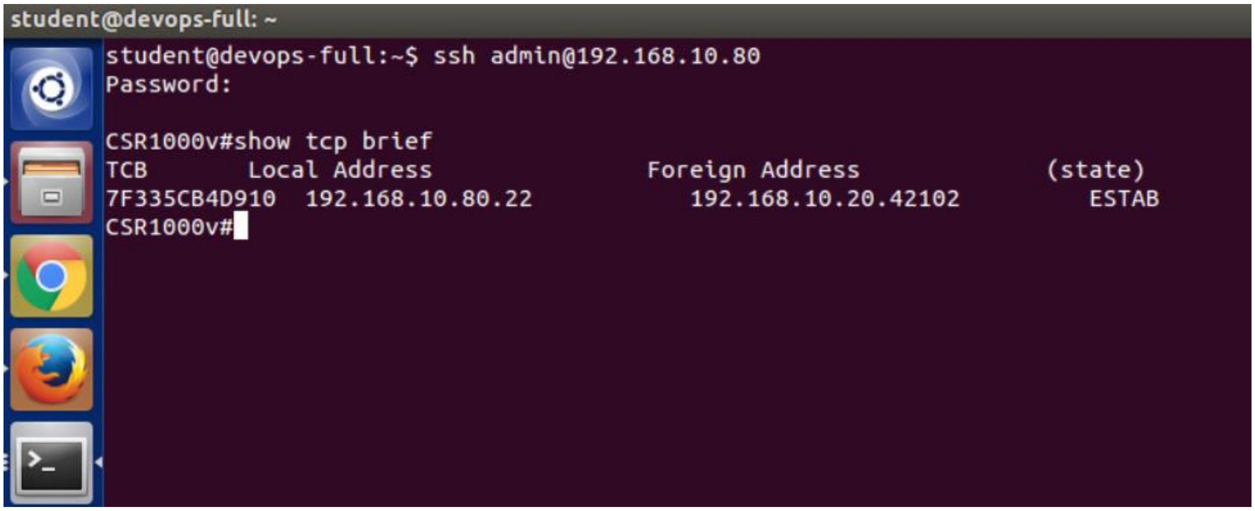
Close window on exit: ○ Always ○ Never ● Only on clean exit

7. Login with user: **admin** and password: **cisco** and verify that you can see your TCP port 22 SSH session:



```
login as: admin
Using keyboard-interactive authentication.
Password:

CSR1000v#show tcp brief
TCB          Local Address               Foreign Address            (state)
7F3366DA30C0  192.168.10.80.22              192.168.10.40.56452           ESTAB
CSR1000v#
```

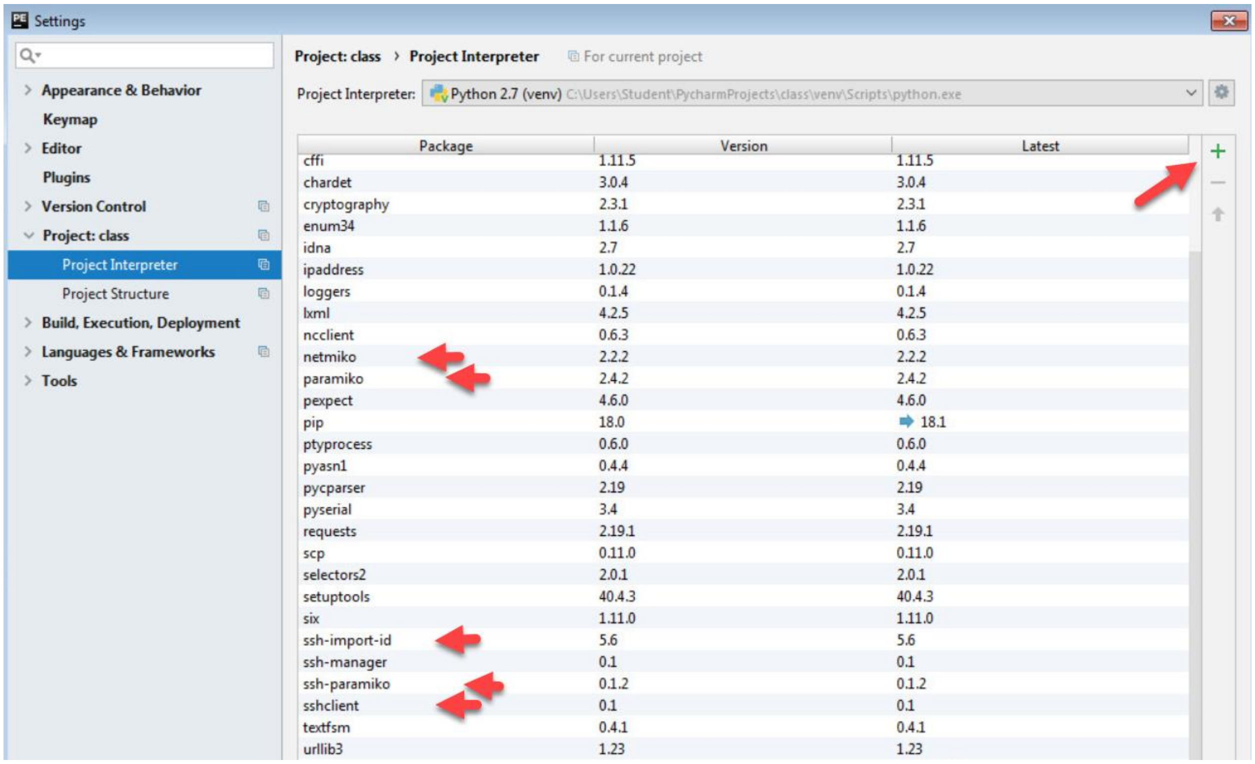
8. To further test, return to your **Ubuntu Configured** VM and open an SSH session to your CSR1000v router.



```
student@devops-full:~$ ssh admin@192.168.10.80
Password:

CSR1000v#show tcp brief
TCB          Local Address              Foreign Address          (state)
7F335CB4D910  192.168.10.80.22           192.168.10.20.42102        ESTAB
CSR1000v#
```

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:

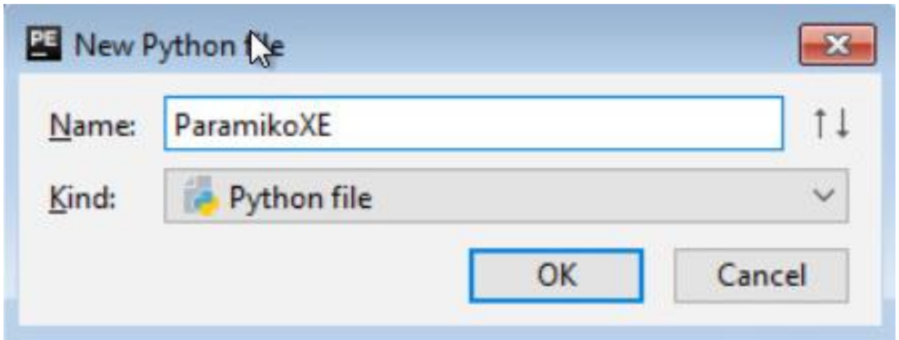
```
paramiko
netmiko
sshclient
ssh-paramiko
ssh-import-id
```



Project Interpreter: Python 2.7 (venv) C:\Users\Student\PycharmProjects\class\venv\Scripts\python.exe

Package	Version	Latest
cffi	1.11.5	1.11.5
chardet	3.0.4	3.0.4
cryptography	2.3.1	2.3.1
enum34	1.1.6	1.1.6
idna	2.7	2.7
ipaddress	1.0.22	1.0.22
loggers	0.1.4	0.1.4
lxml	4.2.5	4.2.5
ncclient	0.6.3	0.6.3
netmiko	2.2.2	2.2.2
paramiko	2.4.2	2.4.2
pexpect	4.6.0	4.6.0
pip	18.0	➡ 18.1
ptyprocess	0.6.0	0.6.0
pyasn1	0.4.4	0.4.4
pycparser	2.19	2.19
pyserial	3.4	3.4
requests	2.19.1	2.19.1
scp	0.11.0	0.11.0
selectors2	2.0.1	2.0.1
setuptools	40.4.3	40.4.3
six	1.11.0	1.11.0
ssh-import-id	5.6	5.6
ssh-manager	0.1	0.1
ssh-paramiko	0.1.2	0.1.2
sshclient	0.1	0.1
textfsm	0.4.1	0.4.1
urllib3	1.23	1.23

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

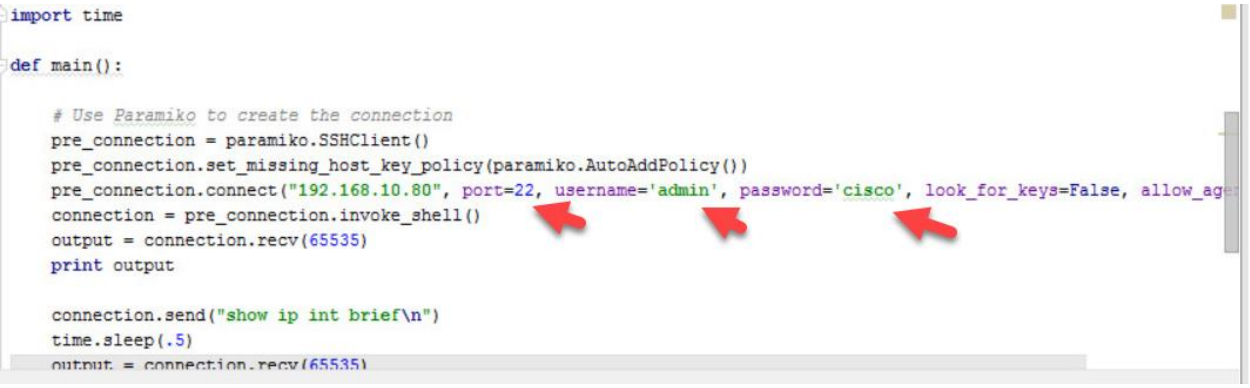


Name: ParamikoXE

Kind: Python file

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

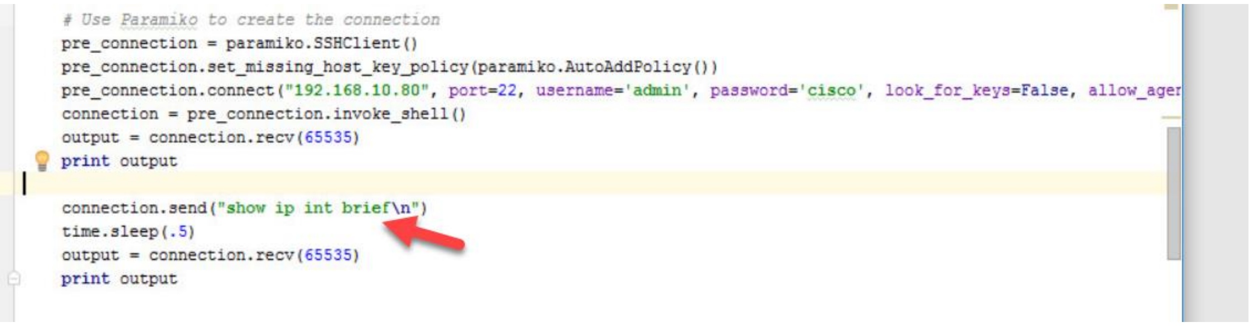
16. Ensure that this Python script has the correct SSH credentials to your IOS XE router:
    Host: **192.168.10.80**
    Port: **22**
    Username: **admin**
    Password: **cisco**



```
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_ag
    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)
```

