

Part 1: Python Network Scanner

Contents

Part 1: Python Network Scanner	1
What is ARP?	2
What is scapy?.....	2
Setup scapy in Windows.....	2
Install Npcap	2
Install scapy in Windows	3
Install scapy in Linux	3
Edit Python in Linux: geany (Optional)	3
Edit Python in Linux: Visual Studio Code (Optional)	4
Install scapy in Linux	4
Determine Your Network IP Range	4
Network Scanner with scapy	5
Assignment Submission.....	7

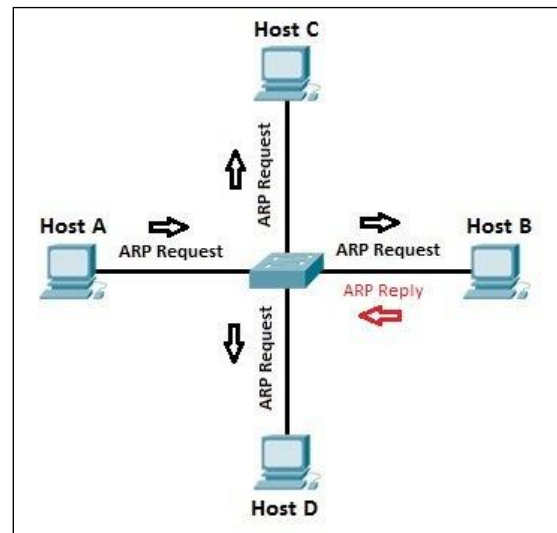
Time required: 60 minutes

NOTE: This assignment was developed and tested on Windows 10, Kali Linux 2021, and Fedora 28.

What is ARP?

Most computer programs/applications use **logical addresses (IP address)** to send/receive messages. The actual communication happens over the **physical address (MAC address)** i.e from layer 2 of OSI model. **Address Resolution Protocol (ARP)** translates **Internet Protocol (IP)** addresses to **Media Access Control (MAC)** addresses.

1. Host A wants to communicate with Host B.
2. Host A sends out a broadcast ARP request to all hosts on the network.
3. Host B replies with its IP address and MAC address.
4. Host A and Host B can communicate using MAC addresses.



What is scapy?

[scapy](#) is a powerful interactive packet manipulation program. It can forge or decode packets of a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more. It can easily handle most classic tasks like scanning, tracerouting, probing, unit tests, attacks, or network discovery. It runs on Linux, Windows, and OSX. The code base runs on Python 2 and Python 3. We will use Python 3.

Setup scapy in Windows

Windows needs three parts to use scapy.

1. Python 3 www.python.org
2. Npcap (For capturing and sending packets.)
3. scapy (Python library for manipulating packets.)

Install Npcap

[Npcap](#) is the Nmap Project's packet sniffing (and sending) library for Windows. This is needed for **scapy** to communicate with the network.

1. Go to <https://nmap.org/npcap>
2. Download and install the latest **Npcap** for Windows.

Install scapy in Windows

In Windows, **scapy** can't be installed using **pip**. We must download and install **scapy** from **github**.

1. Click <https://github.com/secdev/scapy/archive/master.zip> to download **master.zip**.
2. Unzip **master.zip** into a folder you can access using the command prompt.
3. Using the command prompt → Navigate to the **scapy-master** folder with all the files in it.
4. Run the command → **python setup.py install**
5. Your command should be successful and show something like this at the end.

```
Installed c:\program files\python39\lib\site-packages\scapy-git_archive.deva9f8d0244d-py3.9.egg
Processing dependencies for scapy==git-archive.deva9f8d0244d
Finished processing dependencies for scapy==git-archive.deva9f8d0244d
D:\Temp\scapy-master>_
```

Install scapy in Linux

Kali Linux already has Python 3 and libcap for capturing and sending packets. We need to install scapy.

