

# PythonPing Network Scanner Tutorial

## Contents

PythonPing Network Scanner Tutorial .....	1
Python Tabs and Spaces Issue .....	1
Find Network Address .....	1
Find Your Network IP Address in Windows .....	2
How It Works.....	3
Tutorial 1: PythonPing .....	4
Tutorial 2: Scan Your Network from Windows .....	6
Tutorial 3: Scan Your Network from Linux .....	10
PythonPing Scanner in Linux .....	10
Assignment Submission.....	11

Time required: 60 minutes

## Python Tabs and Spaces Issue

Visual Studio Code automatically changes a tab into four spaces. Other editors, like geany and nano in Linux, do not. You can end up with a combination of spaces and tabs. Python doesn't like a combination, it wants either one or the other. The preferred method is spaces.

### Recommendation:

1. Create your Python files in Visual Studio Code in Windows.
2. Copy and paste the code into either nano or geany in Linux.

**Objective:** Write a cross platform Python script that uses branching, looping, and pythonping to scan a local network.

## Find Network Address

If you are using a VM, set to a bridged adapter.

---

## Find Your Network IP Address in Windows

Use the network address of your local network. Example: 192.168.0.0/24

**NOTE:** 192.168.56.1 is the VirtualBox adapter address, that is not your network address.

1. Enter the following command at the command prompt: **ipconfig /all**
2. The screenshot below shows my network at home, 192.168.9 Your IP address will probably be different. I have an Ethernet adapter, you may have a wireless adapter.
3. Notice that my IP address information includes a Default Gateway, DHCP Server, and DNS Servers. Those are needed for a functioning network connection.
4. Note that my IPv4 Address for my computer is **192.168.9.101** My subnet Mask is **255.255.255.0** This makes my network a standard Class C network.

NOTE: A typical home network address is **192.168.1.0/24**

If you are not sure about this, please contact me. You will get a 0 for this assignment if you do not provide a screenshot showing a successful scan of your network.

```

C:\Users\Bill.THECOMPUTERGUY>ipconfig /all

Windows IP Configuration

Host Name . . . . . : Bill-PC
Primary Dns Suffix . . . . . : thecomputerguy.local
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No
DNS Suffix Search List. . . . . : thecomputerguy.local
                                  lan

Ethernet adapter Ethernet:

    Connection-specific DNS Suffix . : lan
    Description . . . . . : Realtek PCIe GbE Family Controller
    Physical Address. . . . . : 2C-F0-5D-A2-AC-3E
    DHCP Enabled. . . . . : Yes
    Autoconfiguration Enabled . . . . : Yes
    Link-local IPv6 Address . . . . . : fe80::b08h:b38e:4b9d:3e9b%7(Preferred)
    IPv4 Address. . . . . : 192.168.9.101(Preferred)
    Subnet Mask . . . . . : 255.255.255.0
    Lease Obtained. . . . . : Friday, April 15, 2022 6:32:36 AM
    Lease Expires . . . . . : Sunday, April 17, 2022 6:32:37 AM
    Default Gateway . . . . . : 192.168.9.1
    DHCP Server . . . . . : 192.168.9.1
    DHCPv6 IAID . . . . . : 103608413
    DHCPv6 Client DUID. . . . . : 00-01-00-01-27-89-4B-A4-2C-F0-5D-A2-AC-3E
    DNS Servers . . . . . : 192.168.9.10
                          8.8.8.8
    NetBIOS over Tcpip. . . . . : Enabled

Ethernet adapter VirtualBox Host-Only Network:

    Connection-specific DNS Suffix . : 
    Description . . . . . : VirtualBox Host-Only Ethernet Adapter
    Physical Address. . . . . : 0A-00-27-00-00-0F
    DHCP Enabled. . . . . : No
    Autoconfiguration Enabled . . . . : Yes
    Link-local IPv6 Address . . . . . : fe80::b0d1:22cf:dacc:d009%15(Preferred)
    IPv4 Address. . . . . : 192.168.56.1(Preferred)
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 
    DHCPv6 IAID . . . . . : 168427559
    DHCPv6 Client DUID. . . . . : 00-01-00-01-27-89-4B-A4-2C-F0-5D-A2-AC-3E
    DNS Servers . . . . . : fec0:0:0:ffff::1%1
                          fec0:0:0:ffff::2%1
                          fec0:0:0:ffff::3%1
    NetBIOS over Tcpip. . . . . : Enabled

```

## How It Works

In this example, the **scan()** function takes a subnet as input (e.g., "192.168.1" for a typical home network) and scans all IP addresses in that subnet using ICMP ping. It uses the pythonping library to send ICMP Echo Request packets and receive ICMP Echo Reply packets to determine if hosts are online. If a host responds to the ping, its IP address is printed along with the response time, and it's added to the list of live hosts. Finally, the list of live hosts is printed at the end.

**Note:** ICMP ping may not always work, as some networks or hosts may have ICMP ping disabled for security reasons. In such cases, the `pythonping` library may not be effective for scanning hosts. Please ensure that you have the necessary permissions and legal rights before scanning any network or host. Always follow ethical guidelines and applicable laws when conducting network scanning or security testing.

