COMPUTER NETWORKS
INDUSTRY PROBLEM PROJECT – 1

Hita Juneja PES1UG20CS645
Ishita Bharadwaj PES1UG20CS648
Meghana N PES1UG20CS663

Port Scanning

A port scanner is an application designed to probe a server or host for open ports. Such an application may be used by administrators to verify security policies of their networks and by attackers to identify network services running on a host and exploit vulnerabilities.

A port scanner **sends a network request to connect to a specific TCP or UDP port on a computer and records the response**. So what a port scanner does is send a packet of network data to a port to check the current status.

```
from tkinter import font
import pyfiglet
import sys
import socket
from datetime import datetime

ascii_banner = pyfiglet.figlet_format("PORT SCANNER",font="digital")
print(ascii_banner)

# Defining a target
if len(sys.argv) == 2:

    # translate hostname to IPv4
    target = socket.gethostbyname(sys.argv[1])
else:
    print("Invalid amount of Argument")

# Add Banner
print("-" * 50)
print("Scanning Target: " + target)
print("Scanning started at:" + str(datetime.now()))
```

```
print("-" * 50)

try:

    # will scan ports between 1 to 65,535
    for port in range(1,65535):

        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        socket.setdefaulttimeout(1)

        # returns an error indicator
        result = s.connect_ex((target,port))
        if result ==0:
            print("Port {} is open".format(port))
        s.close()

except KeyboardInterrupt:
        print("\n Exiting Program !!!!")
        sys.exit()
except socket.gaierror:
        print("\n Hostname Could Not Be Resolved !!!!")
        sys.exit()
except socket.error:
        print("\ Server not responding !!!!")
        sys.exit()
```

```
(base) hitajuneja@Hitas-MacBook-Air Industry Problem Project % python port.py localhost
+-+-+-+-+ +-+-+-+-+-+-+-+-+
|P|O|R|T| |S|C|A|N|N|E|R|
+-+-+-+-+ +-+-+-+-+-+-+-+-+

-------------------------------------------------
Scanning Target: 127.0.0.1
Scanning started at:2022-04-17 17:07:10.852928
-------------------------------------------------
Port 5000 is open
Port 7000 is open
Port 46624 is open
Port 49152 is open
Port 49202 is open
(base) hitajuneja@Hitas-MacBook-Air Industry Problem Project %
```

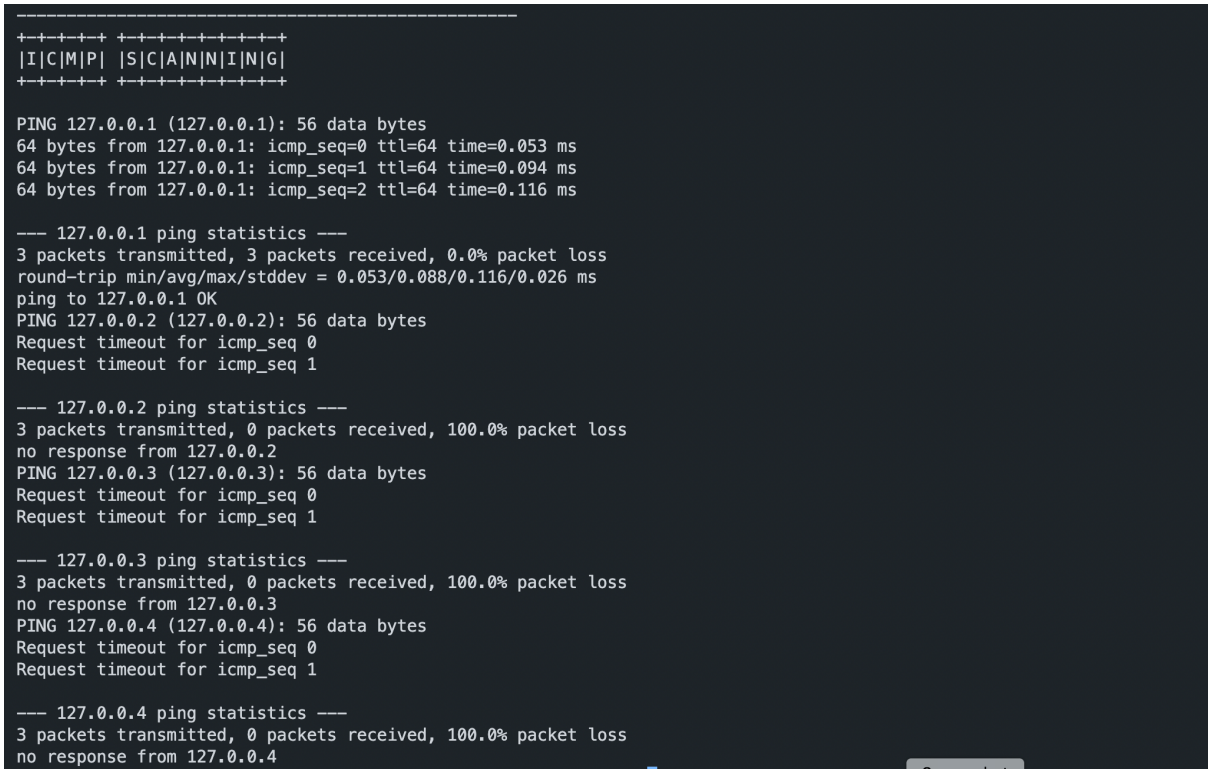
```
(base) hitajuneja@Hitas-MacBook-Air Industry Problem Project % python port.py 192.168.0.110
+-+-+-+-+ +-+-+-+-+-+-+-+
|P|O|R|T| |S|C|A|N|N|E|R|
+-+-+-+-+ +-+-+-+-+-+-+-+

_________________________________________________
Scanning Target: 192.168.0.110
Scanning started at:2022-04-17 17:07:41.622782
_________________________________________________
Port 5000 is open
Port 7000 is open
Port 49152 is open
Port 49202 is open
(base) hitajuneja@Hitas-MacBook-Air Industry Problem Project %
```