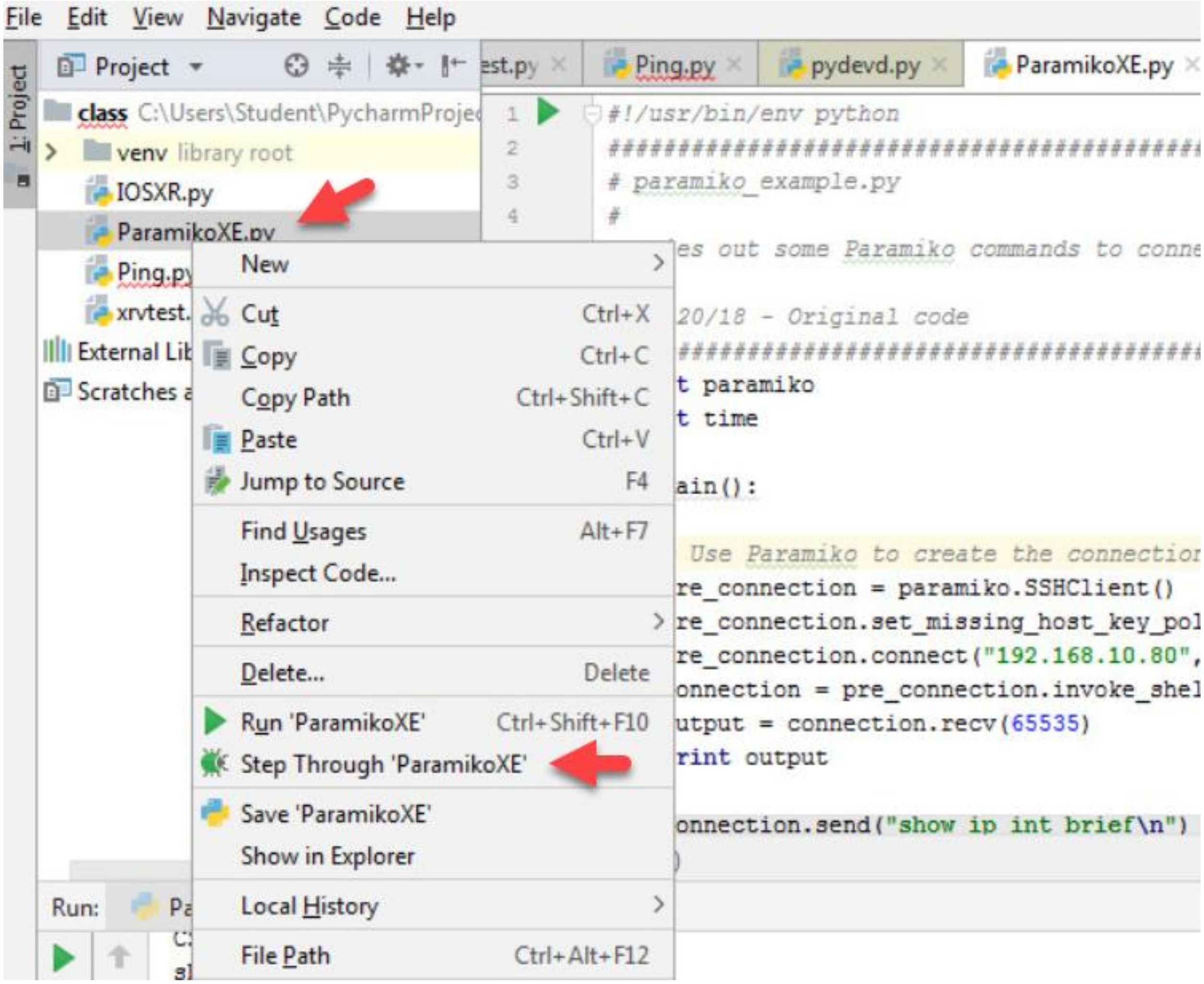
Note: Paramiko only uses TCP port 22.