## Tutorial 1: PythonPing

Security professionals often need to automate or create tools to help them conduct security tests. In this activity, you will write a Python script that uses the ping command and a for loop to ping IP numbers for an entire network.

1. Install the pythonPing library. Enter the following command at a command prompt.

```
# Windows
pip install pythonping
# Linux
sudo pip3 install pythonping
```

2. Create a new file called **pythonping\_scanner\_1.py**.
3. Enter the following code including the comments.

```

1  """
2      Filename: pythonping_scanner_1.py
3      This program prompts the user to enter a host address
4      Uses the pythonping library to send out ping packets
5  """
6
7  # Windows: pip install pythonping
8  # Linux Debian distributions:
9  #     sudo apt update
10 #     sudo pip3 install pythonping
11 #     Use sudo to run script: sudo python3 pythonping_scanner.py
12 from pythonping import ping
13
14
15 def main():
16     # Print program title
17     print(48 * "_")
18     print("|                      Python Network Ping                      |")
19     print(48 * "_")
20     # Get ip address or hostname
21     host_address = input(" Enter single IP address or hostname: ")
22     # Scan host
23     result = scan(host_address)
24     print(result)
25
26
27 # ----- PING HOST ADDRESS -----#
28 def scan(host_address: str) -> str:
29     """Ping the target, return a string response."""
30     result = ping(host_address)
31     return result
32
33
34 # If a standalone program, call the main function
35 # Else, use as a module
36 if __name__ == "__main__":
37     main()

```

Example run:

This program pings a single IP address or host name.

```

-----
|          Python Network Ping Scanner          |
-----
Press CTRL C to exit
IP address or hostname: 1.1.1.1
Reply from 1.1.1.1, 29 bytes in 15.44ms
Reply from 1.1.1.1, 29 bytes in 15.62ms
Reply from 1.1.1.1, 29 bytes in 16.05ms
Reply from 1.1.1.1, 29 bytes in 15.58ms

Round Trip Times min/avg/max is 15.44/15.67/16.05 ms

```

```
-----  
|           Python Network Ping Scanner           |  
-----  
Press CTRL C to exit  
IP address or hostname: lab.wncc.net  
Reply from 198.206.239.241, 29 bytes in 22.59ms  
Reply from 198.206.239.241, 29 bytes in 22.25ms  
Reply from 198.206.239.241, 29 bytes in 21.8ms  
Reply from 198.206.239.241, 29 bytes in 22.06ms  
  
Round Trip Times min/avg/max is 21.8/22.18/22.59 ms
```

## Tutorial 2: Scan Your Network from Windows

Let's scan your entire local network. Only scan networks where you have permission.

```

1  #!/usr/bin/env python3
2  """
3      Filename: pythonping_scanner_2.py
4      This program prompts the user to enter a class C network address
5      it uses the pythonping library to send out ping packets
6  """
7
8  # Windows: pip install pythonping
9  # Linux Debian distributions:
10 # sudo apt update
11 # sudo pip install pythonping
12 # Use sudo to run script: sudo python3 pythonping_scanner.py
13 from pythonping import ping
14 # https://docs.python.org/3/library/ipaddress.html
15 # Convert ip/mask to list of hosts
16 import ipaddress
17 import sys
18
19
20 def main():
21     display_program_title()
22
23     # ----- SET NETWORK ADDRESS -----#
24     """Set network address x.x.x.x/x or x.x.x.x/x.x.x.x from user"""
25     # Change this to the default value of your network
26     default_local_network = "10.0.1.0/24"
27
28     # Prompt the user to input a network address and press Enter
29     # If they press enter without an network address, the default is used
30     network_address = input(
31         "\n Enter your network address (ex. 10.0.1.0/24): "
32     ) or default_local_network
33
34     # Create a network address object from user input
35     ip_net = ipaddress.ip_network(network_address)
36
37     # Convert all hosts on entered network into a list
38     all_hosts = list(ip_net.hosts())
39     scan(all_hosts)

```

```

42 # ----- SCAN NETWORK -----#
43 def scan(all_hosts: str):
44     """Ping all Class C IP addresses 1-254"""
45     for host_address in all_hosts:
46         # Convert the ip address to a string
47         ip = str(host_address)
48         try:
49             # Ping the IP address with two packets
50             result = ping(
51                 ip,          # Target IP address
52                 count=1,     # Number of pings
53                 timeout=2    # Timeout in seconds
54             )
55
56             # If there was a successful ping response
57             if result.success():
58                 # Response time less than 2000ms, target is active
59                 print(f" {ip:14}-> RTT: {result.rtt_avg_ms:>6.2f}ms")
60             else:
61                 print(f" {ip} Inactive")
62
63         except KeyboardInterrupt:
64             # Catch the Keyboard Interrupt exception
65             print("CTRL-C pressed. Exit program")
66             sys.exit()
67
68         except Exception as e:
69             # Catch all other exceptions
70             # Print out the exception error for debugging
71             print("Sorry", e)
72             sys.exit()

```



```

75 # ----- DISPLAY PROGRAM TITLE -----#
76 def display_program_title():
77     """Print program title to console"""
78     print(35 * "-")
79     print("|   Python Network Ping Scanner   |")
80     print(35 * "-")
81     print(" Press CTRL C to exit")
82
83
84 # If a standalone program, call the main function
85 # Else, use as a module
86 if __name__ == "__main__":
87     main()

```

1. Open a command prompt and navigate to the folder containing your **pythonping\_scanner.py** file.
2. Type: **python pythonping\_scanner.py** Press **Enter**.
3. If you have no errors, your program should begin pinging IP addresses, as shown below.
4. You can let the program run to completion (which will take a long time) or press **CTRL C** to terminate the script.