1. pip3 install scapy

Edit Python in Linux: geany (Optional)

Geany is a lightweight code editor that works very well in Linux. This following will install geany on Linux.

1. **sudo apt update**
2. **sudo apt update**
3. **sudo apt install geany**

To use geany from the terminal:

1. Create and edit a Python file: **geany network_scanner1.py**

Edit Python in Linux: Visual Studio Code (Optional)

<https://code.visualstudio.com/docs/setup/linux> contains information on how to install Visual Studio code on other Linux distributions.

1. Go to <https://code.visualstudio.com>
2. Download the **.deb** package. It should download to the **Downloads** folder.
3. Open a **Terminal** session.
4. Navigate to the **Downloads** folder.
5. Run: **sudo apt install ./code_1.55.0-1617120720_amd64.deb**
(Your file name is probably different. This was the filename at the time of this writing.)
6. **Visual Studio Code** will be installed as an application.
7. Create and edit a Python file: **code network_scanner1.py**

Install scapy in Linux

1. Open a terminal session.
2. Update your package list: **sudo apt update**
3. Install pip3: **sudo apt install python3-pip**
4. Install scapy: **sudo pip3 install scapy**
5. Update scapy: **sudo pip3 install scapy -U**

Determine Your Network IP Range

The first step is to determine the address range of your local network.

1. Connect your Linux or any other VM being used in this assignment to the VirtualBox **Bridged Adapter**.
2. In Windows: At a command prompt → type **ipconfig**
In Linux: **ifconfig** or **ip -a**

```

D:\Temp>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

    Connection-specific DNS Suffix  . : lan
    Link-local IPv6 Address . . . . . : fe80::b08b:b38e:4b9d:3e9b%4
    IPv4 Address. . . . . : 192.168.9.101
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.9.1

Ethernet adapter VirtualBox Host-Only Network:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::5c47:3564:10ba:33fe%6
    IPv4 Address. . . . . : 192.168.56.1
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 

```

You will have one network adapter with a Default Gateway. That will help you identify your network.

Subnet Mask: This network has a class C subnet: **255.255.255.0** which can be designated as **/24**. If your subnet mask is different, Google the / address of your subnet mask.

Network Address: The first host address is **192.168.9.1**.

This network can be scanned as **192.168.9.1/24**

Network Scanner with scapy

We are going to manually build a network scanner in Python using the ARP protocol. We will build a custom ARP packet and display the results.

Before we start manually building a network scanner, we are going to use a built-in **scapy.arping()** method to show what our result will look like.

1. Create a Python program using a main function called **network_scanner1.py**
2. The first step is to import the **scapy** module.

```

1  #!/usr/bin/env python3
2  """
3      Name: network_scanner1.py
4      Author:
5      Created:
6      Purpose: Send an ARP packet to all hosts on a network
7  """
8
9  # Import the scapy module
10 import scapy.all as scapy

```

3. Define a **scan** function that accepts an **IP** address or **IP** address range as an argument. **scapy** uses the **arping** method to send an **arp** packet to all IP addresses in the range.

```
13 def scan(ip):
14     """
15         Send ARP packet to all hosts
16         Receive and display ARP information
17     """
18     # Call the scapy.arping() method using an
19     # IP address or IP range argument
20     scapy.arping(ip)
```