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



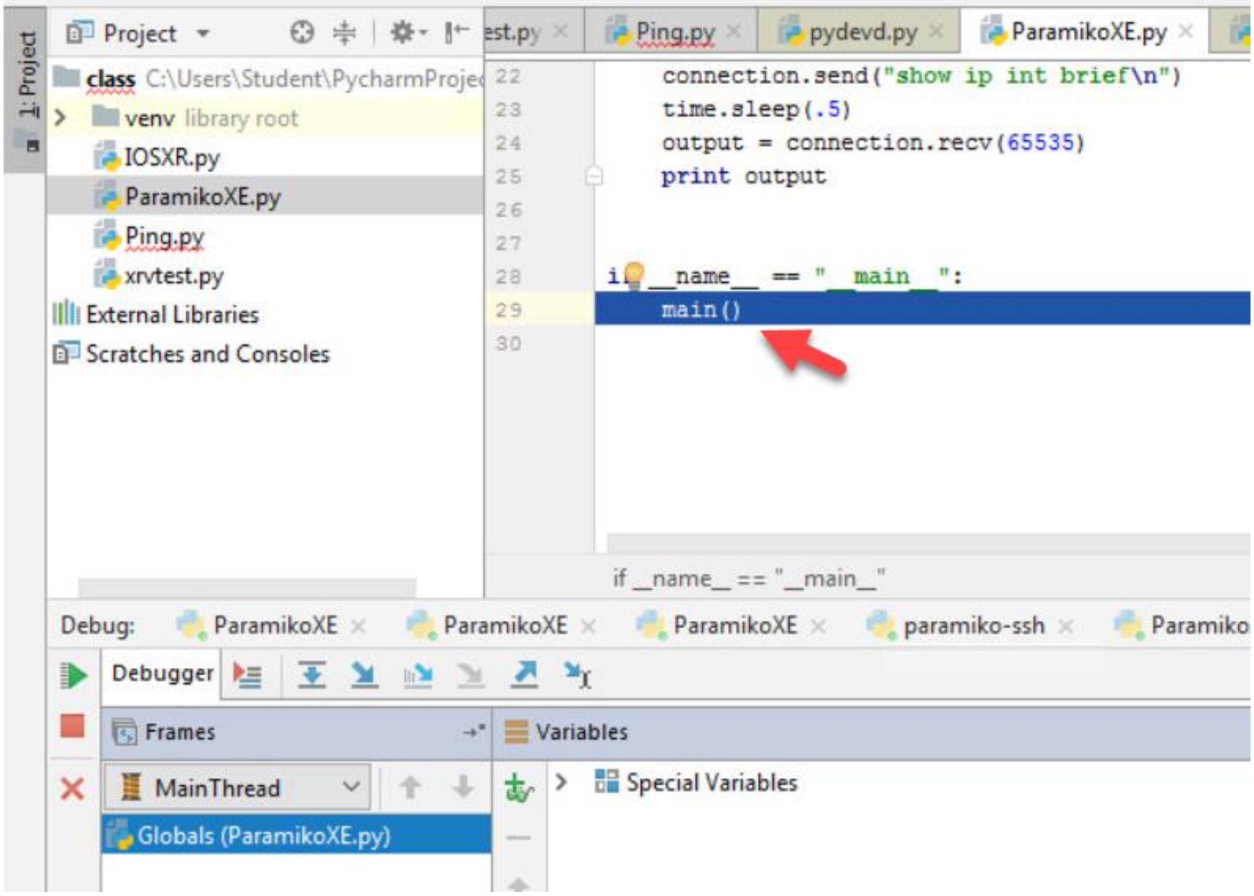
```
    # Use Paramiko to create the connection
    pre_connection = paramiko.SSHClient()
    pre_connection.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    pre_connection.connect("192.168.10.80", port=22, username='admin', password='cisco', look_for_keys=False, allow_ager
    connection =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



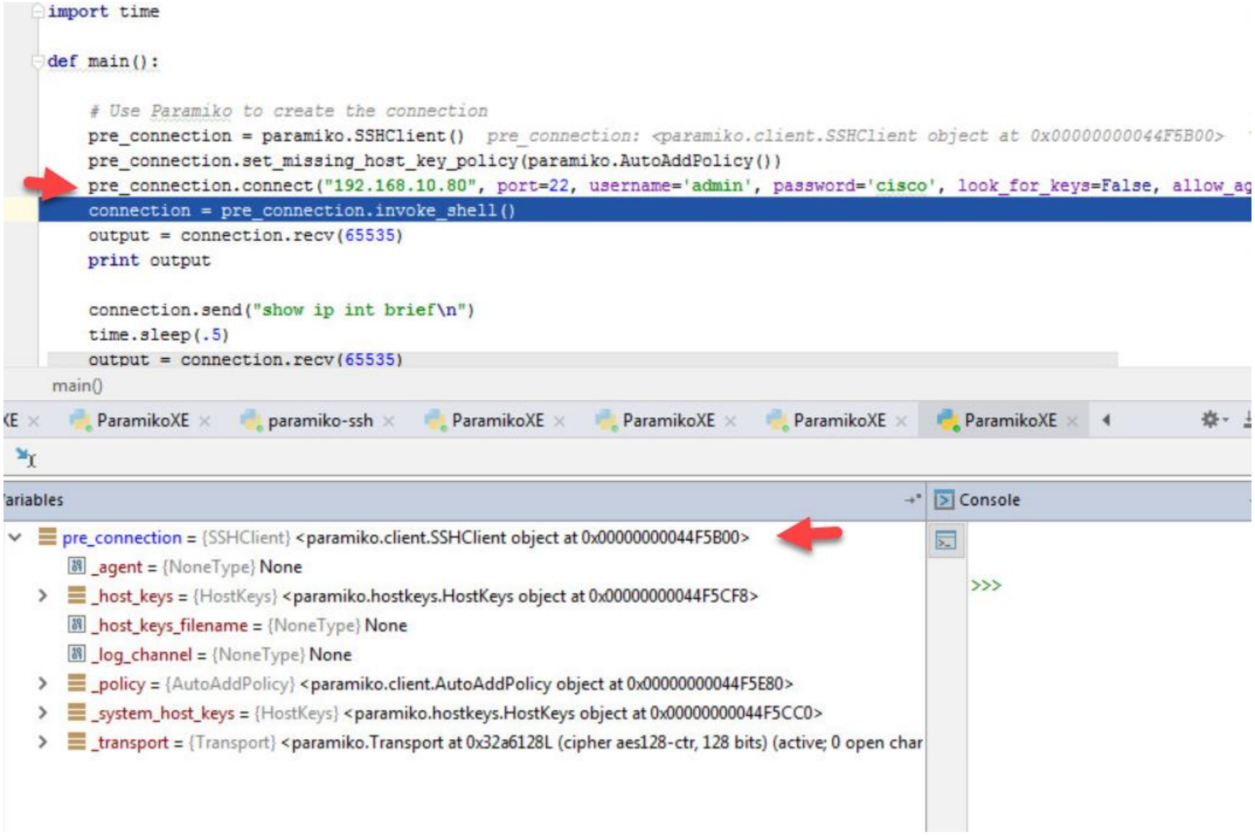
```
#!/usr/bin/env python
##########################################
# paramiko_example.py
#
es out some Paramiko commands to conne

20/18 - Original code
###################################
t paramiko
t time

ain():

 Use Paramiko to create the connectio
re_connection = paramiko.SSHClient()
re_connection.set_missing_host_key_pol
re_connection.connect("192.168.10.80",
onnection = pre_connection.invoke_shel
utput = connection.recv(65535)
rint output

onnection.send("show ip int brief\n")
)
```

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.



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


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

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

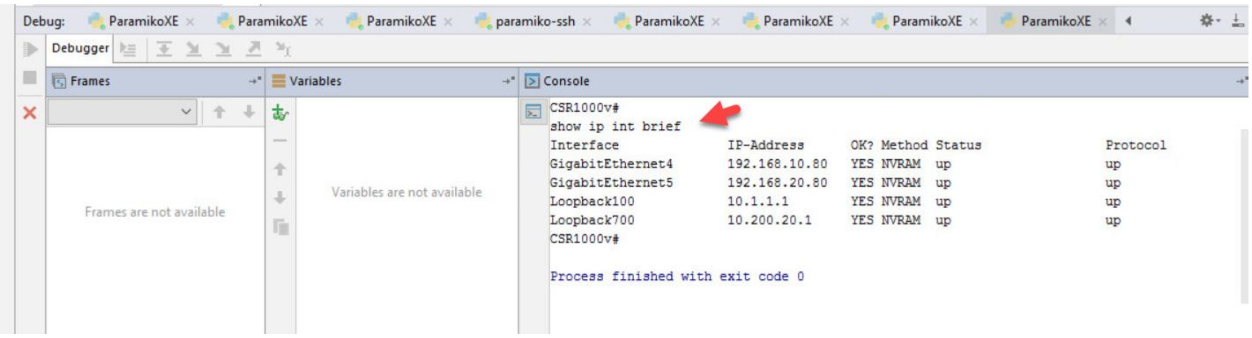


```
import time

def main():