Example run in Windows:

```

-----
|   Python Network Ping Scanner   |
-----
Press CTRL C to exit

Enter your network address (ex. 192.168.9.0/24): 192.168.9.0/24
192.168.9.1 --> Alive RTT:  0.53
192.168.9.2 Inactive
192.168.9.3 Inactive
192.168.9.4 Inactive
192.168.9.5 Inactive
192.168.9.6 Inactive
192.168.9.7 Inactive
192.168.9.8 Inactive
192.168.9.9 Inactive
192.168.9.10 --> Alive RTT:  0.39
192.168.9.11 Inactive

```

## Tutorial 3: Scan Your Network from Linux

**NOTE:** If your class is using Linux for other assignments, complete this portion of the assignment.

This is a cross platform script. It will work on Windows or Linux or Mac. Copy and paste the code into Linux.

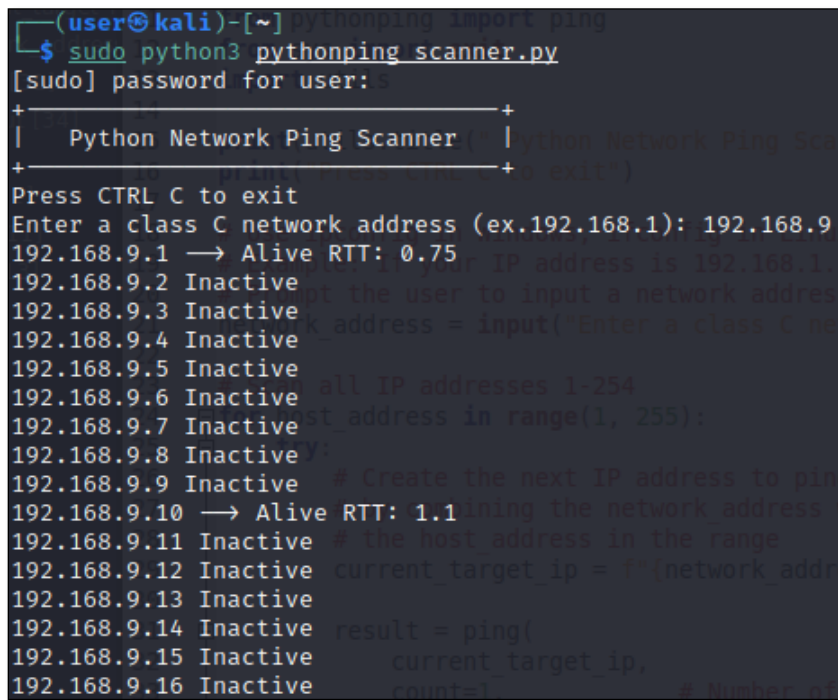
Your network in Linux should be the same as your network in Windows. If it isn't, set your VirtualBox network settings to Bridged Adapter.

---

### PythonPing Scanner in Linux

This will work on most Linux systems include older distributions of Kali Linux. Try this first.

1. Open a terminal session.
2. Determine your version of Python: **python3**
3. Press **CTRL D** to exit the Python prompt.
4. Install the **pythonping** library → **sudo apt install python3-pythonping**
5. In a terminal → **sudo python3 pythonping\_scanner.py**
6. Example run in Linux:



```
(user@kali)-[~] pythonping import ping
$ sudo python3 pythonping_scanner.py
[sudo] password for user:s
+-----+
| Python Network Ping Scanner (" Python Network Ping Scanner") |
+-----+
Press CTRL C to exit
Enter a class C network address (ex.192.168.1): 192.168.9
192.168.9.1 → Alive RTT: 0.75
192.168.9.2 Inactive
192.168.9.3 Inactive
192.168.9.4 Inactive
192.168.9.5 Inactive
192.168.9.6 Inactive
192.168.9.7 Inactive
192.168.9.8 Inactive
192.168.9.9 Inactive
192.168.9.10 → Alive RTT: 1.1
192.168.9.11 Inactive
192.168.9.12 Inactive
192.168.9.13 Inactive
192.168.9.14 Inactive
192.168.9.15 Inactive
192.168.9.16 Inactive
```

This scanner isn't very fast, but it works. To get faster results, we would use multithreading.

---

### **Assignment Submission**

1. Attach all program files.
2. Attach a screenshot of each successful program run.
3. If you do not attach a screenshot of a successful program run on your correct local network address, you will receive a 0 for this assignment.
4. Submit the assignment in Blackboard.