Network Scanning

A network scanner is one major tool for analyzing the hosts that are available on the network. A network scanner is an IP scanner that is used for scanning the networks that are connected to several computers.

1. ICMP Ping
   It is also known by using ‘ping command’.
   An ICMP packet is sent to a host using the IP address and if the ICMP echo is received, that means that the host is online and is receiving the signals.

```python
for ping in range(1,5):
    address = "127.0.0." + str(ping)
    res = subprocess.call(['ping', '-c', '3', address])
    #if res = 1 means destination is unreachable
    if res == 0:
        print( "ping to", address, "OK")
    elif res == 2:
        print("no response from", address)
    else:
        print("ping to", address, "failed!")
```



2. Network scanning refers to scanning of whole network to which we are connected and try to find out what are all the clients connected to our network. We can identify each and every client using their IP and MAC address. We can use ARP ping to find out the alive systems in our network.

   Scapy creates and sends a ARP packet. By default it is a request packet.

   ARP = Address resolution packet.
   ARP is **the protocol used to associate the IP address to a MAC address**. When a host wants to send a packet to another host, say IP address 10.5. 5.1, on its local area network (LAN), it first sends out (broadcasts) an ARP packet.

Steps for creating Network Scanner –

1. Create an ARP packet using ARP() method.
2. Set the network range using variable.
3. Create an Ethernet packet using Ether() method.
4. Set the destination to broadcast using variable hwdst.
5. Combine ARP request packet and Ethernet frame using '/'.
6. Send this to your network and capture the response from different devices.
7. Print the IP and MAC address from the response packets.

```
import scapy.all as scapy

request = scapy.ARP()
print("ARP Request Packet Summary: ",request.summary())
print("-" * 50)
print("ARP Request Packet .show(): ",request.show())
print("-" * 50)
print("LS Command ",scapy.ls(scapy.ARP()))
print("-" * 50)
request.pdst = '192.168.0.1/24'