    # Use Paramiko to create the connection
    pre_connection = paramiko.SSHClient()  pre_connection: <paramiko.client.SSHClient object at 0x00000000044F5B00>
    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_ag
    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)
```

```
pre_connection = {SSHClient} <paramiko.client.SSHClient object at 0x00000000044F5B00>
    _agent = {NoneType} None
    _host_keys = {HostKeys} <paramiko.hostkeys.HostKeys object at 0x00000000044F5CF8>
    _host_keys_filename = {NoneType} None
    _log_channel = {NoneType} None
    _policy = {AutoAddPolicy} <paramiko.client.AutoAddPolicy object at 0x00000000044F5E80>
    _system_host_keys = {HostKeys} <paramiko.hostkeys.HostKeys object at 0x00000000044F5CC0>
    _transport = {Transport} <paramiko.Transport at 0x32a6128L (cipher aes128-ctr, 128 bits) (active; 0 open char
```

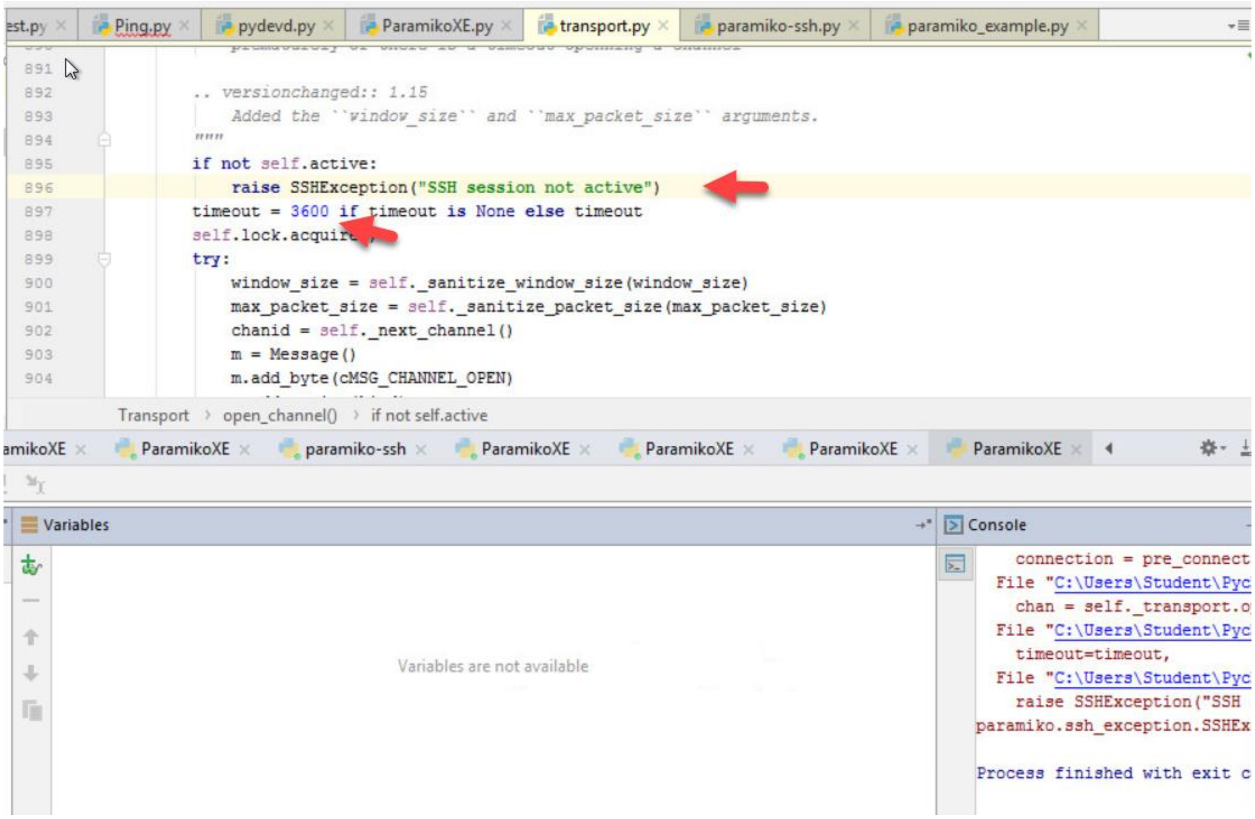
```
>>>
```

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



```
CSR1000v#
show ip int brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet4       192.168.10.80   YES NVRAM  up                    up
GigabitEthernet5       192.168.20.80   YES NVRAM  up                    up
Loopback100            10.1.1.1        YES NVRAM  up                    up
Loopback700            10.200.20.1     YES NVRAM  up                    up
CSR1000v#

Process finished with exit code 0
```

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.

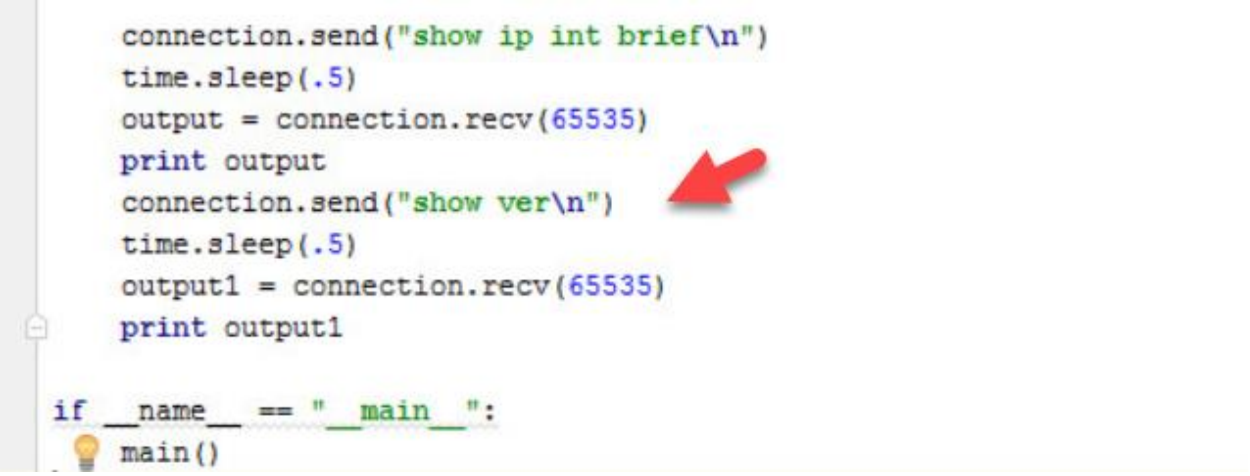