4. Call the **scan** function from **main()**. Replace the IP range with your own.

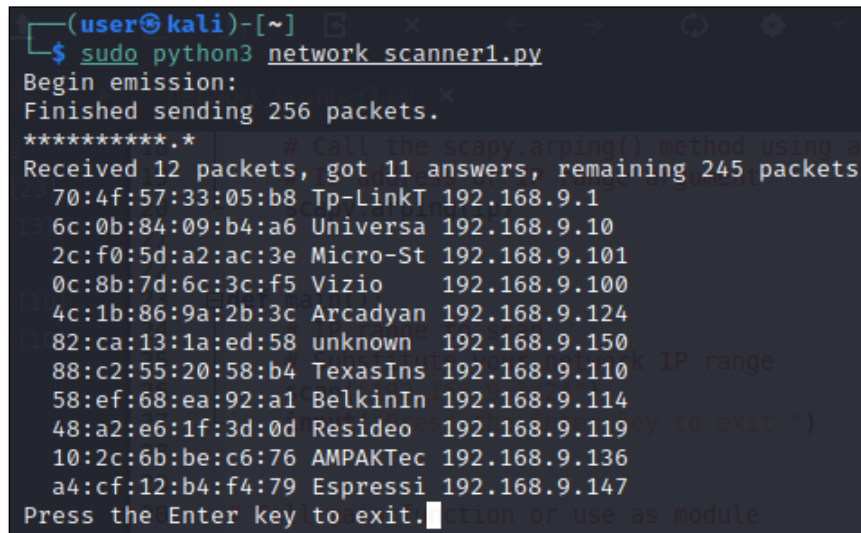
```
21 def main():
22     # IP range to scan
23     # Substitute your network IP range
24     scan("192.168.9.1/24")
25     input("Press the Enter key to exit.")
26
27
28 # Call main function or use as module
29 if __name__ == "__main__":
30     main()
```

5. The program will not work properly from an IDE. Start a **command prompt**.
6. Navigate to the folder that **network_scanner1.py** is located.
7. Windows: **python network_scanner1.py** at the command prompt as shown below
8. Linux: **sudo python3 network_scanner1.py** at a terminal prompt as shown below

Example run Windows:

```
Z:\_WNCC\Ethical Hacking\Assignments\Network Scanner>python network_scanner1.py
Begin emission:
Finished sending 256 packets.
*****
Received 10 packets, got 10 answers, remaining 246 packets
70:4f:57:33:05:b8 Tp-LinkT 192.168.9.1
6c:0b:84:09:b4:a6 Universa 192.168.9.10
2c:f0:5d:a2:ac:3e Micro-St 192.168.9.101
0c:8b:7d:6c:3c:f5 Vizio 192.168.9.100
4c:1b:86:9a:2b:3c Arcadyan 192.168.9.124
82:ca:13:1a:ed:58 unknown 192.168.9.150
88:c2:55:20:58:b4 TexasIns 192.168.9.110
58:ef:68:ea:92:a1 BelkinIn 192.168.9.114
5c:cf:7f:2c:31:9c Espressi 192.168.9.142
10:2c:6b:be:c6:76 AMPAKTec 192.168.9.136
Press the Enter key to exit._
```

Example run Linux:



```
(user@kali)-[~]  
$ sudo python3 network_scanner1.py  
Begin emission:  
Finished sending 256 packets.  
*****.* # Call the scapy arping() method using a  
Received 12 packets, got 11 answers, remaining 245 packets  
70:4f:57:33:05:b8 Tp-LinkT 192.168.9.1  
6c:0b:84:09:b4:a6 Universa 192.168.9.10  
2c:f0:5d:a2:ac:3e Micro-St 192.168.9.101  
0c:8b:7d:6c:3c:f5 Vizio 192.168.9.100  
4c:1b:86:9a:2b:3c Arcadyan 192.168.9.124  
82:ca:13:1a:ed:58 unknown 192.168.9.150  
88:c2:55:20:58:b4 TexasIns 192.168.9.110  
58:ef:68:ea:92:a1 BelkinIn 192.168.9.114  
48:a2:e6:1f:3d:0d Resideo 192.168.9.119  
10:2c:6b:be:c6:76 AMPAKTec 192.168.9.136  
a4:cf:12:b4:f4:79 Espressi 192.168.9.147  
Press the Enter key to exit.
```

Test the program on Windows and Linux.

That's it, we are done. We can use this simple hand-built network scanner on any network that you have permission to scan.

Assignment Submission

Attach all program files and screenshot of your results from both operating systems to the assignment in BlackBoard.