broadcast = scapy.Ether() #creating ethernet packet

broadcast.dst = 'ff:ff:ff:ff:ff:ff'

request_broadcast = broadcast / request
#combining 2 layers, network layer (request IP packet) and link layer (broadcast
Ethernet frame)

clients = scapy.srp(request_broadcast, timeout = 10,verbose = 1)[0]
for element in clients:
    print(element[1].psrc + "    " + element[1].hwsrc)
```

```
(base) hitajuneja@Hitas-MacBook-Air Industry Problem Project % python wifi2.py
WARNING: No IPv4 address found on anpi0 !
WARNING: No IPv4 address found on anpi1 !
WARNING: more No IPv4 address found on en3 !
ARP Request Packet Summary:  ARP who has 0.0.0.0 says 192.168.0.110
--------------------------------------------------
###[ ARP ]###
  hwtype    = 0x1
  ptype     = IPv4
  hwlen     = None
  plen      = None
  op        = who-has
  hwsrc     = fc:e2:6c:12:a9:c4
  psrc      = 192.168.0.110
  hwdst     = 00:00:00:00:00:00
  pdst      = 0.0.0.0

ARP Request Packet .show():  None
--------------------------------------------------
hwtype     : XShortField                         = 1              ('1')
ptype      : XShortEnumField                     = 2048           ('2048')
hwlen      : FieldLenField                       = None           ('None')
plen       : FieldLenField                       = None           ('None')
op         : ShortEnumField                      = 1              ('1')
hwsrc      : MultipleTypeField (SourceMACField, StrFixedLenField) = 'fc:e2:6c:12:a9:c4' ('None')
psrc       : MultipleTypeField (SourceIPField, SourceIP6Field, StrFixedLenField) = '192.168.0.110' ('None')
hwdst      : MultipleTypeField (MACField, StrFixedLenField) = '00:00:00:00:00:00' ('None')
pdst       : MultipleTypeField (IPField, IP6Field, StrFixedLenField) = '0.0.0.0'       ('None')
LS Command  None
--------------------------------------------------
Begin emission:
Finished sending 256 packets.

Received 3200 packets, got 2 answers, remaining 254 packets
192.168.0.1      38:6b:1c:57:9b:12
192.168.0.110    fc:e2:6c:12:a9:c4
(base) hitajuneja@Hitas-MacBook-Air Industry Problem Project %
```

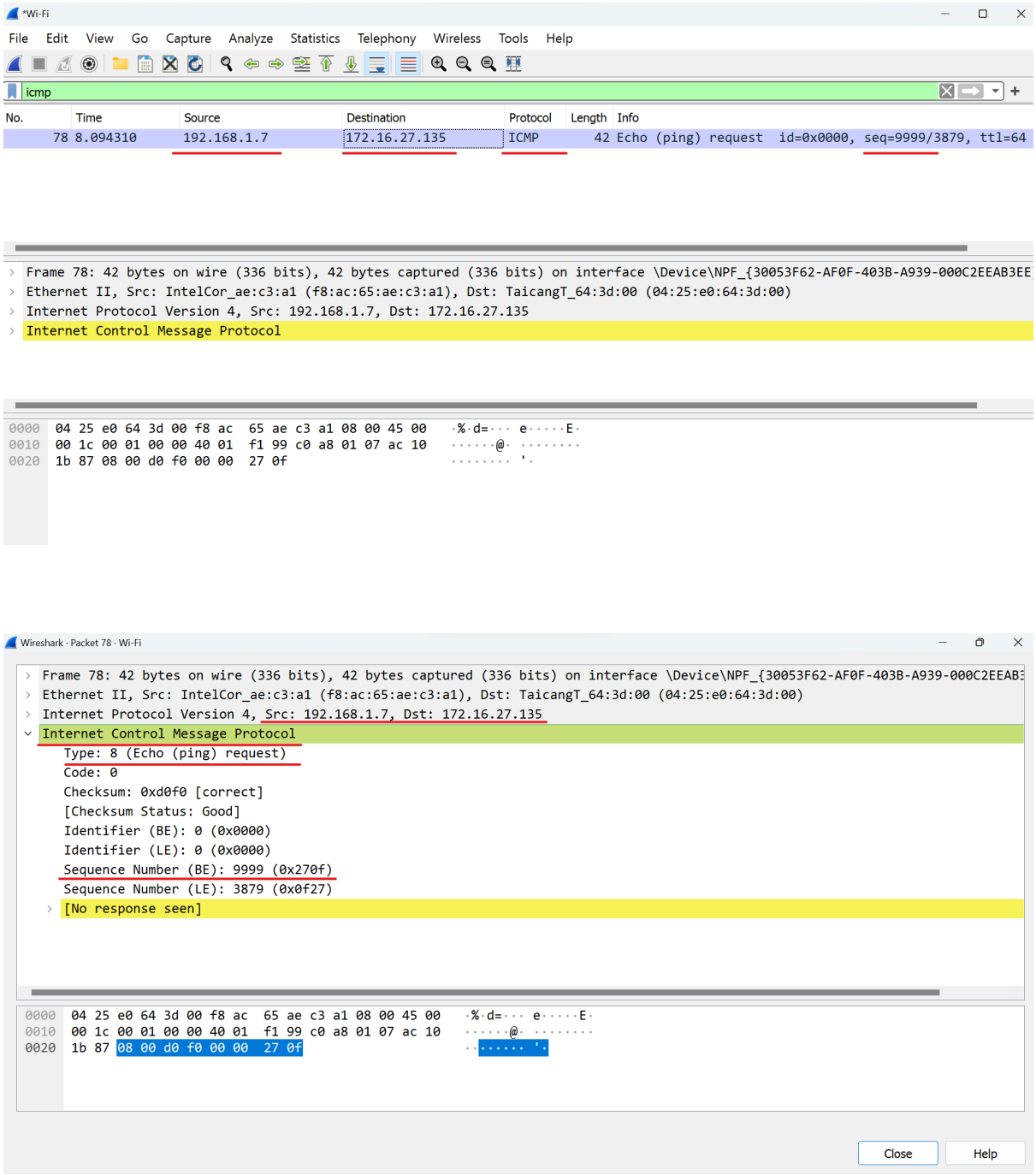
Packet Crafting with Scapy: ICMP Echo Request