```
        .. versionchanged:: 1.15
            Added the ``window_size`` and ``max_packet_size`` arguments.
        """
        if not self.active:
            raise SSHException("SSH session not active")
        timeout = 3600 if timeout is None else timeout
        self.lock.acquire()
        try:
            window_size = self._sanitize_window_size(window_size)
            max_packet_size = self._sanitize_packet_size(max_packet_size)
            chanid = self._next_channel()
            m = Message()
            m.add_byte(cMSG_CHANNEL_OPEN)
```

Variables are not available

```
    connection = pre_connect
  File "C:\Users\Student\Pyc
    chan = self._transport.o
  File "C:\Users\Student\Pyc
    timeout=timeout,
  File "C:\Users\Student\Pyc
    raise SSHException("SSH
paramiko.ssh_exception.SSHEx

Process finished with exit c
```

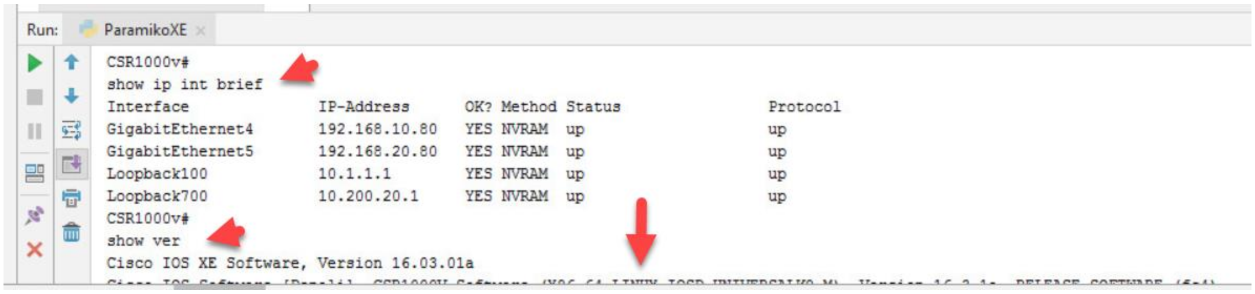
22. Add another command **show ver** to your script with a new output variable.



```
    connection.send("show ip int brief\n")
    time.sleep(.5)
    output = connection.recv(65535)
    print output
    connection.send("show ver\n")
    time.sleep(.5)
    output1 = connection.recv(65535)
    print output1

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

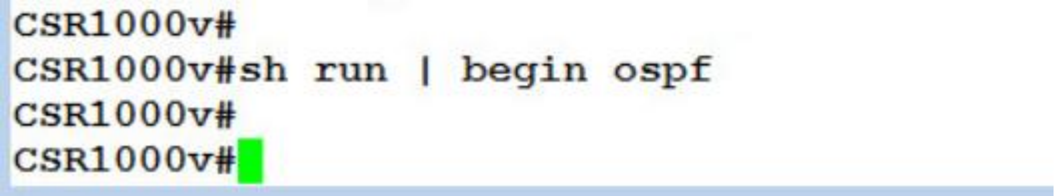
Note: You can abbreviate the **show version** command as **show ver** exactly as you can in the IOS device. Paramiko is only sending the commands exactly as you give them.

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.



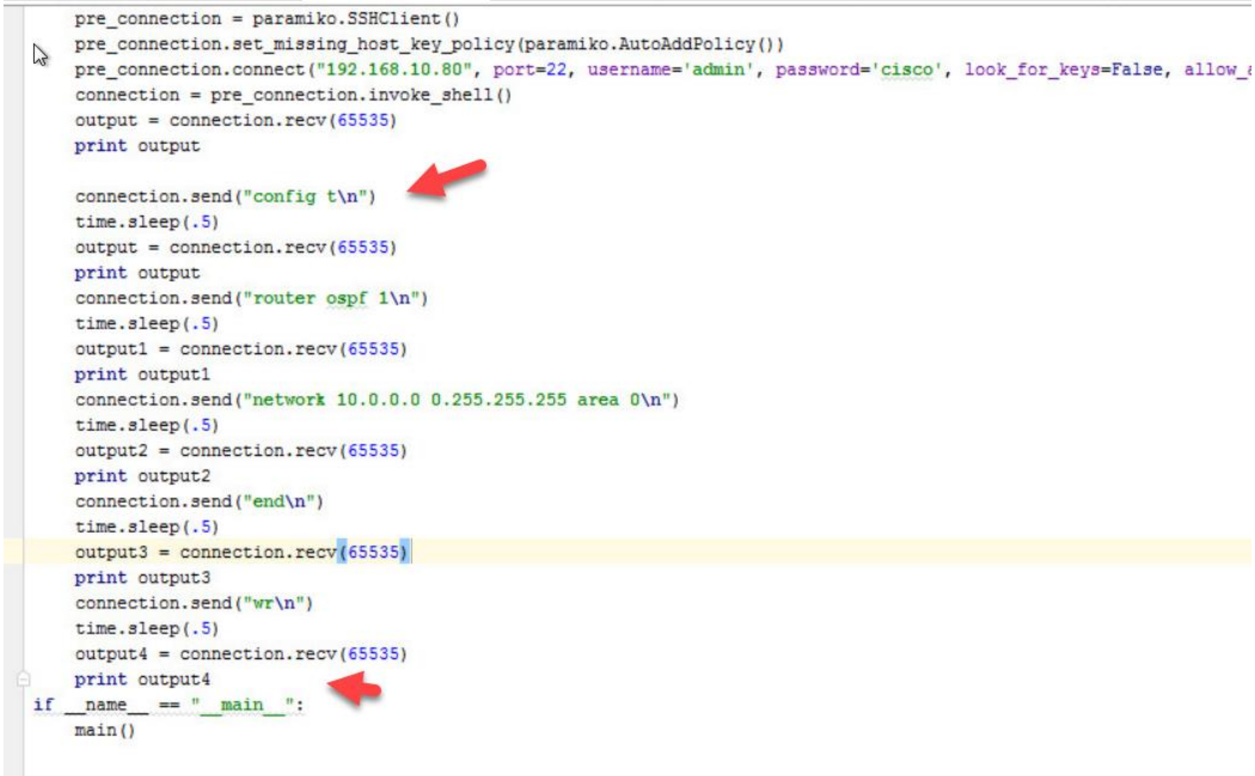
```
CSR1000v#
show ip int brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet4       192.168.10.80   YES NVRAM  up                    up
GigabitEthernet5       192.168.20.80   YES NVRAM  up                    up
Loopback100            10.1.1.1        YES NVRAM  up                    up
Loopback700            10.200.20.1     YES NVRAM  up                    up
CSR1000v#
show ver
Cisco IOS XE Software, Version 16.03.01a
```