Scapy is a *tool (python program) that enables the user to send, sniff and dissect and forge network packets. This capability allows construction of tools that can probe, scan or attack networks.*

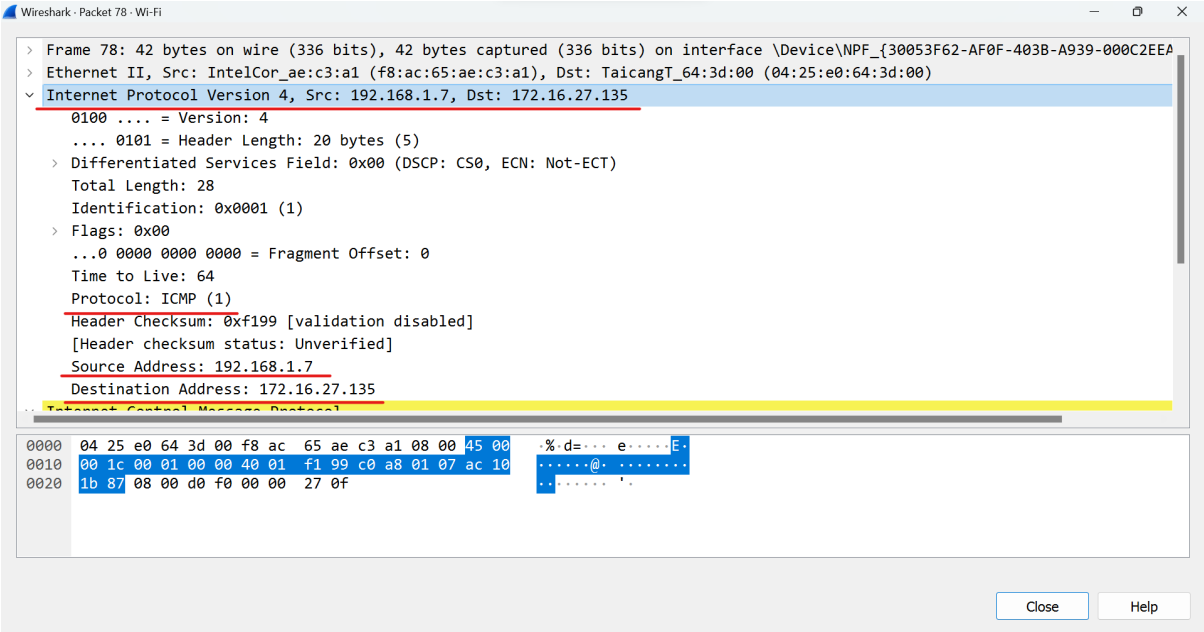
ip_layer = IP (dst=”172.16.27.135” - specifies the destination IP address to which our request will be sent.
icmp_layer= ICMP (seq=9999) – specifies the sequence number of our ICMP packet being sent.
Packet = ip_layer/icmp_layer – combines the icmp_layer and ip_layer.
Send(packet) – sends the packet.

```python
#! /usr/bin/env python
from scapy.all import *

ip_layer = IP(dst="172.16.27.135")
icmp_layer = ICMP(seq=9999)
packet = ip_layer / icmp_layer
send(packet)
```

Wireshark Packet Capture:



ICMP Echo (ping) request packet: ICMP header (zoom)



IP header (zoom)