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



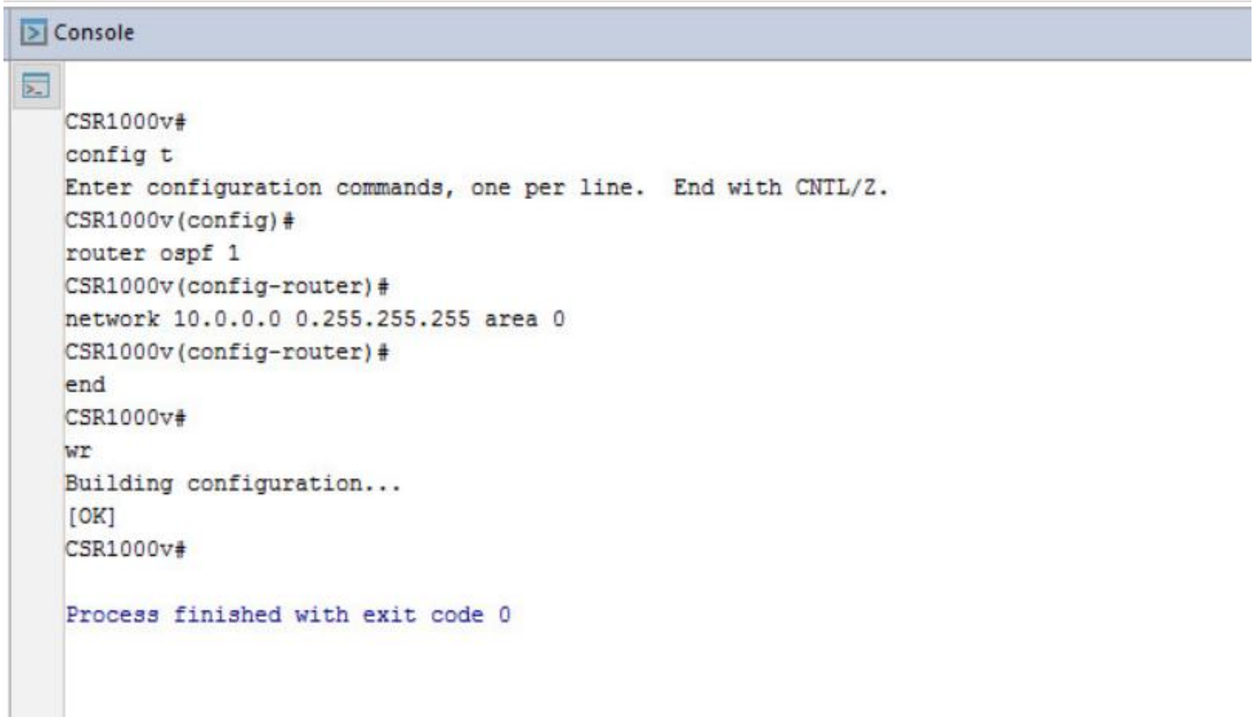
```
CSR1000v#
CSR1000v#sh run | begin ospf
CSR1000v#
CSR1000v#
```

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



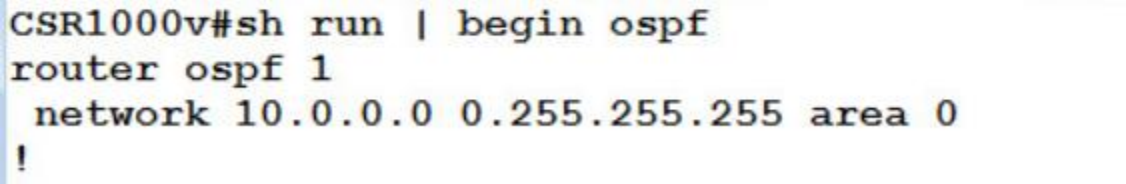
```
    pre_connection = paramiko.SSHClient()
    pre_connection.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    pre_connection.connect("192.168.10.80", port=22, username='admin', password='cisco', look_for_keys=False, allow_
    connection = pre_connection.invoke_shell()
    output = connection.recv(65535)
    print output

    connection.send("config t\n")
    time.sleep(.5)
    output = connection.recv(65535)
    print output
    connection.send("router ospf 1\n")
    time.sleep(.5)
    output1 = connection.recv(65535)
    print output1
    connection.send("network 10.0.0.0 0.255.255.255 area 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()
```

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.



```
CSR1000v#
config t
Enter configuration commands, one per line.  End with CNTL/Z.
CSR1000v(config)#
router ospf 1
CSR1000v(config-router)#
network 10.0.0.0 0.255.255.255 area 0
CSR1000v(config-router)#
end
CSR1000v#
wr
Building configuration...
[OK]
CSR1000v#

Process finished with exit code 0
```

27. Verify these OSPF commands appear in your router



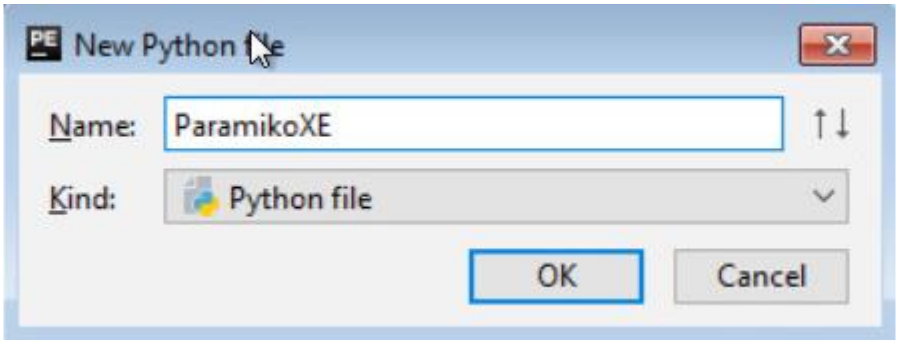
```
CSR1000v#sh run | begin ospf
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
!
```

Task 2: Router 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.**

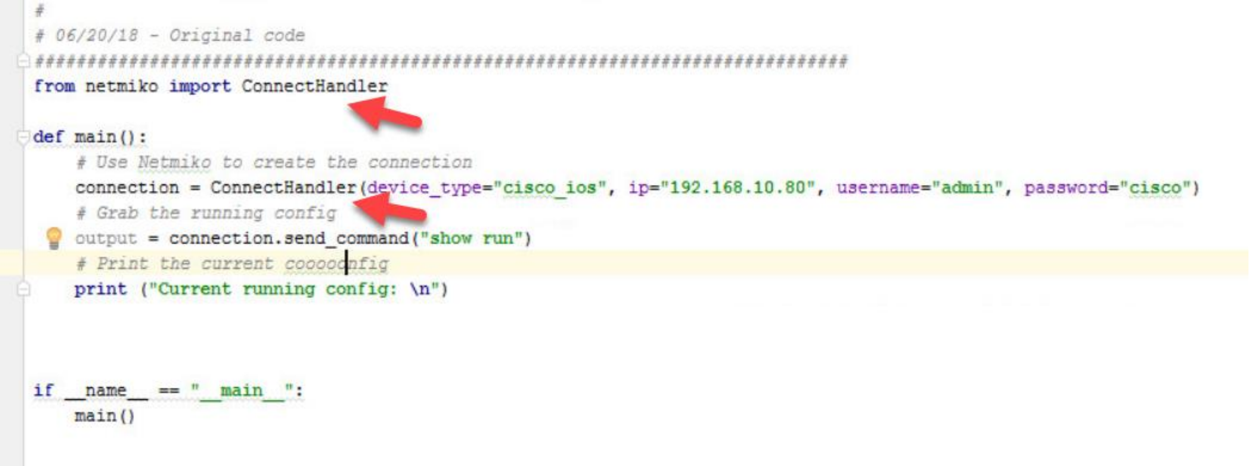


Name: ParamikoXE

Kind: Python file

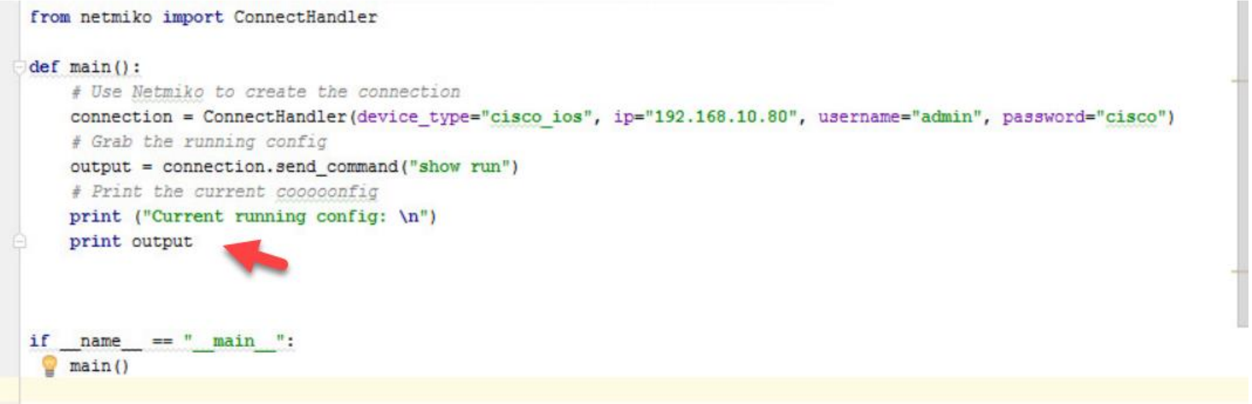
2. Select **File > Open**. Select the file **Z:\Python\netmiko_example.py**
3. Copy the code into the **NetmikoXE.py** window.
4. Study the code.

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**.



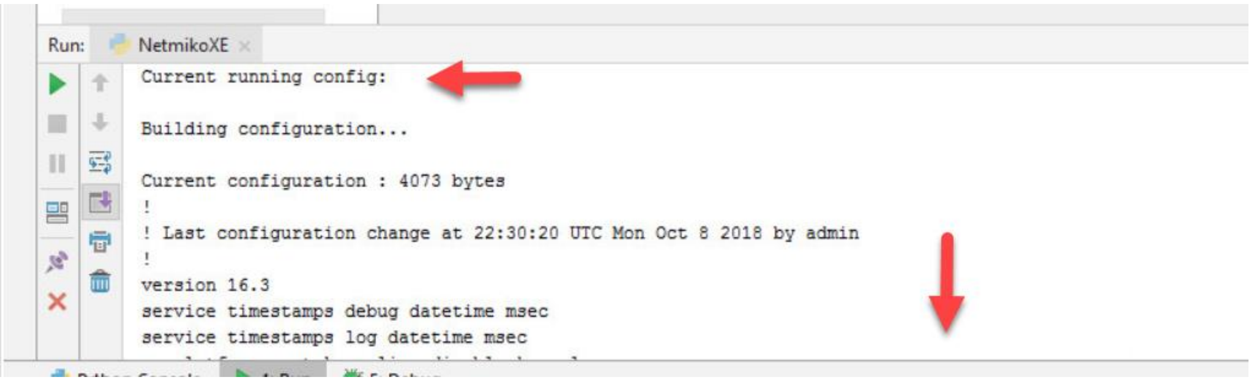
```
#
# 06/20/18 - Original code
################################################################################
from netmiko import ConnectHandler

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.



```
Current running config:

Building configuration...

Current configuration : 4073 bytes
!
! Last configuration change at 22:30:20 UTC Mon Oct 8 2018 by admin
!
version 16.3
service timestamps debug datetime msec
service timestamps log datetime msec
```

Challenge results

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